Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Магнитогорский государственный технический университет им. Г.И. Носова»

Многопрофильный колледж

УТВЕРЖДАЮ
Директор
_____ / С.А. Махновский
«09» февраля 2022 г.

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ВЫПОЛНЕНИЮ ЛАБОРАТОРНЫХ РАБОТ

ПМ.01 Разработка модулей программного обеспечения для компьютерных систем

для обучающихся специальности

09.02.07 Информационные системы и программирование
Квалификация: Программист

Магнитогорск, 2022

ОДОБРЕНО:

ОДОБРЕНО:

Предметной / Предметно-цикловой комиссией «Информатики и вычислительной техники»
Председатель И.Г. Зорина
Протокол № __5____ от __19.01.2022г._____

Методической комиссией МпК
Протокол № _4_ от __09.02.2022г.____

Разработчики:

преподаватель ФГБОУ ВО «МГТУ им. Г.И. Носова» Многопрофильный колледж
Л.А.Фетисова

преподаватель ФГБОУ ВО «МГТУ им. Г.И. Носова» Многопрофильный колледж
В.Д. Тутарова

Методические указания по выполнению практических и лабораторных работ разработаны на основе рабочей программы профессионального модуля ПМ 1. «РАЗРАБОТКА МОДУЛЕЙ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ КОМПЬЮТЕРНЫХ СИСТЕМ».

Содержание практических и лабораторных работ ориентировано на формирование общих и профессиональных компетенций по программе подготовки специалистов среднего звена по специальности 09.02.07 Информационные системы и программирование (квалификация Программист).

Содержание

1 ВВЕДЕНИЕ

Важную часть теоретической и профессиональной практической подготовки обучающихся составляют практические и лабораторные занятия.

Состав и содержание практических и лабораторных работ направлены на реализацию действующих федеральных государственных образовательных стандартов среднего профессионального образования.

Ведущей дидактической целью практических занятий является формирование профессиональных практических умений (умений выполнять определенные действия, операции, необходимые в последующем в профессиональной деятельности) или учебных практических умений (умений решать задачи по математике, физике, информатике и др.), необходимых в последующей учебной деятельности.

Ведущей дидактической целью лабораторных занятий является экспериментальное подтверждение и проверка существенных теоретических положений (законов, зависимостей).

В соответствии с рабочей программой профессионального модуля ПМ.01 «Разработка модулей программного обеспечения для компьютерных систем», МДК 01.01 Разработка программных модулей; МДК.01.02 Поддержка и тестирование программных модулей; МДК. 01.03 Разработка мобильных приложений и МДК. 01.04 Системное программирование предусмотрено проведение лабораторных работ. В рамках лабораторного занятия обучающиеся могут выполнять одну или несколько лабораторных работ.

В результате их выполнения, обучающийся должен:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.

Содержание практических и лабораторных работ ориентировано на подготовку студентов к освоению профессионального модуля основной профессиональной образовательной программы по специальности и овладению профессиональными компетенциями:

ПК 1.1. Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;

ПК 1.2. Разрабатывать программные модули в соответствии с техническим заданием;

ПК.1.3. Выполнять отладку программных модулей с использованием специализированных программных средств;

ПК 1.4. Выполнять тестирование программных модулей;

ПК 1.5. Осуществлять рефакторинг и оптимизацию программного кода;

ПК 1.6. Разрабатывать модули программного обеспечения для мобильных платформ.

А также формированию общих компетенций:

ОК 01. Выбирать способы решения задач профессиональной деятельности, применительно к различным контекстам.

ОК 02. Осуществлять поиск, анализ и интерпретацию информации, необходимой для выполнения задач профессиональной деятельности.

ОК 03. Планировать и реализовывать собственное профессиональное и личностное развитие.

ОК 04. Работать в коллективе и команде, эффективно взаимодействовать с коллегами, руководством, клиентами.

ОК 05. Осуществлять устную и письменную коммуникацию на государственном языке с учетом особенностей социального и культурного контекста.

ОК 06. Проявлять гражданско-патриотическую позицию, демонстрировать осознанное поведение на основе традиционных общечеловеческих ценностей, применять стандарты антикоррупционного поведения.

ОК 07. Содействовать сохранению окружающей среды, ресурсосбережению, эффективно действовать в чрезвычайных ситуациях.

ОК 08. Использовать средства физической культуры для сохранения и укрепления здоровья в процессе профессиональной деятельности и поддержания необходимого уровня физической подготовленности.

ОК 09. Использовать информационные технологии в профессиональной деятельности.

ОК 10. Пользоваться профессиональной документацией на государственном и иностранном языках.

ОК 11. Использовать знания по финансовой грамотности, планировать предпринимательскую деятельность в профессиональной сфере.

Выполнение обучающихся практических лабораторных работ по профессиональному модулю ПМ.01 «Разработка модулей программного обеспечения для компьютерных систем», МДК 01.01 Разработка программных модулей; МДК.01.02 Поддержка и тестирование программных модулей; МДК. 01.03 Разработка мобильных приложений и МДК. 01.04 Системное программирование направлено на:

- обобщение, систематизацию, углубление, закрепление, развитие и детализацию полученных теоретических знаний по конкретным темам учебной дисциплины;
- формирование умений применять полученные знания на практике, реализацию единства интеллектуальной и практической деятельности;
- развитие интеллектуальных умений у будущих специалистов: аналитических, проектировочных, конструктивных и др.;
- выработку при решении поставленных задач профессионально значимых качеств, таких как самостоятельность, ответственность, точность, творческая инициатива.

Продолжительность выполнения лабораторной работы составляет не менее двух академических часов и проводится после соответствующей темы, которая обеспечивает наличие знаний, необходимых для ее выполнения.

2 МЕТОДИЧЕСКИЕ УКАЗАНИЯ

МДК.01.01 Разработка программных модулей
Тема:1.1.2 Структурное программирование
Лабораторная работа № 1 Оценка сложности алгоритмов сортировки.

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Разрабатывать программные модули в соответствии с техническим заданием;

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценивать сложность алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Задание:

1. **Подсчет операций. Классы входных данных**

Одним из способов оценки трудоемкости ($T\,n$) является *подсчет количества выполняемых операций*. Рассмотрим в качестве примера алгоритм поиска минимального элемента массива.

1. начало; поиск минимального элемента массива array из N элементов
2. min := array[1]
3. для i от 2 до N выполнять:
4. если array[i] < min
5. min := array[i]
6. конец; вернуть min

При выполнении этого алгоритма будет выполнена:

1. N — 1 операция присваивания счетчику цикла i нового значения;
2. N — 1 операция сравнения счетчика со значением N;
3. N — 1 операция сравнения элемента массива со значением min;
4. от 1 до N операций присваивания значения переменной min.

Точное количество операций будет зависеть от обрабатываемых данных, поэтому имеет смысл говорить о наилучшем, наихудшем и среднем случаях. При этом худшему случаю всегда уделяется особое внимание, в том числе потому, что «плохие» данные могут быть намеренно поданы на вход злоумышленником.

Понятие *среднего случая* используется для оценки поведения алгоритма с расчетом на то, что наборы данных равновероятны. Однако, такая оценка достаточно сложна:

1. исходные данные разбиваются на группы так, что трудоемкость алгоритма ($t\,i$) для любого набора данных одной группы одинакова;
2. исходя из доли наборов данных группы в общем числе наборов, рассчитывается вероятность для каждой группы ($p\,i$);
3. оценка среднего случая вычисляется по формуле: $\sum i{=}1\ m\ p\,i \cdot t\,i$.

2. **Алгоритм пузырьковой сортировки (bubble sort)** использует два вложенных цикла. Во внутреннем последовательно сравниваются пары элементов и если

оказывается, что элементы стоят в неправильном порядке — выполняется перестановка. Внешний цикл выполняется до тех пор, пока в массиве найдется хоть одна пара элементов, нарушающих требуемый порядок.

1. начало; пузырьковая сортировка массива array из N элементов
2. nPairs := N; количество пар элементов
3. hasSwapped := false; пока что ни одна пара не нарушила порядок
4. для всех i от 1 до nPairs-1 выполнять:
5. если array[i] > array[i+1] то:
6. swap(array[i], array[i+1]); обменять элементы местами
7. hasSwapped := true; найдена перестановка
8. nPairs := nPairs - 1; наибольший элемент гарантированно помещен в конец
9. если hasSwapped = true - то перейти на п.3
10. конец; массив array отсортирован

Трудоемкость функции swap не зависит от количества элементов в массиве, поэтому оценивается как $T_{swap} = \Theta(1)$. В результате выполнения внутреннего цикла, наибольший элемент смещается в конец массива неупорядоченной части, поэтому через N таких вызовов массив в любом случае окажется отсортирован. Если же массив отсортирован, то внутренний цикл будет выполнен лишь один раз.

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Написать и отладить программу.

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Лабораторная работа № 2 Оценка сложности алгоритмов поиска.

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценивать сложность алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Выполнять тестирование программных модулей.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Задание:

1. **Алгоритм линейного поиска**

Линейный поиск (Linear search) (Последовательный поиск, полный перебор, брутфорс)
Последовательный просмотр каждого элемента последовательности, пока не найден нужный. Выполняется за один проход цикла, сложность, соответственно – O(n).
Двоичный поиск (Binary search) (Бинарный поиск, метод деления пополам)
Поиск элемента в отсортированном массиве.
Суть алгоритма – последовательные запросы с предлагаемым вероятным ответом. Ответ на запрос может быть «нужный элемент меньше», «нужный элемент больше» или «это ответ!». Первым предлагается серединный элемент массива. После получения ответа алгоритм отбрасывает половину массива, содержащую неподходящие значения, и начинает поиск на оставшейся половине.
Троичный поиск (Ternary search) (Тернарный поиск)
Используется для поиска максимума функции, которая строго возрастает, а потом убывает, либо наоборот.
Если известно, что ответ лежит между точками A и B, на этом отрезке выбираются некоторые точки a и b (обычно делящие отрезок на три равные части). Проверяют, на какой из этих двух точек функция принимает меньшее значение. Крайняя треть, отделяемая той точкой, отбрасывается, и поиск продолжается на двух оставшихся третях.
В соответствии с этим алгоритмом эталонный массив просматривается последовательно от первого до последнего элемента. Наиболее сложным, как уже отмечалось, является случай, когда аргумента (ключа) нет в таблице (не найден).
Уточненный алгоритм будет таким.
1. Ввести исходный массив (ключей).
2. Выполнять
2. 1. Ввести аргумент поиска (целое число);
2. 2. Если аргумент поиска больше или равен нулю,
2.2.1. Результат (номер в массиве, num) = - 1 (не найден, такого номера нет);
2.2.2. Для индекса массива (i) от 0 до Длина.массива Если аргумент поиска = ключ[i], номер в массиве (num) = i;
2.2.3. Если номер num = - 1, вывести: «Такого ключа в массиве нет», Иначе вывести: «Ключ найден под номером num». Пока будет аргумент поиска больше или равен нулю.
3. Закончить.

Основной недостаток алгоритма линейного поиска – большое время. Он предполагает использование оператора For в пункте 2.2.2.
При этом ВСЕГДА выполняется ровно n операций сравнения, не зависимо от того, найден ключ или нет. Программа, обнаружив аргумент в начале массива, продолжает его просмотр до конца, т.е. выполняет бесполезную работу. Время поиска может быть

существенно сокращено, если обеспечить его прекращение, когда ключ найден. При равномерном распределении элементов в таблице эталонов (ключей) среднее время поиска может стать пропорциональным величине n / 2.
Этого можно достичь, изменив пункт 2.2 в рассмотренном выше алгоритме следующим образом.
Ускорение линейного поиска
2. Выполнять
2. 1. Ввести аргумент поиска (целое число);
2. 2. Если аргумент поиска больше или равен нулю,
2.2.1. Результат (num) = - 1 (не найден);
2.2.2. Начальный индекс, i=0;
2.2.3. Пока (num=-1) и (i<=n – Длины.массива)
a) Если аргумент поиска = ключ[i], номер в массиве (num) = i;
b) i=i + 1
2.2.4. Если номер num = - 1,
вывести: «Такого ключа в массиве нет», Иначе вывести: «Ключ num найден». Пока будет аргумент поиска больше или равен нулю.

Трудоемкость (временная сложность) алгоритма линейного поиска определяется числом операций сравнения, выполняемых при просмотре таблицы эталонов. В лучшем случае количество таких операций равно 1, в худшем – n, а в среднем, если возможные значения ключей равновероятны, - n / 2. Таким образом, асимптотическая оценка O(n) = n.

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Написать и отладить программу.

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Лабораторная работа № 3 Оценка сложности рекурсивных алгоритмов.

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;

- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценивать сложность алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Выполнять тестирование программных модулей.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Задание:

- Создайте программы, реализующие рекурсивный и итеративный алгоритмы вычисления n-го числа Фибоначчи. Сравните время их работы для n = 5, 25, 45, 65, 85, 100. Чтобы избежать переполнения, для n < 100 используйте беззнаковое представления чисел размером в 8 байт (тип unsigned long в языке Си). Результаты сравнения оформите в виде таблицы и в виде графика.
- Проанализируйте вычислительную сложность алгоритмов умножения чисел. На основе анализа поведите сравнения данных алгоритмов.
- Разработать два алгоритма возведения числа в целую неотрицательную степень, различающиеся по сложности. Для разработанных алгоритмов определите вычислительную сложность алгоритма.
- Разработать два алгоритма возведения числа в целую неотрицательную степень, различающиеся по сложности. Для разработанных алгоритмов проведите сравнительный анализ.
- Проанализируйте пространственную сложность алгоритмов умножения чисел. На основе анализа поведите сравнения данных алгоритмов.
- Создайте программы, реализующие рекурсивный и итеративный алгоритмы вычисления n-го числа Фибоначчи. Сравните время их работы для n = 5, 25, 45, 65, 85, 100. Чтобы избежать переполнения, для n < 100 используйте беззнаковое представления чисел размером в 8 байт(тип unsigned long в языке Си). Результаты сравнения оформите в виде таблицы и в виде графика.
- Разработать два алгоритма возведения числа в целую неотрицательную степень, различающиеся по сложности. Для разработанных алгоритмов определите вычислительную сложность алгоритма.
- Проанализируйте вычислительную сложность алгоритмов умножения чисел. На основе анализа поведите сравнения данных алгоритмов.
- Разработать два алгоритма возведения числа в целую неотрицательную степень, различающиеся по сложности. Для разработанных алгоритмов определите вычислительную сложность алгоритма.
- Разработать два алгоритма возведения числа в целую неотрицательную степень, различающиеся по сложности. Для разработанных алгоритмов проведите сравнительный анализ.

В целях дальнейшего анализа примем следующие допущения: каждая команда выполняется не более чем за фиксированное время; исходные данные алгоритма представляются машинными словами по битов каждое.

Конкретная проблема задается N словами памяти, таким образом, на входе алгоритма – N = N* бит информации.
Программа, реализующая алгоритм для решения общей проблемы состоит из M машинных инструкций по м битов – M = M* м бит информации.
Кроме того, алгоритм может требовать следующих дополнительных ресурсов абстрактной машины:

- Sd – память для хранения промежуточных результатов;
- Sr – память для организации вычислительного процесса (память, необходимая для реализации рекурсивных вызовов и возвратов).

При решении конкретной проблемы, заданной N словами памяти алгоритм выполняет не более, чем конечное количество «элементарных» операций абстрактной машины в силу условия рассмотрения только финитных алгоритмов. В связи с этим введем следующее определение:
Под трудоёмкостью алгоритма для данного конкретного входа – Fa(N), будем понимать количество «элементарных» операций совершаемых алгоритмом для решения конкретной проблемы в данной формальной системе.
Комплексный анализ алгоритма может быть выполнен на основе комплексной оценки ресурсов формальной системы, требуемых алгоритмом для решения конкретных проблем. Очевидно, что для различных областей применения веса ресурсов будут различны, что приводит к следующей комплексной оценке алгоритма:
c1 * Fa(N) + c2 * + c3 * Sd + c4 * Sr, где ci – веса ресурсов.
При более детальном анализе трудоемкости алгоритма оказывается, что не всегда количество элементарных операций, выполняемых алгоритмом на одном входе длины N, совпадает с количеством операций на другом входе такой же длины. Это приводит к необходимости введения специальных обозначений, отражающих поведение функции трудоемкости данного алгоритма на входных данных фиксированной длины.
Пусть DA – множество конкретных проблем данной задачи, заданное в формальной системе. Пусть D DA – задание конкретной проблемы и |D| = N.
В общем случае существует собственное подмножество множества DA, включающее все конкретные проблемы, имеющие мощность N:

- обозначим это подмножество через DN: DN = {D DN,: |D| = N};
- обозначим мощность множества DN через MDN → MDN = |DN |. Тогда содержательно данный алгоритм, решая различные задачи размерности N, будет выполнять в каком-то случае наибольшее количество операций, а в каком-то случае наименьшее количество операций. Ведем следующие обозначения:

1. Fa(N) – худший случай – наибольшее количество операций, совершаемых алгоритмом A для решения конкретных проблем размерностью N:
DDN Fa(N) = max {Fa (D)} – худший случай на DN
2. Fa(N) – лучший случай – наименьшее количество операций, совершаемых алгоритмом A для решения конкретных проблем размерностью N:
DDN Fa(N) = min {Fa (D)} – лучший случай на DN
3. Fa(N) – средний случай – среднее количество операций, совершаемых алгоритмом A для решения конкретных проблем размерностью N:
DDNFa(N) = (1 / MDN)* {Fa (D)} – средний случай на DN.
В зависимости от влияния исходных данных на функцию трудоемкости алгоритма может быть предложена следующая классификация, имеющая практическое значение для анализа алгоритмов: Количественно - зависимые по трудоемкости алгоритмы
Это алгоритмы, функция трудоемкости которых зависит только от размерности конкретного входа, и не зависит от конкретных значений:
Fa (D) = Fa (|D|) = Fa (N). Примерами алгоритмов с количественно-зависимой функцией трудоемкости могут служить алгоритмы для стандартных операций с массивами и матрицами – умножение матриц, умножение матрицы на вектор и т.д. Параметрически -

зависимые по трудоемкости алгоритмы
Это алгоритмы, трудоемкость которых определяется не размерностью входа (как правило, для этой группы размерность входа обычно фиксирована), а конкретными значениями обрабатываемых слов памяти:
Fa (D) = Fa (d1,…,dn) = Fa (P1,…,Pm), m n
Краткие теоретические сведения:
Примерами алгоритмов с параметрически-зависимой трудоемкостью являются алгоритмы вычисления стандартных функций с заданной точностью путем вычисления соответствующих степенных рядов. Очевидно, что такие алгоритмы, имея на входе два числовых значения – аргумент функции и точность выполняют существенно зависящее от значений количество операций.
а) Вычисление xk последовательным умножением Fa(x, k) = Fa (k).
б) Вычисление ex=(xn/n!), с точностью до Fa = Fa (x,)
Количественно-параметрические по трудоемкости алгоритмы
Однако в большинстве практических случаев функция трудоемкости зависит как от количества данных на входе, так и от значений входных данных, в этом случае:
Fa (D) = Fa (||D||, P1,…,Pm) = Fa (N, P1,…,Pm)
В качестве примера можно привести алгоритмы численных методов, в которых параметрически-зависимый внешний цикл по точности включает в себя количественно-зависимый фрагмент по размерности.
Порядково - зависимые по трудоемкости алгоритмы
Среди разнообразия параметрически - зависимых алгоритмов выделим еще оду группу, для которой количество операций зависит от порядка расположения исходных объектов.
Пусть множество D состоит из элементов (d1,…,dn), и ||D||=N,
Определим Dp = {(d1,…,dn)}-множество всех упорядоченных N-ок из d1,…,dn, отметим, что |Dp|=n!.
Если Fa (iDp) Fa (jDp), гдеiDp, jDp Dp, то алгоритм будем называть порядково-зависимым по трудоемкости. Примерами таких алгоритмов могут служить ряд алгоритмов сортировки, алгоритмы поиска минимума и максимума в массиве.
Рассмотрим более подробно алгоритм поиска максимума в массиве S, содержащим n элементов:
MaxS (S,n; Max)
Max S1
For i2 to n
if Max < Si then Max Si (количество выполненных операций присваивания зависит от порядка следования элементов массива)
При анализе поведения функции трудоемкости алгоритма часто используют принятые в математике асимптотические обозначения, позволяющие показать скорость роста функции, маскируя при этом конкретные коэффициенты.
Такая оценка функции трудоемкости алгоритма называется сложностью алгоритма и позволяет определить предпочтения в использовании того или иного алгоритма для больших значений размерности исходных данных.
В асимптотическом анализе приняты следующие обозначения [6]:
Пусть f(n) и g(n) – положительные функции положительного аргумента, n ≥ 1 (количество объектов на входе и количество операций – положительные числа), тогда:
n0
f,gnc2g(n)
f(n)
c1g(n)
f(n) = (g(n)),
если существуют положительные
c1, c2, n0, такие, что:

c1 * g(n) f(n) c2 * g(n),
при n > n0
Обычно говорят, что при этом функция g(n) является асимптотически точной оценкой функции f(n), т.к. по определению функция f(n) не отличается от функции g(n) с точностью до постоянного множителя.
Отметим, что из f(n) = (g(n)) следует, что g(n) = (f(n)).
Примеры:
1) f(n)=4n2+nlnN+174 – f(n)= (n2);
2) f(n)=(1) – запись означает, что f(n) или равна константе, не равной нулю, или f(n) ограничена константой на : f(n) = 7+1/n = (1).

В отличие от оценки , оценка O требует только, что бы функция f(n) не превышала g(n) начиная с n > n0, с точностью до постоянного множителя:
cg(n)
f(n)
c > 0, n0 > 0 :
0 f(n) c * g(n), n n0
Вообще, запись O(g(n)) обозначает класс функций, таких, что все они растут не быстрее, чем функция g(n) с точностью до постоянного множителя, поэтому иногда говорят, что g(n) мажорирует функцию f(n).
Например, для всех функций:
f(n)=1/n, f(n)= 12, f(n)=3*n+17, f(n)=n*Ln(n), f(n)=6*n2 +24*n+77
будет справедлива оценка O(n2)Указывая оценку O есть смысл указывать наиболее «близкую» мажорирующую функцию, поскольку например для f(n)= n2 справедлива оценка O(2n), однако она не имеет практического смысла.
В отличие от оценки O, оценка является оценкой снизу – т.е. определяет класс функций, которые растут не медленнее, чем g(n) с точностью до постоянного множителя:
F(n)
cg(n)
c > 0, n0 >0 :
0 c * g(n) f(n)
Например, запись (n*Ln(n)) обозначает класс функций, которые растут не медленнее, чем g(n) = n*Ln(n), в этот класс попадают все полиномы со степенью большей единицы, равно как и все степенные функции с основанием большим единицы.
Отметим, что не всегда для пары функций справедливо одно из асимптотических соотношений, например для f(n)=n1+sin(n) и g(n)=n не выполняется ни одно из асимптотических соотношений.
В асимптотическом анализе алгоритмов разработаны специальные методы получения асимптотических оценок, особенно для класса рекурсивных алгоритмов. Очевидно, что оценка является более предпочтительной, чем оценка O. Знание асимптотики поведения функции трудоемкости алгоритма - его сложности, дает возможность делать прогнозы по выбору более рационального с точки зрения трудоемкости алгоритма для больших размерностей исходных данных.

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Написать и отладить программу.

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Лабораторная работа № 4,5 Оценка сложности эвристических алгоритмов.

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценивать сложность алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Выполнять тестирование программных модулей.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Краткие теоретические сведения:

Эвристический алгоритм — это алгоритм решения задачи, правильность которого для всех возможных случаев не доказана, но про который известно, что он даёт достаточно хорошее решение в большинстве случаев.

Наиболее частые эвристики:

- Жадный алгоритм
- Ограниченный перебор только перспективных вариантов
- Последовательное улучшение («локальный поиск»)

Жадный алгоритм — однопроходный итерационный алгоритм. Строит решение, добавляя на каждом шаге к текущему *частичному* решению новый элемент. Добавляемый элемент выбирается на основе локального оптимума («наилучший на текущем шаге»).

Жадные алгоритмы:

- Дают точное решение для задач на матроидах (с аддитивной целевой функцией)
- некоторых задач дают приближённое решение с гарантированной (не обязательно константной) оценкой приближения
- В общем случае используются как эвристики

Задание:

1. Некий путешественник задумал повторить путь А.Н. Радищева "Из Петербурга в Москву", для чего решил воспользоваться личным автомобилем. Путешественник решил так спланировать свой маршрут, чтобы минимизировать затраты. Для этого ему нужно было знать, сколько на его пути встретится заправочных станций и на каком расстоянии друг от друга они находятся. Помимо всего прочего приходилось учитывать, что емкость бензобака машины ограничена. Требуется определить, какое минимальное количество заправок ему нужно посетить.

Считать, что первая АЗС находится в Петербурге, а последняя в Москве.
Di - расстояние от i-ой до (i+1)-ой заправки;
S - расстояние, которое машина может проехать с полным баком;
L - расстояние от Петербурга до Москвы.
В ходе решения задачи нужно получить номера заправок, на которых придётся заправляться.
Решение:
В данной задаче жадный алгоритм будет работать таким образом:

1. Путешественник заправляет полный бак в исходной точке маршрута(Петербурге);
2. Если от данной заправки до Москвы бензина достаточно, то едем в Москву; иначе находим самую удалённую заправку, до которой можно доехать и едем туда, заправляем полные баки и повторяем шаг 2.

2. Перед праздниками шеф получает очень много приглашений на торжественные заседания. Чтобы лучше планировать свое время, шеф ввел правило, чтобы в каждом из приглашений четко указывался отрезок времени заседаний. Шеф не любит половинчатых решений, поэтому или находится на заседании все указанное время, или не приходит на него. Между посещениями заседаний должен быть хотя бы минимальный перерыв, то есть шеф может успеть на j-е заседание по списку приглашений, если. Напишите программу, позволяющую шефу посетить как можно больше заседаний.

Решение:
В этой задаче правильным оказывается неожиданно простой жадный алгоритм: на первом шаге выбрать заседание с наименьшим значением b, на каждом следующем шаге- с наименьшим значением b, но только среди тех заседаний, которые начинаются после конца предыдущего выбранного.
Для реализации представленного алгоритма сначала отсортируем массив интервалов времени по неубыванию значений b. Поскольку для окончательного ответа нужны исходные номера заседаний, целесообразно организовать данные в записи с полями a, b, idx (границы интервала заседания и его номер в начальном списке) и сортировать записи по значениям в поле b.

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Написать и отладить программу.

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Тема 1.1.3 Объектно-ориентированное программирование
Лабораторная работа № 6 Работа с классами. Перегрузка методов.

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Задание:

1. Определить класс «Вектор в трехмерном пространстве».

Реализовать в виде класса методы для выполнения следующих операций над векторами:

- вывод координат вектора;
- получение длины вектора;
- сложение векторов;
- скалярное произведение векторов;
- определение угла между векторами;

Сложение векторов реализовать в виде перегрузки операции «+», взятие модуля вектора реализовать как результат приведения объекта класса к типу double.

В программе продемонстрировать использование объектов класса «Вектор в трехмерном пространстве»

Перегрузка операции присваивания реализована в классе vector3d для того, чтобы в выражении `c=a+b` вектор *c* не потерял свое имя. Если не определять перегрузку операции присваивания, то результат сложения в виде временно созданного в функции *operator+* объекта с именем вектора "t_copy" будет присвоен вектору *c* и поэлементно изменит значение всех его компонентных данных, в том числе и имя вектора. Чтобы вектор не сменил имя, а только получил новые координаты, перегружена операция присваивания.

2. Создать внешний вид формы программы
3. Написать программные коды для созданной формы
4. Запустить и отладить программу

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Написать и отладить программу.

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Лабораторная работа № 7 Определение операций в классе. Создание наследованных классов.

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.

- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Задание:

1. Разработать перечисленные ниже классы.

При разработке каждого класса возможны два варианта решения:

а) данные-члены класса представляют собой переменные и массивы фиксированной размерности;

б) память для данных-членов класса выделяется динамически.

- «**Комплексное число**» – **Complex**. Класс должен содержать несколько конструкторов и операции для сложения, вычитания, умножения, деления, присваивания. Создать два вектора размерности *n* из комплексных координат. Передать их в функцию, которая выполняет сложение комплексных векторов.
- Разработать класс **«Многочлен» – Polynom** степени *n* . Написать несколько конструкторов, в том числе конструктор копирования. Реализовать методы для вычисления значения полинома; сложения,

вычитания и умножения полиномов. Перегрузить операции сложения, вычитания, умножения, инкремента, декремента, индексирования, присваивания. Создать массив объектов класса. Передать его в функцию, вычисляющую сумму полиномов массива и возвращающую полином-результат, который выводится на экран в головной программе.

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Создать внешний вид формы программы
- Написать программные коды для созданной формы
- Запустить и отладить программу
- Написать и отладить программу.

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой

обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Лабораторная работа № 8 Работа с объектами через интерфейсы. Использование стандартных интерфейсов.

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Задание:

1. Разработать приложение исходя их следующих требований:

1.1. Приложение должно состоять минимум из двух форм. Первая – для работы со списком объектов, вторая – для работы с содержимым самого объекта

1.2. На форме должны быть предусмотрены операции “Добавления”, “Удаления”, “Изменения”, “Сортировки” и “Поиска” элементов списка.

1.3. Операции сортировки должны выполняться с использованием метода Sort() шаблонного класса List<T>. Количество вариантов сортировки – не менее 3.

1.4. Операция поиска выполняется только по ключевому атрибуту, с использование возможностей класса Dictionary<T, T>.

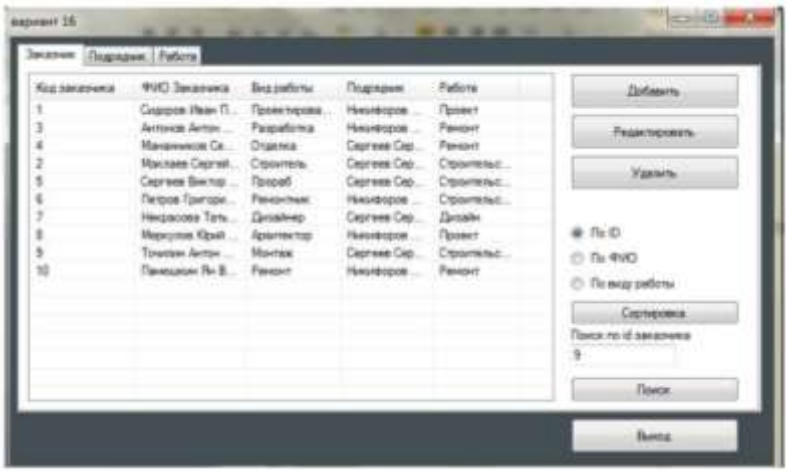
Любой класс или структура, реализующие интерфейс IEquatable<T>, должны содержать определение метода Equals, который соответствует сигнатуре, определяемой интерфейсом. В результате можно рассчитывать, что любой класс, реализующий интерфейс IEquatable<T>, будет содержать метод Equals, с которым экземпляр класса может определить, равен ли он другому экземпляру класса.

Для реализации члена интерфейса соответствующий член класса должен быть открытым, нестатическим, и иметь то же имя и сигнатуру, что и член интерфейса.

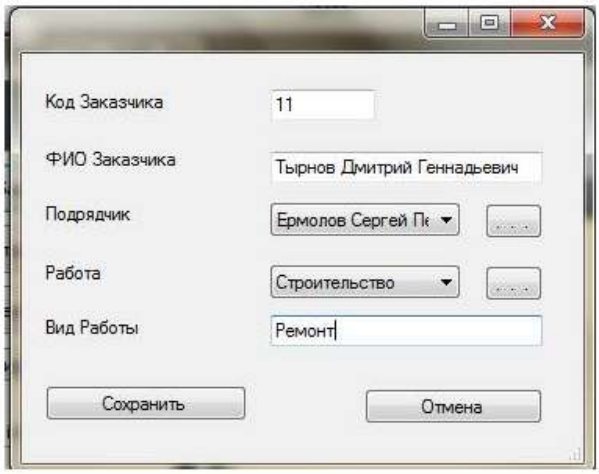
Когда класс или структура реализуют интерфейс, класс или структура должны обеспечивать реализацию всех членов которые определяет интерфейс. Сам интерфейс не предоставляет никакой функциональности, класс или структура может наследовать таким способом, она может наследовать функциональность базового класса. Однако если базовый класс реализует интерфейс, то любой класс, производный от базового класса наследует ее реализацию.

Работа с коллекциями объектов, класс List<T>.

Шаблонный класс List<T> является удобным механизмом для работы со списком однотипных объектов и представляет строго типизированный список объектов, доступных по индексу. Поддерживает методы для поиска по списку, выполнения сортировки и других операций со списками.
Сортировка элементов списка осуществляется с помощью метода Sort(). Если тип элементов списка является системным, то это означает в частности, что данные типы уже реализуют интерфейс IEquatable<T> (метод Equals()) и интерфейс IComparable<T> (метод CompareTo(T)). Таким образом, сортировка выполняется методом Sort(), без написания дополнительного кода.
Если тип элементов списка – пользовательский, то это означает, что методы проверки равенства (Equals) и сравнения элементов (CompareTo) должны быть переопределены.
Форма для работы со списком объектов показана ниже:
На форме предусмотрены операции “Добавления”, “Удаления”, “Редактирования”, “Сортировки” и “Поиска” элементов списка.

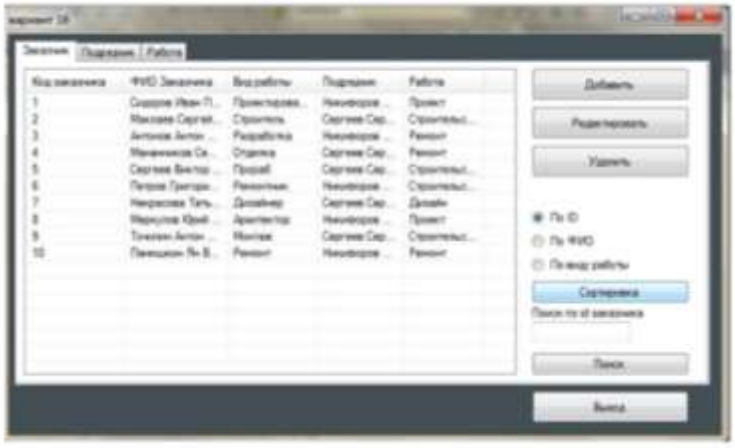


Форма для работы с содержимым самого объекта выглядит так:

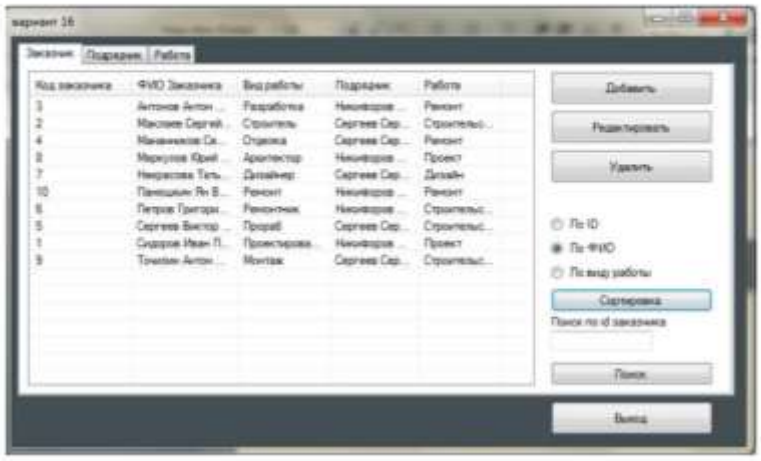


Сортировка может осуществляться 3 способами

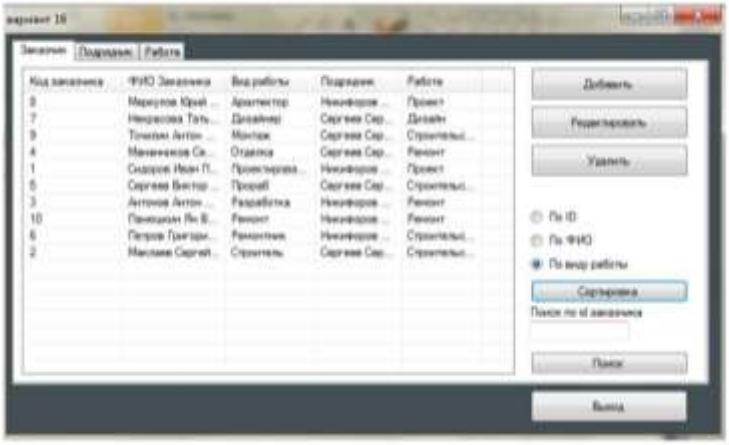
По ID заказчика



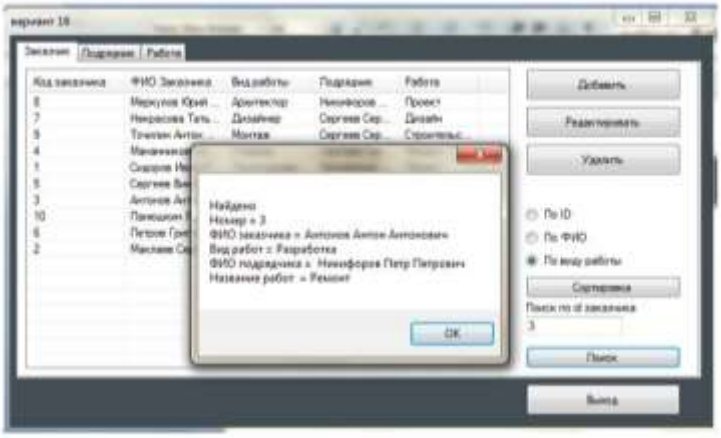
По ФИО заказчика (в алфавитном порядке)

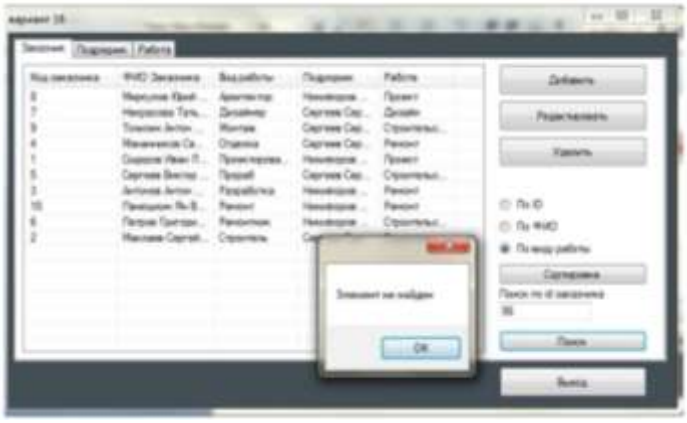


По виду работы (в алфавитном порядке)



Поиск происходит по ключевому полю:





Операция поиска выполняется только по ключевому атрибуту, с использованием возможностей класса Dictionary<T, T>.

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Создать внешний вид формы программы
- Написать программные коды для созданной формы
- Запустить и отладить программу

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Лабораторная работа№ 9 Работа с типом данных структура. Коллекции. Параметризованные классы.

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Задание:

1. Создать шаблон заданного класса. Определить конструкторы, деструктор, перегруженную операцию присваивания (“=”) и операции, заданные в варианте задания.
2. Написать программу тестирования, в которой проверяется использование шаблона для стандартных типов данных.
3. Выполнить тестирование.
4. Определить пользовательский класс, который будет использоваться в качестве параметра шаблона. Определить в классе необходимые функции и перегруженные операции.
5. Написать программу тестирования, в которой проверяется использование шаблона для пользовательского типа.
6. Выполнить тестирование.

Методические указания

1. Класс АТД реализовать как динамический массив. Для этого определение класса должно иметь следующие поля:

– указатель на начало массива;
– максимальный размер массива;
– текущий размер массива.

2. Для вода и вывода определить в классе функции inputи print.3

3. Чтобы у вас не возникало проблем, аккуратно работайте с константными объектами.
   Например:
   *конструктор копирования следует определить так: MyTmp (const MyTmp& ob);
   *операцию присваивания перегрузить так: MyTmp& operator= (const MyTmp& ob);4.
   Для шаблонов множеств, списков, стеков и очередей в качестве стандартных типов использовать символьные, целые и вещественные ти-пы. Для пользовательского типа взять класс из лабораторной работы No 1.5. Для шаблонов массивов в качестве стандартных типов использовать целые и вещественные типы. Для пользовательского типа взять класс “комплексное число”complex.classcomplex{intre; // действительная часть intim; // мнимая часть public;// необходимые функции и перегруженные операции};

6. Реализацию шаблона следует разместить вместе с определением в заголовочном файле.
7. Программа создается как EasyWin-приложение в BorlandC++5.02.

Постановка задачи.

- Следует дать конкретную постановку, т.е указать шаблон какого класса должен быть создан, какие должны быть в нем конструкторы, компоненты-функции, перегруженные операции и т.д. То же самое следует указать для пользовательского класса.
- Определение шаблона класса с комментариями.
- Определение пользовательского класса с комментариями.
- Реализация конструкторов, деструктора, операции присваивания и операций, которые заданы в варианте задания
- То же самое для пользовательского класса.
- Результаты тестирования. Следует указать для каких типов и какие операции проверены и какие выявлены ошибки (или не выявлены)

Варианты заданий

1.Класс –одномерный массив.

Дополнительно перегрузить следующие операции: * – умножение массивов; [] – доступ по индексу.

2. Класс –одномерный массив. Дополнительно перегрузить следующие операции:int() – размер массива;[] – доступ по индексу.

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Создать внешний вид формы программы
- Написать программные коды для созданной формы
- Запустить и отладить программу

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Лабораторная работа№ 10 Использование регулярных выражений. Операции со списками.

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Задание:

1. Определить класс-строку. В класс включить два конструктора: для определения класса строки строкой символов и путем копирования другой строки (объекта класса строки). Предусмотреть функции поиска слова в строке и добавления другой строки, начиная с позиции N.
2. Создать внешний вид формы программы
3. Написать программные коды для созданной формы
4. Запустить и отладить программу

2. Определить класс-строку. В класс включить два конструктора: для определения класса строки строкой символов и путем копирования другой строки (объекта класса строки). Предусмотреть функции слияния двух строк и функцию подсчёта предложений в строке.

Порядок выполнения работы

- Составить математическую модель задачи.

- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Создать внешний вид формы программы
- Написать программные коды для созданной формы
- Запустить и отладить программу

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Тема 1.1.4 Паттерны проектирования

Лабораторная работа№ 11,12 Использование основных шаблонов

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.

Материальное обеспечение:

– Мультимедийные средства хранения, передачи и представления информации.

Задание:

1. В каждом из вариантов указан шаблон для реализации и проект, использующий этот шаблон.

Необходимо сделать следующее:

– Нарисовать в UML диаграмму классов реализуемой программы. (проектирование)
– Реализовать программу на C++. (реализация)

Для каждого из шаблонов, предложенных в вариантах можно найти пример реализации UML и кода в приложенной книге “Паттерны проектирования”.

2. Шаблон “Стратегия”. Проект “Принтеры”. В проекте должны быть реализованы разные модели принтеров, которые выполняют разные виды печати.

3. Шаблон “Наблюдатель”. Проект “Оповещение постов ГАИ”. В проекте должна быть реализована отправка сообщений всем постам ГАИ.

Порядок выполнения работы

– Составить математическую модель задачи.
– Выбрать и обосновать наиболее рациональный метод решения задачи;
– Разработать алгоритм для решения задачи.
– Создать внешний вид формы программы
– Написать программные коды для созданной формы
– Запустить и отладить программу

Форма предоставления результата

– Блок – схема.
– Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Лабораторная работа № 13. Использование порождающих шаблонов.

Цель работы:

– Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
– Выполнять отладку программных модулей с использованием специализированных программных средств;
– Выполнять тестирование программных модулей;
– Осуществлять рефакторинг и оптимизацию программного кода.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Задание:

1. В каждом из вариантов указан шаблон для реализации и проект, использующий этот шаблон.

Необходимо сделать следующее:

- Нарисовать в UML диаграмму классов реализуемой программы. (проектирование)
- Реализовать программу на C++. (реализация)

Для каждого из шаблонов, предложенных в вариантах можно найти пример реализации UML и кода в приложенной книге “Паттерны проектирования”.

2. Шаблон “Декоратор”. Проект “Универсальная электронная карта”. В проекте должна быть реализована универсальная электронная карта, в которой есть функции паспорта, страхового полиса, банковской карты и т. д.
3. Шаблон “Фабричный метод”. Проект “Фабрика смартфонов”. В проекте должно быть реализовано создание смартфонов с различными характеристиками.

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Создать внешний вид формы программы
- Написать программные коды для созданной формы
- Запустить и отладить программу

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Лабораторная работа № 14. Использование структурных шаблонов.

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Задание:

1. В каждом из вариантов указан шаблон для реализации и проект, использующий этот шаблон.

Необходимо сделать следующее:

- Нарисовать в UML диаграмму классов реализуемой программы. (проектирование)
- Реализовать программу на C++. (реализация)

Для каждого из шаблонов, предложенных в вариантах можно найти пример реализации UML и кода в приложенной книге “Паттерны проектирования”.

1. Постановка задачи

Шаблон “Стратегия”. Проект “Принтеры”. В проекте должны быть реализованы разные модели принтеров, которые выполняют разные виды печати.

2. Шаблон “Наблюдатель”. Проект “Оповещение постов ГАИ”. В проекте должна быть реализована отправка сообщений всем постам ГАИ.

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Создать внешний вид формы программы
- Написать программные коды для созданной формы
- Запустить и отладить программу

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Лабораторная работа № 15. Использование поведенческих шаблонов.

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Задание:

В каждом из вариантов указан шаблон для реализации и проект, использующий этот шаблон.

Необходимо сделать следующее:

- Нарисовать в UML диаграмму классов реализуемой программы. (проектирование)
- Реализовать программу на C++. (реализация)

Для каждого из шаблонов, предложенных в вариантах можно найти пример реализации UML и кода в приложенной книге “Паттерны проектирования”.

1. Шаблон “Фабричный метод”. Проект “Фабрика смартфонов”. В проекте должно быть реализовано создание смартфонов с различными характеристиками. Пример использования шаблона в главе 4.
2. Шаблон “Абстрактная фабрика”. Проект “Заводы по производству автомобилей”. В проекте должно быть реализована возможность создавать автомобили различных типов на разных заводах.

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Создать внешний вид формы программы
- Написать программные коды для созданной формы
- Запустить и отладить программу
-

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

–

Тема 1.1.5. Событийно-управляемое программирование
Лабораторная работа № 16,17. Разработка приложения с использованием текстовых компонентов

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;

– Выполнять отладку программных модулей с использованием специализированных программных средств;
– Выполнять тестирование программных модулей;
– Осуществлять рефакторинг и оптимизацию программного кода.

Выполнив работу, Вы будете:

уметь:

– Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
– Оценка сложности алгоритма.
– Создавать программу по разработанному алгоритму как отдельный модуль.
– Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
– Выполнять отладку и тестирование программы на уровне модуля.
– Оформлять документацию на программные средства.
– Применять инструментальные средства отладки программного обеспечения.
– Выполнять оптимизацию и рефакторинг программного кода.
– Работать с системой контроля версий.

Материальное обеспечение:

– Мультимедийные средства хранения, передачи и представления информации.

Задание:

Для решения задачи использовать простые текстовые компоненты и кнопки:

– TEdit - для ввода скалярных исходных данных,
– TLabel – для задания подписей элементов формы,
– TButton – для активизации действий («Вычислить», «Выполнить новый расчет»),
– TMemo – для вывода условия задачи, исходных массивов и вывода результата.

Все компоненты расположить на панелях (TPanel). При разработке программного интерфейса выровнять панели относительно границ формы, а компоненты относительно границ панели с использованием свойств выравнивания и фиксации компонент относительно контейнера (Align, Anchor). Компоненты, в которых выводятся массивы и другие результаты, должны быть недоступными для изменения.

Примеры внешнего вида форм приведены ниже на рисунках.

Рис. 2. Пример расположения компонентов на форме для задания 1

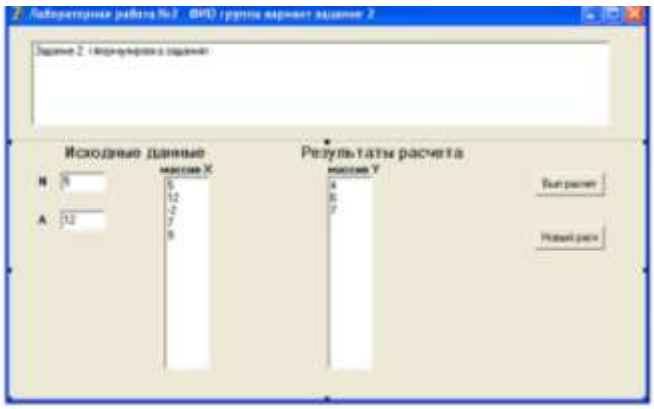


Рис. 3. Пример расположения компонентов на форме для задания 2

Программирование событий

Управление работой оконного приложения выполняется по событиям (открытие формы, щелчок мышью по кнопке и т.д.).

Действия (выполняемые операторами программы) при открытии формы необходимо добавить в обработчик события открытия формы OnShow.

Действия (выполняемые операторами программы) при нажатии кнопки (щелчке мышью по кнопке) необходимо добавить в обработчик события «щелчок» по кнопке OnClick. Для этого в инспекторе объектов необходимо в выпадающем списке выбрать объект-кнопку (допустим кнопке «Выполнить расчет» соответствует объект Button1). Перейти на закладку Events, найти свойство OnClick и справа в пустом поле сделать двойной щелчок мышью (можно на форме сделать двойной щелчок по кнопке). В программном модуле будет добавлен шаблон процедуры обработчика:

Разработать оконное приложение для решения задач индивидуального задания. Исходные массивы сгенерировать с использованием датчика случайных чисел и вывести в компоненте формы. В программе использовать динамические массивы.

1. Даны координаты точки (X,Y). Определить, попадает ли точка внутрь кольца с центром в точке (C,D) и радиусами R1, R2.

2. Дан одномерный массив Zm. Сформировать массив Yn, состоящий из элементов массива Zm, значение которых меньше среднего арифметического нечетных элементов исходного массива.

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Создать внешний вид формы программы
- Написать программные коды для созданной формы
- Запустить и отладить программу

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Лабораторная работа № 18,19. Разработка приложения с несколькими формами.

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Задание:

1.Разработайте два класса потомка от Animal, которые будут отображать особенности двух пород собак. Разработайте методы для этих классов, позволяющие получить некоторые характеристики породы (рост, длина шерсти, длина ушей и т.д.). Дополните форму компонентами, позволяющими увидеть все характеристики разработанных классов.

Примерная форма проекта представлена на рисунке 4.4.

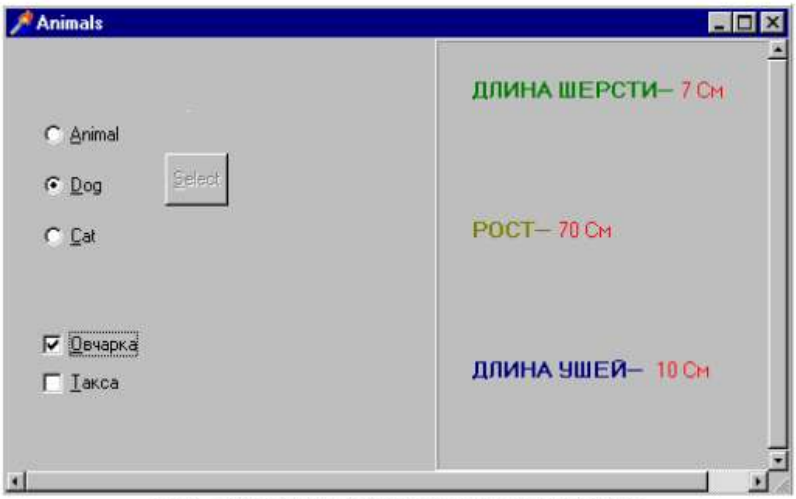


Рис. 4.4. Форма характеристик пород собак

После запуска программы представляется возможность выбора между значениями: Animal, Dog, Cat.
Выбор осуществляется при помощи кнопки SELECT. Далее предоставляется дополнительная возможность для выбора породы собаки. При выделении знаком: ☑
Интересующей породы можно получить дополнительную информацию.

2. Создать внешний вид формы программы
3. Написать программные коды для созданной формы
4. Запустить и отладить программу

3. Разработайте двакласса потомка от Animal, которые будут отображать особенности двух новых животных Wolf (волк) и Jascal (шакал). Разработайте методы для этихк лассов, позволяющие получить некоторые характеристики этих видов животных (рост по холке, длина клыков, вес и т.д.).

Дополните форму компонентами, позволяющими увидеть все характеристики разработанных классов.

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Создать внешний вид формы программы
- Написать программные коды для созданной формы
- Запустить и отладить программу

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой

обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Лабораторная работа№ 20,21. Разработка приложения с не визуальными компонентами.
Лабораторная работа № 22,23. Разработка игрового приложения.
Лабораторная работа № 24. Разработка приложения с анимацией.

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Задание:

1. Игра «Шашки». Игра для двух игроков. Шашки передвигаются с помощью мыши. Шашки удаляются с поля двойным щелчком мыши.

2. Игра «Сапер». На поле размером 8х8 случайно размещается N мин. Задача игрока найти все мины не подорвавшись.

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Создать внешний вид формы программы
- Написать программные коды для созданной формы

– Запустить и отладить программу

Форма предоставления результата

– Блок – схема.
– Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Тема 1.1.6 Оптимизация и рефакторинг кода

Лабораторная работа№ 25,26,27,28,29,30,31,32. Оптимизация и рефакторинг кода.

Цель работы:

– Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
– Выполнять отладку программных модулей с использованием специализированных программных средств;
– Выполнять тестирование программных модулей;
– Осуществлять рефакторинг и оптимизацию программного кода.

Выполнив работу, Вы будете:

уметь:

– Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
– Оценка сложности алгоритма.
– Создавать программу по разработанному алгоритму как отдельный модуль.
– Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
– Выполнять отладку и тестирование программы на уровне модуля.
– Оформлять документацию на программные средства.
– Применять инструментальные средства отладки программного обеспечения.
– Выполнять оптимизацию и рефакторинг программного кода.
– Работать с системой контроля версий.

Материальное обеспечение:

– Мультимедийные средства хранения, передачи и представления информации.

Задание:

Выполните следующие задания:

1. Создайте C# решение в среде Visual Studio. Назовите его в соответствии с вашим вариантом задания. В качестве исходного типа проекта выберите проект динамической

библиотеки (.dll). Назовите проект также согласно вашему варианту или просто Model. Данный проект будет содержать в себе бизнес-логику вашей будущей программы, т.е. содержать ключевые сущности, структуры данных и способы их взаимодействия.
2. Создайте сущность-интерфейс согласно вашему варианту. Интерфейс в языке C# описывается с помощью ключевого слова interface. Опишите ключевые свойства и методы интерфейса. Не забудьте о правильном именовании элементов в соответствии с RSDN. Подумайте, какие свойства и методы в дальнейшем будут являться общими (т.е. будут в интерфейсе), а какие должны быть реализованы в конкретных классах.
3. Создайте два класса, реализующих данный интерфейс. Классы ОБЯЗАТЕЛЬНО должны иметь различные реализации методов интерфейса. При этом, дочерние классы не должны иметь никаких ссылок друг на друга, также, как и интерфейс не должен ничего знать о классах, его реализующих.
4. Реализуйте валидацию свойств с помощью механизмов обработки исключений – если на вход свойству приходит некорректное значение, выходящее за допустимые пределы, свойство должно выбросить исключение соответствующего типа с описанием возникшей ошибки. Например, если в свойство «Возраст» пытаются присвоить отрицательное значение, необходимо выбросить экземпляр IncorrectArgumentException. Внимательно продумайте все возможные некорректные варианты данных, в том числе и ссылки на null. В случае, если механизмы валидации одинаковы у всех свойств, измените архитектуру бизнес-логики: вместо реализации интерфейса двумя классами, сделайте наследование двух классов от абстрактного класса с приобретением механизмов валидации.
5. Добавьте в решение еще один проект. Это можно сделать с помощью контекстного меню панели «Обозреватель решения». Для этого кликните правой кнопкой по названию решения в панели и выберите пункт «Добавить проект». Создайте исполняемый проект консоли (Win32 Console Application) и назовите его «ConsoleLoader». Данный проект является временным и необходим для начального тестирования бизнес-логики.
Примечание: в последующих лабораторных вы избавитесь от этого проекта и создадите вместо него проект графического интерфейса (WinForms Application). Однако если вы уже можете продемонстрировать работу бизнес-логики на оконном пользовательском интерфейсе – можете сразу создать необходимый проект.
6. Продемонстрируйте корректную работу бизнес-логики простыми заданиями в консоли: создайте переменную-ссылку на интерфейс, поочередно присвойте в неё экземпляры реализующих классов и вызовите реализации методов – покажите, что это разные реализации. Вероятно, для демонстрации ваших классов в проекте ConsoleLoader придется добавить методы ввода/вывода классов бизнес-логики через консоль.

Задача

1. Геометрические фигуры с собственными реализациями расчета площади фигуры: круг, прямоугольник, разносторонний треугольник.

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Создать внешний вид формы программы
- Написать программные коды для созданной формы
- Запустить и отладить программу

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Тема 1.1.7 Разработка пользовательского интерфейса.
Лабораторная работа 33,34,35,36,37,38,39,40 Разработка интерфейса пользователя

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Краткие теоретические сведения:

- В ходе данной работы мы рассмотрим создание средствами СУБД Access пользовательского интерфейса. Данные можно вводить и прямо в таблицы, но для конечных пользователей удобнее осуществлять ввод в отдельных окнах, предоставляющих дополнительные элементы управления для навигации и работы с записями. В Access такие окна называются формами, они служат для организации пользователю удобного графического интерфейса для работы с данными. При работе с формами есть несколько режимов: • режим Формы;
- режим Конструктора;
- режим Макета.

В режиме Формы пользователь работает с данными; производятся добавление новых записей, просмотр, удаление или редактирование записей таблицы или запроса, являющегося источником данных для формы. В режиме Конструктора осуществляется модификация структуры формы
(изменение внешнего вида, добавление и удаление элементов управления и т.д.). Режим Макета позволяет увидеть данные и в то же время редактировать структуру формы.
Макет формы имеет следующие разделы:

- заголовок формы;
- область данных;
- примечание формы.

Область данных определяет основную часть формы. Этот раздел может содержать элементы управления, отображающие данные из источника данных (таблицы или запроса), а также неизменяемые данные (например, надписи и линии). При печати

формы область данных будет повторяться для каждой записи таблицы, тогда как заголовок и примечание – только один раз.

Размещаемые на форме элементы управления могут быть разных типов.

Связанные элементы управления связаны с полями базовой таблицы, которая является источником данных для формы. Если источником данных является запрос, то присоединенные элементы управления могут связываться с полями в разных таблицах. При изменении значения в элементе управления обновляется и значение поля соответствующей записи в таблице.

Свободные элементы управления не связаны с таблицами, в частности они служат для оформления (линии, надписи и т.д.).

Вычисляемые элементы управления – это элементы, значения которых рассчитываются на основе значений других элементов. В качестве источника данных для этих элементов используются выражения и функции.

Ниже перечислены некоторые элементы управления, которые могут быть размещены на форме (полный перечень можно увидеть, работая в режиме Конструктора).

Надпись (Label) содержит небольшой текст, как правило, пользователем не изменяемый. *Поле* (Text Box) служит для отображения, ввода и изменения данных. *Флажок* (Check box), *Переключатель* (Option button), *Выключатель* (Toggle button) – элементы, определяющие значения логического типа. В Access, в зависимости от настроек поля, это может быть "Истина/ Ложь", "Да/Нсг" или "Вкл./Выкл.".

Элементы *Список* (List Box) и *Поле со списком* (Combo Box) позволяют выбрать значения из определенного списка (это может быть просто заданный набор возможных значений или, например, данные одного из столбцов таблицы). Поле со списком также позволяет ввести в поле новое значение.

Кроме "простых" форм, Access позволяет создавать "сложные" формы, включающие данные, собранные из нескольких связанных таблиц. При этом *подчиненной формой* называется форма, которая встраивается в другую. Форма, содержащая подчиненную форму, называется *главной.*

В папке с файлами к лабораторной работе находятся две базы Access – уже знакомая но прошлому занятию база lib.accdb, в которую добавлены две формы, и база lib_l .accdb, отличающаяся от первой только отсутствием форм.

Начнем с создания формы с информацией об изданиях. Создадим ее с помощью мастера и потом изменим в редакторе. Перейдем на закладку Создание → Мастер форм. В первом окне "мастера" надо выбрать таблицу (или запрос), данные из которой будут отображаться в форме (таблица

Book), и поля из этой таблицы. Надо отметить, что идентификатор книги (типа Счетчик) показывать пользователю на форме не надо (рис. П.2.1): изменяется идентификатор автоматически и задать его значение вручную пользователь не сможет, что может привести к путанице.

Рис. П.2.1. Выбор полей таблицы

В представленном примере поля располагаются в один столбец, форма названа "Информация об изданиях". В последнем окне мастера выберите опцию "Изменить макет формы", после чего откроется конструктор, в котором нужно изменить размер

полей, текст надписей, добавить элементы оформления. В примере в примечание формы добавлена надпись "F1 – справка", а заголовок – дата и время.

Задание:

В базе данных lib_1.acc.db в соответствии с приведенным выше описанием и примером создайте новую форму. Не забудьте в заголовке формы поставить дату и время, а в примечании – надпись про справку. Введите через форму какие-нибудь данные. Посмотрите, как будет выглядеть форма при печати (Файл → Печать → Предварительный просмотр).

Самостоятельно создайте форму для редактирования информации о статусах книг.

Теперь рассмотрим создание составных форм. Такие формы очень 5'добны при работе со связанными таблицами – в одной форме можно представить данные из нескольких таблиц (рис. П.2.2).

Проще всего создать подобную форму с помощью мастера. В первом окне мастера укажите, что на форме надо представить информацию об авторе, названии, издательстве, годе издания, взятую из таблицы *Book,* и информацию об идентификаторе и статусе книги из таблицы *Book_in_lib.* Мастер предложит создать подчиненные формы. Дальнейшая работа мастера не отличается от предыдущего случая. Обратите внимание, что мастер создал две формы – основную и подчиненную.

После окончания работы мастера можно отредактировать форму в режиме конструктора, например поменять подписи с подставляемых по умолчанию названий столбцов на какие-то более подробные пояснения.

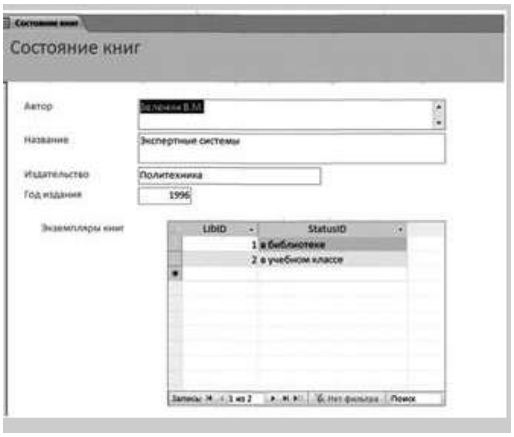


Рис. П.2.2. Составная форма

Задание.

Создайте составную форму для отображения информации об издании и текущем состоянии экземпляров книг в библиотеке.

Нередко для удобства работы пользователя нужно разместить на форме какие-нибудь дополнительные элементы, например кнопки. Внизу каждой формы расположены кнопки, позволяющие передвигаться по записям (следующая, предыдущая, первая, последняя, новая запись). Однако для того чтобы удалить запись, приходится использовать соответствующую кнопку с основной напели инструментов, что не очень удобно.

Добавим кнопку удаления на форму "Информация об изданиях". Для этого в режиме редактирования надо воспользоваться панелью элементов. На панели элементов надо выбрать элемент "кнопка" и мышкой нарисовать его на форме. После этого появится диалоговое окно Создание кнопок, в котором надо указать, что создаваемая кнопка относится к категории "Обработка записей" и требуемое действие – "Удалить запись". Далее для созданной кнопки можно выбрать подходящую пиктограмму или надпись, а также ее имя.

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;

- Разработать алгоритм для решения задачи.
- Создать внешний вид формы программы
- Написать программные коды для созданной формы
- Запустить и отладить программу
- .

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Тема 1.1.8 Программирование в среде 1С Предприятие
Лабораторная работа №41,42,43,44,45 Создание приложения с БД

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Задание:

1. Сформировать новую конфигурацию.
2. Создать в конфигурации подсистемы: «Лабораторная работа№», «Администрирование». Расположить их на панели разделов в указанной последовательности.
3. Создать две роли: Администратор, наделенный всеми правами, в т. числе и административными, и пользователь без административных прав.
4. Создать двух пользователей, один из которых должен играть роль администратора, а второй – пользователя.
5. Создать константу, указанную в задании, включив ее в подсистему «Администрирование». Дополнительно создать форму констант, включив ее в подсистему «Администрирование». В пользовательском режиме задать значение данной константы. В дальнейшем при запуске приложения выводить значение данной константы в окно сообщений. Если для разработки типа константы необходимо разработать дополнительные объекты, включить их в подсистему «Лабораторная работа№».
6. Разработать справочники, указанные в задании. Включить их в подсистему «Лабораторная работа№». Основной (первый в каждом задании) справочник должен быть многоуровневым. Количество уровней - произвольно. Название групп в справочнике - произвольно. Решение задачи состоит из следующих этапов:
   - Описать структуры справочников средствами конфигуратора;
   - Разработать формы диалога ввода данных в справочники (как для ввода групп, так и для ввода элементов). Группы справочников должны состоять только из кодов и наименований;
   - В режиме 1С:Предприятие ввести несколько групп, содержащий не менее 2-3 элементов.
   - Дополнительно в случае необходимости разработать объекты метаданных, необходимые для работы заданного справочника в Конфигурации.
   - Обеспечить проверку заполнения всех реквизитов шапки основного справочника. Разрешать запись элемента справочника только в случае заполненности всех реквизитов шапки.
   - Поместить команду создания нового элемента основного справочника в задании на рабочий стол приложения.
   - В форму списка основного справочника вставить кнопку с названием «Справка». При нажатии на эту кнопку программа должна вывести в окно сообщений справку. Содержание справки – см. варианты заданий. Использовать команду Сообщить(…)
   - Для разработанного основного справочника сформировать подчиненный справочник. Структура справочника – см. варианты.
   - Проверку заполненности трех обязательных (произвольных) реквизитов основного справочника перед записью элемента справочника выполнить программным путем, поместив процедуру проверки в модуль менеджера.
7. Создать обработку, которая должна выводить список всех справочников, разработанных в конфигурации. Для вывода информации использовать команду Сообщить(…)

Задача

1. Константа «Главный бухгалтер», тип СправочникСсылка.Сотрудники

2 Справочник сотрудников:

- Табельный номер сотрудника;
- ФИО сотрудника;
- Тип сотрудника (выбирается из списка – перечисления со значениями: штатный, внутренний совместитель, внешний совместитель);

- Дата поступления на работу;
- Оклад;
- Признак членства в профсоюзе.

Табличная часть элементов справочника содержит список детей сотрудника и имеет следующую структуру:

- Пол (выбирается из списка, заданного перечислением);
- ФИО ребенка.
- Дата рождения

3 Справочник стандартных вычетов сотрудника (подчиненный):

- Тип вычета (выбирается из списка, задаваемого перечислением со значениями: Собственный, на ребенка, Герой России);
- Сумма вычета.
- Дата начала действия
- Дата окончания действия

Замечание. При вводе нового элемента справочника обеспечить автоматическое заполнение реквизита "Тип сотрудника" предопределенным значением - "ИТР" (по умолчанию).

Замечание 2. Сформировать предопределенную группу в справочнике «Уволенные»

Получить справку: вывести наименования всех сотрудников – не членов профсоюза

Задача

1. Константа «Главный контрагент», тип СправочникСсылка.Контрагенты

2. Справочник контрагентов:

- Код контрагента;
- Наименование контрагента;
- Адрес контрагента;
- Основной расчетный счет(элемент справочника расчетных счетов контрагента);
- Банк контрагента (элемент справочника банков)
- Телефон ;
- Размер кредита.

Табличная часть элементов справочника содержит список договоров контрагента и имеет следующую структуру:

- Код договора;
- Наименование договора.
- Дата заключения договора
- условия договора (текст)

3. Справочник расчетных счетов контрагента(подчиненный)

- код
- наименование
- банк (элемент справочника банков)

Замечание. При вводе нового элемента справочника обеспечить автоматическое заполнение реквизита "Размер кредита" предопределенным значением - "5 000 руб" (по умолчанию).

Замечание 2. Сформировать предопределенную группу в справочнике «Зарубежные контрагенты»

Получить справку: вывести наименования всех контрагентов с нулевым значением кредита.

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.

– Написать и отладить программу.

Форма предоставления результата

– Блок – схема.
– Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Лабораторная работа №46,47,48,49,50 Создание запросов к БД

Цель работы:

– Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
– Выполнять отладку программных модулей с использованием специализированных программных средств;
– Выполнять тестирование программных модулей;
– Осуществлять рефакторинг и оптимизацию программного кода.

Выполнив работу, Вы будете:

уметь:

– Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
– Оценка сложности алгоритма.
– Создавать программу по разработанному алгоритму как отдельный модуль.
– Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
– Выполнять отладку и тестирование программы на уровне модуля.
– Оформлять документацию на программные средства.
– Применять инструментальные средства отладки программного обеспечения.
– Выполнять оптимизацию и рефакторинг программного кода.
– Работать с системой контроля версий.

Материальное обеспечение:

– Мультимедийные средства хранения, передачи и представления информации.

Задание:

Разработать отчет с нестандартной расшифровкой вывода списка элементов основного справочника (см. варианты заданий в главе 4). Для этого необходимо:

☐ Создать новую подсистему «Лабораторная работа3». Все новые объекты данного

задания поместить в данную подсистему.

☐ Создать в конфигурации новый объект-отчет.

☐ Создать форму отчета. Для формирования отчета использовать конструктор запроса с обработкой результата.

☐ Внести корректировку в результаты работы конструктора, обеспечивающие вывод отчета и его нестандартную расшифровку

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Написать и отладить программу.

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Лабораторная работа № 51,52,53,54 Создание хранимых процедур.

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.

- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Задание:

Выполнение работы состоит из следующих этапов:

- Создать новую подсистему «Лабораторная работа№». Все новые объекты, созданные в данной работе, поместить в данную подсистему.
- Разработать, если это необходимо, дополнительные документы
- Разработать тип и структуру регистров накопления для хранения движений, формируемых при проведении документов.
- Разработать модули проведения документов.
- Разработать отчет. Средства обращения к источнику данных – запрос. Использовать конструктор запроса с обработкой результатов. Отчет должен выводить общие итоги.
- Отчет должен иметь расшифровку. Тип расшифровки – см. варианты заданий.
- В случае разработки нестандартного отчета вид отчета разработать самостоятельно.

Задача

Ведомость поступления товара _____ на склады от поставщиков
за ___ период с _____ по ___
Склад Документ Дата Поставщик Колич. Единицы Цена Стоимость

Замечание 1. Строки отчета группируются по поставщикам. В конце группы следует включать строку «Итого по поставщику».
Замечание 2. Разработать документ «Поступление товаров от поставщиков» Расшифровка – стандартная.

Задача

Ведомость поступления товара _____ на склады от поставщика_____
за ___ период с _____ по ___
Склад Документ Дата Кол. Цена Стоимость Сумма НДС Всего с НДС

Замечание 1. Строки отчета группируются по складам. В конце группы следует включать строку «Итого по складу».
Замечание 2. Разработать документ «Поступление товаров от поставщиков» Расшифровка – стандартная.

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Написать и отладить программу.

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

МДК.01.02 Поддержка и тестирование программных модулей

Тема 1.2.1 Отладка и тестирование программного обеспечения
Лабораторная работа № 1,2 Тестирование «белым ящиком».

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Краткие теоретические сведения:

Изучить методы тестирования логики программы, формализованные описания результатов тестирования и стандарты по составлению схем программ.

Стратегия «белого ящика»

Существуют следующие методы тестирования по принципу «белого ящика»:

- покрытие операторов;
- покрытие решений;
- покрытие условий;
- покрытие решений/условий;
- комбинаторное покрытие условий.

Метод покрытия операторов

Целью этого метода тестирования является выполнение каждого оператора программы хотя бы один раз.

Задание:

Если для тестирования задать значения переменных $A = 2$, $B = 0$, $X = 3$, будет реализован путь *ace,* т. е. каждый оператор программы выполнится один раз (рис. Л5.1, *а).* Но если внести в алгоритм ошибки — заменить в первом условии *and* на *or,* а во втором $X > 1$ на $X < 1$ (рис. Л5.1, *б),* ни одна ошибка не будет обнаружена (табл. Л5.1). Кроме того, путь *abd* вообще не будет охвачен тестом, и если в нем есть ошибка, она также не будет обнаружена. В табл. Л5.1 ожидаемый результат определяется по блок-схеме на рис. Л5.1, *а,* а фактический — по рис. Л5.1, *б.*

Как видно из этой таблицы, ни одна из внесенных в алгоритм ошибок не будет обнаружена.

Таблица Л5.1. Результат тестирования методом покрытия операторов

Тест	Ожидаемый результат	Фактический результат	Результат тестирования
A = 2, *B*= 0, *X*=Ь	*X*= 2,5	*X*= 2,5	Неуспешно

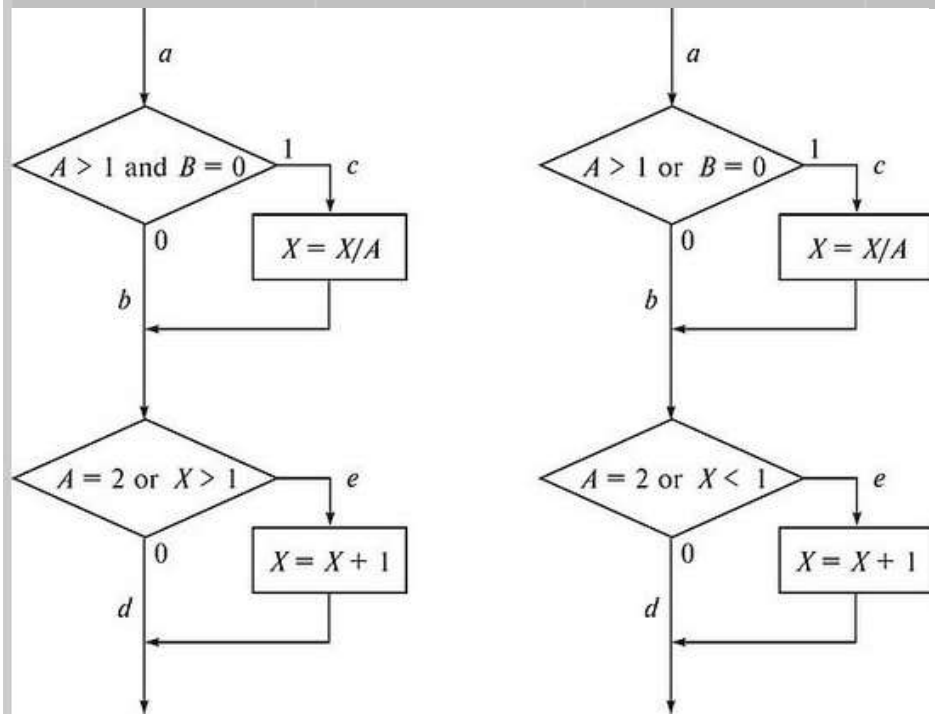


Рис. Л5.1. Пример алгоритма программы: *а* — правильный; *б* — с ошибкой

Метод покрытия решений (покрытия переходов)

Согласно методу покрытия решений каждое направление перехода должно быть реализовано, по крайней мере, один раз. Этот метод включает в себя критерий покрытия операторов, так как при выполнении всех направлений переходов выполнятся все операторы, находящиеся на этих направлениях.

Для программы, приведенной на рис. Л5.1, покрытие решений может быть выполнено двумя тестами, покрывающими пути *{ace, aЬc*!}, либо *{acc1, aЬe*}. Для этого выберем следующие исходные данные; *{A* = 3, *B*= О, *X=3}* — в первом случае и *{A = 2, B*= 1, *X*= 1} — во втором. Однако путь, где *X* не меняется, будет проверен с вероятностью 50 %: если во втором условии вместо условия *X*> 1 записано *X*<, то ошибка не будет обнаружена двумя тестами.

Результаты тестирования приведены в табл. Л5.2.

Таблица Л5.2. Результат тестирования методом покрытия решений

Тест	Ожидаемый результат	Фактический результат	Результат тестирования
/1 = 3, B= 0, X=2	X = 1	X = 1	Неуспешно
A = 2, B = 1, X= 1	X=2	* II	Успешно

Метод покрытия условий

Этот метод может дать лучшие результаты по сравнению с предыдущими. В соответствии с методом покрытия условий записывается число тестов, достаточное для того, чтобы все возможные результаты каждого условия в решении выполнялись, по крайней мере, один раз.

В рассматриваемом примере имеем четыре условия: *{A*> 1, 5=0}, *{A* = 2, *X*> 1}. Следовательно, требуется достаточное число тестов, такое, чтобы реализовать ситуации, где *A* > 1, *A* < 1, 5 = 0 и й#0 в точке а и /1 = 2, *A* *2, *X*> и Т< 1 в точке *Ь.* Тесты, удовлетворяющие критерию покрытия условий (табл. Л5.3), и соответствующие им пути:

а) *A = 2,* 5 = 0, *X=4 ace;*

б) *A* = 1, 5= 1, *X=0 aЬc!.*

Таблица Л5.3. Результаты тестирования методом покрытия условий

Тест	Ожидаемый результат	Фактический результат	Результат тестирования
/1 = 2, B= 0, X= 4	*=з	X= 3	Неуспешно
A= 1, B= 1, X=0	*=0	X= 1	Успешно

Метод покрытия решений/условий

Критерий покрытия решений/условий требует такого достаточного набора тестов, чтобы все возможные результаты каждого условия выполнялись по крайней мере один раз, все результаты каждого решения выполнялись по крайней мере один раз и, кроме того, каждой точке входа передавалось управление по крайней мере один раз.

Недостатки метода:

- • не всегда можно проверить все условия;
- • невозможно проверить условия, которые скрыты другими условиями;
- • недостаточная чувствительность к ошибкам в логических выражениях.

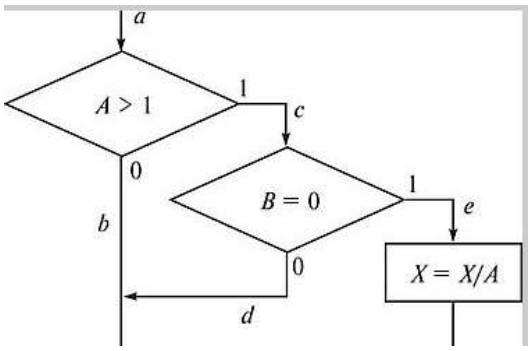
Так, в рассматриваемом примере два теста метода покрытия условий

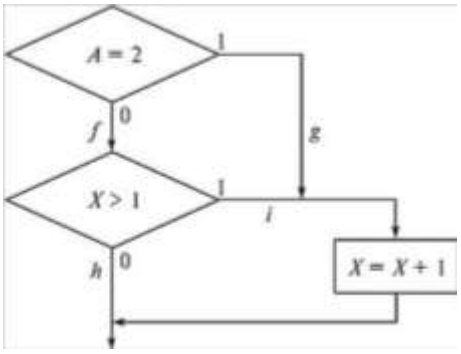
а) *A* = 2, *B* = О, *X= 4 ace;*

б*) A*= 1, *B*= 1, *X=0 aЬc!*

отвечают и критерию покрытия решений/условий. Это является следствием того, что одни условия приведенных решении скрывают другие условия в этих решениях. Так, если условие *A* > 1 будет ложным, транслятор может не проверять условия *B*= О, поскольку при любом результате условия *B*= 0 результат решения *((A >)&(B=0))* примет значение *ложь.* То есть в варианте на рис. Л5.1 не все результаты всех условий выполнятся в процессе тестирования.

Рассмотрим реализацию того же примера на рис. Л5.2. Наиболее полное покрытие тестами в этом случае осуществляется так, чтобы выполнялись все возможные результаты каждого простого решения.





Для этого нужно покрыть пути *aceg* (тест *A = 2, B*= О, *X*= 4), *асё/к* (тест *A* = 3, *B* = 1, *X*= 0), *аЬ/к* (тест *A* = 0, *B*= 0, *X*= 0), *аЬр* (тест Л = 0, *B* = 0, *X*= *2).*

Протестировав алгоритм на рис. Л5.2, нетрудно убедиться в том, что критерии покрытия условий и критерии покрытия решений/условий недостаточно чувствительны к ошибкам в логических выражениях.

Метод комбинаторного покрытия условий

Критерий комбинаторного покрытия условий удовлетворяет также и критериям покрытия решений, покрытия условий и покрытия решений/условий.

Этот метод требует создания такого числа тестов, чтобы все возможные комбинации результатов условия в каждом решении выполнялись по крайней мере один раз. По этому критерию в рассматриваемом примере должны быть покрыты тестами следующие восемь комбинаций:

1. *A*> 1, *B*= 0. *5. A = 2, X*> 1.
2. *A* > 1, *Bф* 0.
3. *A*< 1, *B*= 0.
4. *A*< 1, Вф 0.
6. *A* = 2, 1.
7. Аф2, *X*> 1.
8. Аф2, *X*< 1.

Для того чтобы протестировать эти комбинации, необязательно использовать все 8 тестов. Фактически они могут быть покрыты четырьмя тестами (табл. Л5.4):

- *A* = 2, *B = 0, X=4* (покрывает 1,5);
- *A = 2, B = 1, X= 1* (покрывает 2,6);
- *A* = 0,5, *B*= 0, *X= 2* (покрывает 3, 7);
- *A = I, B = 0, X*= (покрывает 4, 8_.

Таблица Л5.4. Результаты тестирования методом комбинаторного покрытия условий

Тест	Ожидаемый результат	Фактический результат	Результат тестирования
A-2, B=0, X- 4	*=з	*=з	Неуспешно
A = 2, B= 1, *X=* 1	*X= 2*	1 *Г)* СЧ II	Успешно
A = 0,5, *B=* 0, *X=2*	*X=3*	*X= 4*	Успешно
A= 1, ?=0, *= 1	*X=* 1	*X=* 1	Неуспешно

Порядок выполнения работы

1. Спроектировать тесты по принципу «белого ящика» для программы, разработанной в лабораторной работе № 4. Использовать схемы алгоритмов, разработанные и уточненные в лабораторных работах № 2, 3.
2. Выбрать несколько алгоритмов для тестирования и обозначить буквами или цифрами ветви этих алгоритмов.
3. Выписать пути алгоритма, которые должны быть проверены тестами для выбранного метода тестирования.
4. Записать тесты, которые позволят пройти по путям алгоритма.
5. Протестировать разработанную вами программу. Результаты оформить в виде таблиц (см. табл. Л5.1— Л5.4).
6. Проверить все виды тестов и сделать выводы об их эффективности.

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Лабораторная работа № 3,4 Тестирование «черным ящиком»

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Краткие теоретические сведения:

Тестирование «черного ящика» — это метод тестирования функционального поведения объекта (программной системы) с точки зрения внешнего мира.

Основная задача тестировщика для данного метода тестирования состоит в последовательной проверке соответствия поведения системы требованиям. Кроме того, тестировщик должен проверить работу системы в критических ситуациях — в случае подачи неверных входных значений. В идеальной ситуации все варианты критических ситуаций должны быть описаны в требованиях на систему, и тестировщику остается только придумывать конкретные проверки этих требований. Однако на самом деле в результате тестирования обычно выявляются следующие проблемы системы:

- поведение системы не соответствует требованиям;
- в ситуациях, не предусмотренных требованиями, система ведет себя неадекватно (при этом под неадекватным поведением может пониматься как полный крах системы, так и вообще любое поведение, не описанное в требованиях).

В первом случае обычно требуются изменения программного кода, гораздо реже — изменения требований. Изменение требований может потребоваться из-за их противоречивости (несколько разных требований описывают разные модели поведения системы в одной и той же ситуации) или некорректности (требования не соответствуют действительности).

Во втором случае однозначно требуются изменения требований ввиду их неполноты — в требованиях явно не учтена ситуация, приводящая к неадекватному поведению системы.

Методы «черного ящика» обеспечивают:

- эквивалентное разбиение;

• анализ граничных значений;
• анализ причинно-следственных связей;
• предположение об ошибке.

Эквивалентное разбиение состоит в разбиении входной области данных программы на конечное число классов эквивалентности так, чтобы каждый тест, являющийся представителем некоторого класса, был эквивалентен любому другому тесту этого класса. Классы эквивалентности выделяются путем перебора входных условий и разбиения их на две (или более) группы. При этом различают два типа классов эквивалентности: правильные, задающие входные данные для программы, и неправильные, основанные на задании ошибочных входных значений. Разработка тестов методом эквивалентного разбиения осуществляется в два этапа: выделение классов эквивалентности и построение тестов. При построении тестов, основанных на выборе входных данных, проводится символическое выполнение программы. Одна из целей, которая стоит перед разработчиком тестов — сокращение количества тестов, чтобы уложиться в адекватные сроки тестирования, но при этом не пропустить серьезных ошибок. Именно для этих целей используется техника разбиения на классы эквивалентности, которая и заключается в разбиении всего набора тестов на классы эквивалентности с последующим сокращением числа тестов.

Анализ граничных значений — это проверка ошибок на границах классов эквивалентности. Если техника анализа классов эквивалентности ориентирована на тестовое покрытие, то эта техника основана на рисках. Она основывается на идее о том, что программа может «дать сбой» в области граничных значений, поскольку при разработке большое число проблем возникает именно на границах входных переменных. Даже если эквивалентные классы найдены правильно, то граничные значения могут быть ошибочно отнесены к другому классу. Эффективное применение этой техники зависит от способности правильно выделить классы эквивалентности и затем выбрать тесты для проверки границ этих классов.

Анализ причинно-следственных связей происходит следующим образом. Спецификация разбивается на рабочие участки. В спецификации определяется множество причин и следствий (под причиной понимается отдельное входное условие или класс эквивалентности, следствие представляет собой выходное условие или преобразование системы). На основе анализа семантического содержания спецификации строится таблица истинности, в которой последовательно перебираются всевозможные комбинации причин и определяются следствия для каждой комбинации причин. Недостатком этого подхода является плохое исследование граничных условий.

Предположение об ошибке в значительной степени основано на интуиции. Основная идея метода состоит в том, чтобы составить список, который перечисляет возможные ошибки и ситуации, в которых эти ошибки могли проявиться. После этого на основе списка составляются тесты.

Во время подготовки к тестированию методом «черного ящика» строятся таблицы условий, причинно-следственные графы и области разбивки. Кроме того, подготавливаются тестовые наборы, учитывающие параметры и условия среды, которые влияют на поведение функций.

Задание:

Пусть необходимо выполнить тестирование программы, определяющей точку пересечения двух прямых на плоскости. Попутно, она должна определять параллельность прямой одной их осей координат.

В основе программы лежит решение системы линейных уравнений:

$Ax + By = C$ *и* $Dx + Ey = F$.

1**.** Используя **метод эквивалентных разбиений,** получаем для всех коэффициентов один правильный класс эквивалентности (коэффициент - вещественное

число) и один неправильный (коэффициент - не вещественное число). Откуда можно предложить 7 тестов:

- поочередно каждый из коэффициентов - не вещественное число.

2. По **методу граничных условий**:

можно считать, что для исходных данных граничные условия отсутствуют (коэффициенты - ";любые"; вещественные числа);

для результатов - получаем, что возможны варианты: единственное решение, прямые сливаются (множество решений), прямые параллельны (отсутствие решений). Следовательно, можно предложить тесты, с результатами внутри области:

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Написать и отладить программу.

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Лабораторная работа № 5,6,7,8,9,10, 11,12, 13, 14. Модульное тестирование

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.

- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Краткие теоретические сведения:

Задание:

- Изучить основы разработки модульных тестов
- Получить навыки работы со средствами тестирования Unit Testing Frameworkot Microsoft, NUnit.
- Изучить средства тестирования, доступные в VisualStudio–Unit TestingFramework, NUnit. Допускается использование альтернативных средств (необходимо согласовать с преподавателем).
- Разработать набор unit-тестов для алгоритма в соответствии с номером варианта.
- Реализовать алгоритм в соответствии с номером варианта на любом языке, поддерживаемом платформой .NET. Использование других языков необходимо заранее согласовать с преподавателем.
- Обеспечить максимально возможное покрытие кода тестами.
- Продемонстрировать: Работающую функциональность в соответствии с заданием; Проходящие unit-тесты.

При реализации алгоритма необходимо учитывать следующие требования:

- Объем набора данных –не менее 500 миллионов элементов.
- Потребляемые алгоритмом источники данных задаются интерфейсом IEnumerable<T>.
- Результат работы алгоритма представлен интерфейсом IEnumerable<T>.
- Параметризуемые условия для алгоритма задаются делегатами Func<T, bool>.
- При защите необходимо продемонстрировать понимание реализованного алгоритма

В качестве справочных материалов предлагается набор статей: http://msdn.microsoft.com/ru-ru/library/dd264975.aspx

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Написать и отладить программу.

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным

материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Лабораторная работа №15,16,17,18,19,20,21,22,23. Интеграционное тестирование

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления

Задание:

Провести интеграционное тестирование программы, осуществляющей вычисление системы функций.

1. Все составляющие систему функции, как тригонометрические, так и логарифмические, должны быть выражены через базовые (тригонометрическая зависит от варианта; логарифмическая - натуральный логарифм).
2. Структура приложения, тестируемого в рамках лабораторной работы, должна выглядеть следующим образом (пример приведён для базовой тригонометрической функции sin(x)):

3. Обе ”базовые”функции (в примере выше - sin(x) и ln(x)) должны быть реализованы при помощи разложения в ряд с задаваемой погрешностью. Использовать тригонометрические / логарифмические преобразования для упрощения функций ЗАПРЕЩЕНО. Для КАЖДОГО модуля должны быть реализованы табличные заглушки. При этом, необходимо найти область допустимых значений функций, и, при необходимости, определить взаимозависимые точки в модулях.
4. Разработанное приложение должно позволять выводить значения, выдаваемое любым модулем системы, в файл вида «X, Результаты модуля (X)», позволяющее произвольно менять шаг наращивания X. Разделитель в файле можно использовать произвольный.
Порядок выполнения работы:
1. Разработать приложение, руководствуясь приведёнными выше правилами.
2.С помощью JUNIT4 разработать тестовое покрытие системы функций, проведя анализ эквивалентности и учитывая особенности системы функций. Для анализа особенностей системы функций и составляющих ее частей можно использовать сайт https://www.wolframalpha.com/.
3. Собрать приложение, состоящее из заглушек. Провести интеграцию приложения по 1 модулю, с обоснованием стратегии интеграции, проведением интеграционных тестов и контролем тестового покрытия системы функций.

1.1 Система уравнений

$$\begin{cases} \left(\frac{\left(\left(\left(\cot(x)^2\right)\cdot\sec(x)\right)^2\right)-(\cot(x)-\cos(x))}{\frac{\tan(x)}{\sec(x)+\tan(x)}}\right) & \text{if } x \leq 0 \\ \left(\left(\left(\frac{(\ln(x)-\log_{10}(x))^2}{\log_2(x)}\right)^3\right)\cdot\log_2(x)\right) & \text{if } x > 0 \end{cases}$$

2 Выполнение

2.1 Диаграмма классов

2.2 Описание тестового покрытия

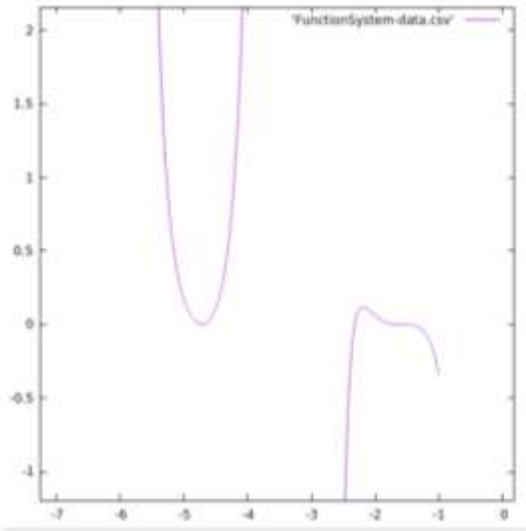
$$\left(\frac{\left(\left(\left(\cot(x)^2\right)\cdot\sec(x)\right)^2\right)-(\cot(x)-\cos(x))}{\frac{\tan(x)}{\sec(x)+\tan(x)}}\right) \quad \text{if} \quad x \leq 0$$

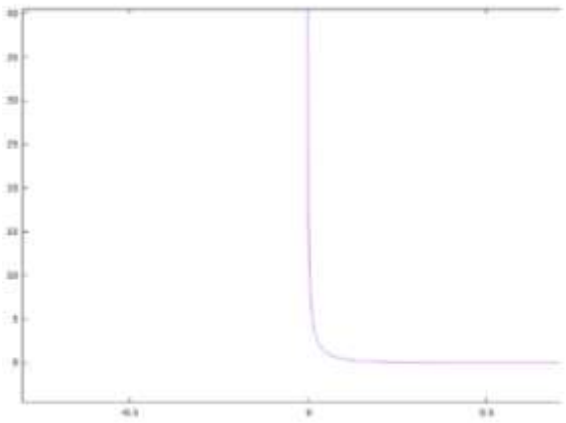
Функция периодичная, определена на $-\frac{\pi}{2} < x - 2\pi m < 0$
Краевые точки:
$\pi < x < 0$
$x = -1,5708, y = 0$
$x = -2,1904, y = 0.1202$
$x = -2,3197, y = 0$
$2\pi < x < \pi$
$x = -4,713, y = 0$

$$\left(\left(\left(\frac{(\ln(x)-\log_{10}(x))^2}{\log_2(x)}\right)^3\right)\cdot\log_2(x)\right) \quad \text{if} \quad x > 0$$

$0 < x < 1$
$x = 0, y = NaN$
$x = 1, y = NaN$
$1 < x < \infty$

2.3 Графики





Выводы

В ходе данной лабораторной работы были получены навыки разработки ПО с использованием интеграционного тестирования

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Написать и отладить программу.

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Тема 1.2.2 Документирование

Лабораторная работа №24,25,26,27,28,29,30,31,32,33. Оформление документации на программные средства с использованием инструментальных средств

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Краткие теоретические сведения:

Виды программных документов

Документирование программного обеспечения осуществляется в соответствии с Единой системой программной документации (ГОСТ 19.XXX). ГОСТ 19.101—77 содержит виды программных документов для программного обеспечения различных типов. В данном ГОСТе перечислены документы следующих типов:

• *спецификация* должна содержать перечень и краткое описание назначения всех файлов программного обеспечения, в том числе и файлов документации на него, и является обязательной для программных систем, а также их компонентов, имеющих самостоятельное применение;

• *ведомость держателей подлинников* (код вида документа — 05) должна содержать список предприятий, на которых хранятся подлинники программных документов. Необходимость этого документа определяется на этапе разработки и утверждения технического задания только для программного обеспечения со сложной архитектурой;

• *текст программы* (код вида документа — 12) должен содержать текст программы с необходимыми комментариями. Необходимость этого документа определяется на этапе разработки и утверждения технического задания;

• *описание программы* (код вида документа — 13) должно содержать сведения о логической структуре и функционировании программы. Необходимость данного документа также определяется на этапе разработки и утверждения технического задания;

• *ведомость эксплуатационных документов* (код вида документа — 20) должна содержать перечень эксплуатационных документов на программу, к которым относятся документы с кодами 30, 31, 32, 33, 34, 35, 46. Необходимость этого документа также определяется на этапе разработки и утверждения технического задания;

• *формуляр* (код вида документа — 30) должен содержать основные характеристики программного обеспечения, комплектность и сведения об эксплуатации программы;

• *описание применения* (код вида документа — 31) должно содержать сведения о назначении программного обеспечения, области применения, применяемых методах, классе решаемых задач, ограничениях для применения, минимальной конфигурации технических средств;

• *руководство системного программиста* (код вида документа — 32) должно содержать сведения для проверки, обеспечения функционирования и настройки программы на условия конкретного применения;
• *руководство программиста* (код вида документа — 33) должно содержать сведения для эксплуатации программного обеспечения;
• *руководство оператора* (код вида документа — 34) содержит сведения для обеспечения процедуры общения оператора с вычислительной системой в процессе выполнения программы;
• *описание языка* (код вида документа — 35) — описание синтаксиса и семантики языка программы;
• *руководство по техническому обслуживанию* (код вида документа — 46) содержит сведения для применения программы при обслуживании технических средств.

Задание:

1. Написать код программ для решения поставленной задачи на языке программирования, выбранном на этапе эскизного проектирования.
2. Отладить программный модуль.
3. Получить результаты работы.
4. Оформить документацию к разработанному программному обеспечению.

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Написать и отладить программу.

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

МДК.01.03 Разработка мобильных приложений

Тема 1.3.1 Основные платформы и языки разработки мобильных приложений Лабораторная работа№ 1,2,3,4,5 Установка инструментария и настройка среды для разработки мобильных приложений

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Разрабатывать программные модули в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода;
- Разрабатывать модули программного обеспечения для мобильных платформ.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.
- Разрабатывать модули программного обеспечения для мобильных платформ.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Задание:

1.Установка и настройка среды разработки

Описание приложения

- Архитектура приложения
- Модель «Дизайн-XML»
- Используемые API
- Установка на устройство
- Отладка

Установить среду разработки Eclipse

- http://www.eclipse.org/downloads/
- Установить JDK — Java Development Kit, http://www.oracle.com/technetwork/java/javase/downloads/index.html
- Для установки ADT откройте панель расширения среды (Install NewSoftware), и затем в появившемся окне добавьте новый путь - https://dl-ssl.google.com/android/eclipse/
- Далее Eclipse выполнит поиск требуемых данных, и в выпадающем списке вы сможете выбрать ADT и установить его. После установки у вас появится AVD Manager. Его ожно запустить прямо из Eclipse (Window-> Android SDK and AVD Manager)

- Настройте Virtual Device
- Установите версии API

Необходимые знания и навыки

- Общие представления об ОС Android
- Знакомство с материалом лекции№ 2
- Базовое знание языка программирования Java

Необходимые программные и аппаратные средства

- ПК под ОС Windows (или любой другой, подходящей для Android SDK)
- Устройство под управлением Android или эмулятор Android SDK

Обзор приложения

- Расположение элементов управления: TabLayout, LinearLayout, RelativeLayout и GridLayout
- Простейшие элементы управления: Button, EditText, TextView
- Список GridView

Архитектура приложения

Обычно, если приложение не имеет сложную структуру, то в одном состоянии приложения используется только один стиль размещения. Мы рассмотрим приложение, которое содержит в себе много различных layouts. Достичь этого, можно путем создания в качестве главного layout, TabLayout, у которого каждый tab имеет свой способ размещения.

l TabLayout →

- Tab1: -> LinearLayout.
- Tab2: -> RelativeLayout.
- ØTab3: -> GridLayout.

Архитектура Tab1

Интерфейс для работы с датой в Андроиде

Компоненты

-TextView - простое текстовое поле, в котором будет отображаться текущая или ручную установленная дата.

- Button - кнопка, которая инициирует появление окошка DitaPicker, в котором отображается текущая дата. В окне стандартно есть контролы позволяющие изменить дату.

Архитектура Tab2

Некоторые виджеты UI

l Компоненты.

-TextView - текстовое поле отображающее статическую информацию.

-EditText - текстовое поле с поддержкой редактирования.

-Button - кнопка OK.

-Button - кнопка Cancel.

- Spinner - выпадающий список с обработкой события выбора.

Архитектура Tab3

Графические объекты (картинки) размещенные в виде таблицы

Компоненты.

-GridView — представление, отображающее элементы в двумерной сетке с поддержкой вертикального или горизонтального скрола. Элементы, ассоциированные с Grid представлением, должны быть представлены с помощью ListAdapter.

Дизайн-XML

-Традиционное разделение кода (Java) и ресурсов (XML) — дизайна, строк и т.п.

-Удобно управлять ресурсами

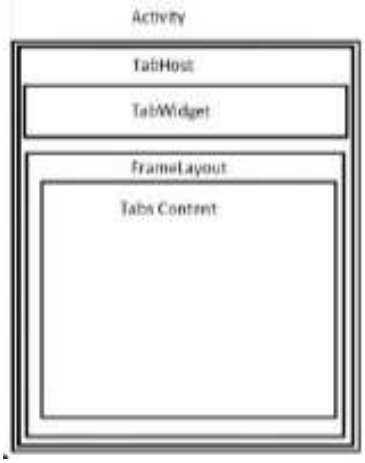
-Упрощение локализации

-Разделение работы программиста и дизайнера

Используемые API
Date Picker — виджет позволяющий работать
Создать этот виджет можно при помощи конструкора:
-DatePickerDialog(Context, DatePickerDialog.
-OnDateSetListener, Year, Month, Day).
Установить дату можно при помощи объекта слушателя DatePickerDialog.OnDateSetListener () , который вызывается всякий раз, когда пользователь нажимает на кнопку «Set». Реализуя метод экземпляра слушателя OnDataSet(DataPicker,year, month, day), мы сможем получить данные о дате из виджета и в дальнейшем работать с ними.

Компонеты Tab2
Spinner — это виджет похожий на выпадающий вниз список, отображающий один элемент и позволяющий выбрать любой другой, среди содержимого раскрывающегося списка.
- Spinner в программе можно создать с помощью конструкторов:
-Spinner(Context, int mode) — параметр mode позволяет задать способ отображения элементов списка MODE_DIALOG или MODE_DROPDOWN.
-Spinner(Context context, AttributeSet attrs, int defStyle, int mode).
Элементы спиннера определяются классом Adapter, а затем присоединяются к нему с помощью метода SetAdapter(adapter).

Tab Layout
Рассмотрение структуры приложения начнем с графических компонентов. В приложении все компоненты UI определяются в xml файле. Основным менеджером размещения в примере является Tab- layout.
Использовать менеджер закладок можно двумя подходами:
- Можно создать представления для каждой закладки в одной Activity(форма приложения), а затем прицепить их к TabHost.
-Можно описать представления конкретными Activity и через главный Activity связать их с TabHost.
В приложении используется первый подход (см. Рис.).



Установка на устройство и отладка

Проделайте следующие шаги, чтобы посмотреть предварительно подготовленное приложение в действии.

1). Импорт приложения. Загрузите пример из каталога lab01 файла labAtom21.rar.

2). Запуск приложения.

Run as->Androidapplication

В результате проект перестроится и запустится приложение. После загрузке эмулятора можно убедиться в работоспособности программы и проанализировать ее код по описанию работы.

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Написать и отладить программу.

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Лабораторная работа№ 6,7,8,9,10,11 Установка среды разработки мобильных приложений с применением виртуальной машины

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Разрабатывать программные модули в соответствии с техническим заданием;

- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода;
- Разрабатывать модули программного обеспечения для мобильных платформ.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.
- Разрабатывать модули программного обеспечения для мобильных платформ.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Задание:

1.Установка и запуск гостевой ОС в виртуальной машине VirtualBox. Виртуальные машины поддерживают установку как с ISO-образа, так и установочного диска ОС. Для установки используем ISO – образ операционной системы Ubuntu (Linux) ubuntu-11.10-desktop-i386.iso.

- Укажем папку для виртуальных машин.
- Необходимо создать виртуальную машину, используя VirtualBox (VMware Workstation, Player или другое ПО на выбор). Установить и запустить гостевую ОС Linux (Дистрибутив Linux либо предложенный, либо на выбор любой другой – Linux Mint, Ubuntu, Debian...). В качестве имени пользователя использовать свою фамилию, пароль – ваше имя. Дополнительно реализовать задание по варианту. В отчет включить описание процесса установки, настройки виртуальной машины и ОС, сопровождая скриншотами.
-

1.Используя проводник файлов Nautilus создать в каталоге /home/имяпользователя/ папку с названием предмета, где будут содержаться отчеты по лабораторным работам. Также научиться производить основные операции над файлами, включая создание, копирование, переименование файлов и удаление. Отсортировать файлы по имени.
2.Используя проводник файлов Nautilus создать в каталоге /home/имяпользователя/ папку для хранения изображений. Также научиться производить основные операции над файлами, включая копирование, переименование файлов и удаление. Отсортировать файлы по размеру.

Порядок выполнения работы

- Составить математическую модель задачи.
- Написать код программ для решения поставленной задачи на языке программирования, выбранном на этапе эскизного проектирования
- Отладить программный модуль.
- Получить результаты работы.

– Оформить документацию к разработанному программному обеспечению.

Форма предоставления результата

– Блок – схема.
– Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Тема 1.3.2 Создание и тестирование модулей для мобильных приложений Лабораторная работа №12. Создание эмуляторов и подключение устройств. Лабораторная работа №13. Настройка режима терминала.

Цель работы:

– Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
– Разрабатывать программные модули в соответствии с техническим заданием;
– Выполнять отладку программных модулей с использованием специализированных программных средств;
– Выполнять тестирование программных модулей;
– Осуществлять рефакторинг и оптимизацию программного кода;
– Разрабатывать модули программного обеспечения для мобильных платформ.

Выполнив работу, Вы будете:

уметь:

– Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
– Оценка сложности алгоритма.
– Создавать программу по разработанному алгоритму как отдельный модуль.
– Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
– Выполнять отладку и тестирование программы на уровне модуля.
– Оформлять документацию на программные средства.
– Применять инструментальные средства отладки программного обеспечения.
– Выполнять оптимизацию и рефакторинг программного кода.
– Работать с системой контроля версий.
– Разрабатывать модули программного обеспечения для мобильных платформ.

Материальное обеспечение:

– Мультимедийные средства хранения, передачи и представления информации.

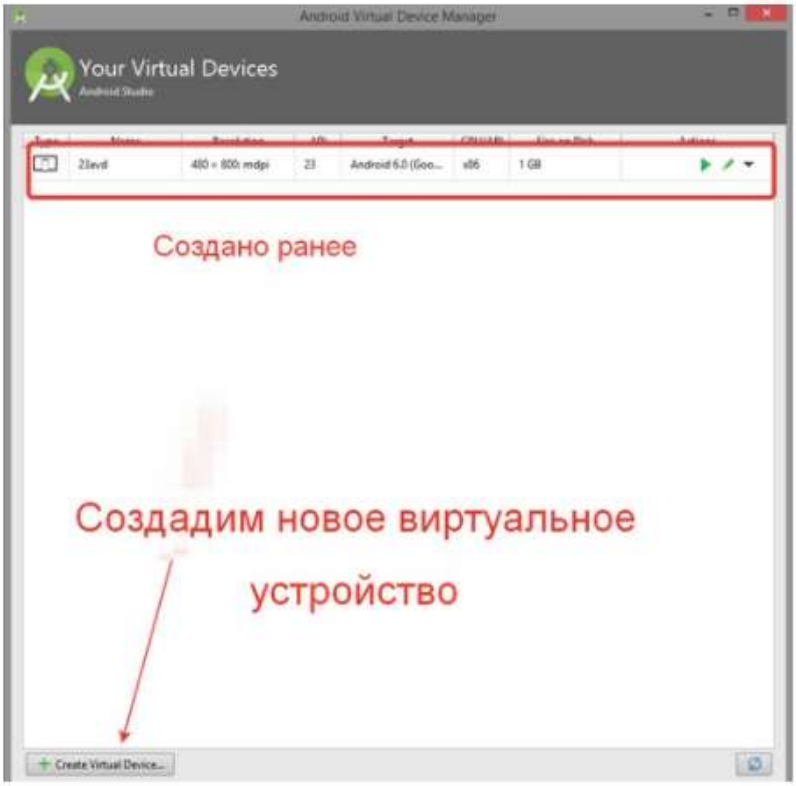
Задание:

– Создать эмулятор Android.

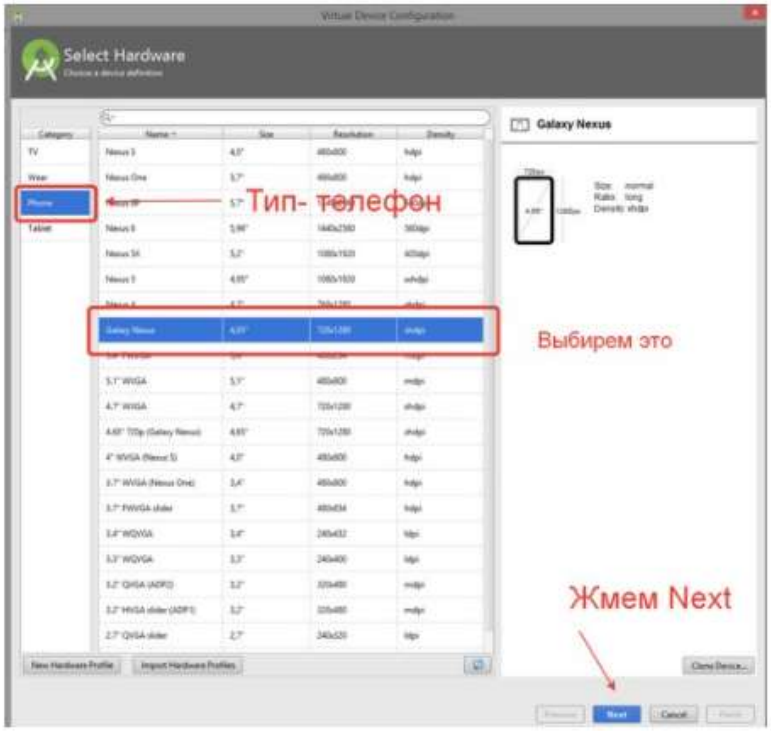
– Настроить эмулятор.

Порядок выполнения работы

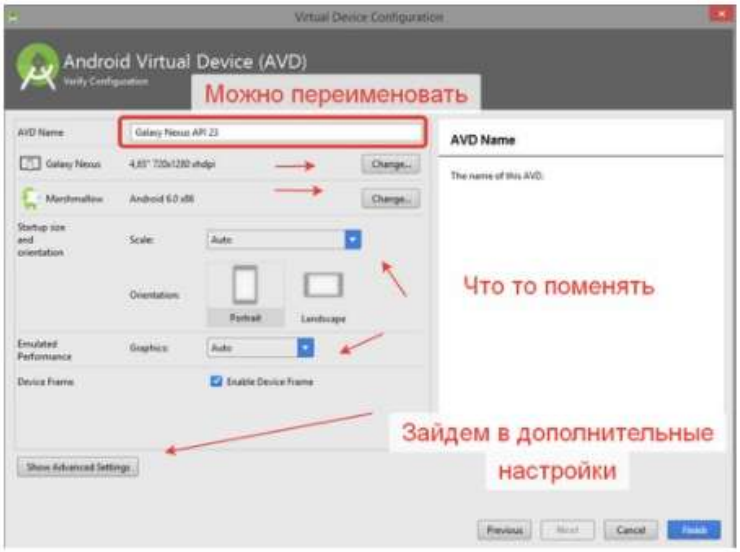
1. Запустить Android Studio, в верхнем меню выбрать Tools->Android->AVD Manader. В нем нажать кнопку Create Virtual Device…



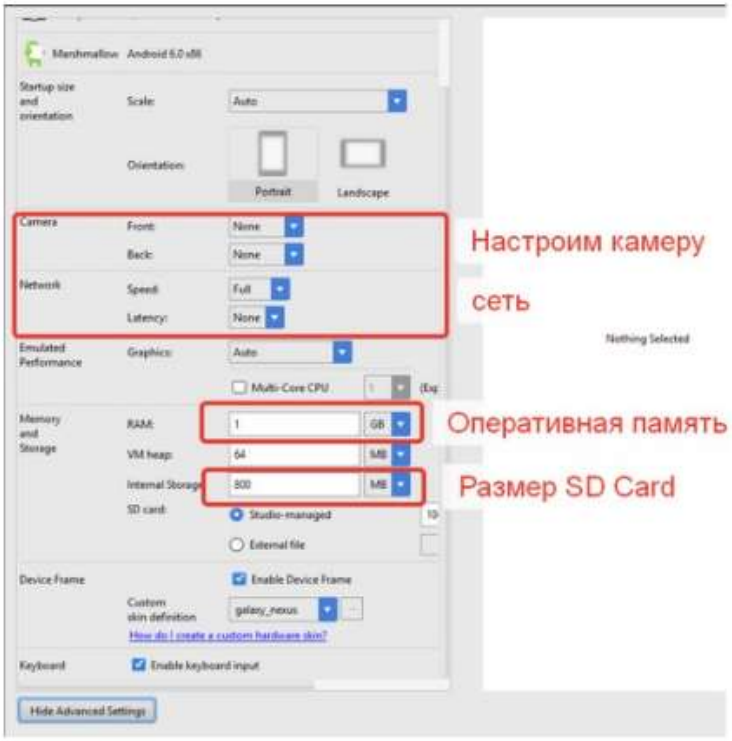
2. В Category выбрать Phone.



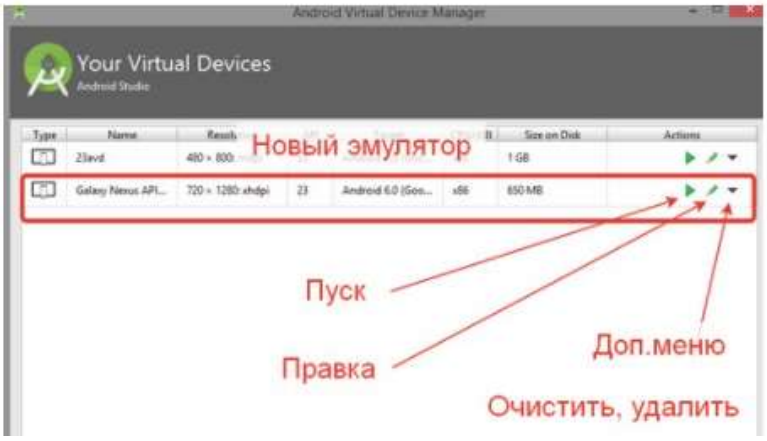
3. Переименовать виртуальное устройств, сменить размер экрана, версию андроида.



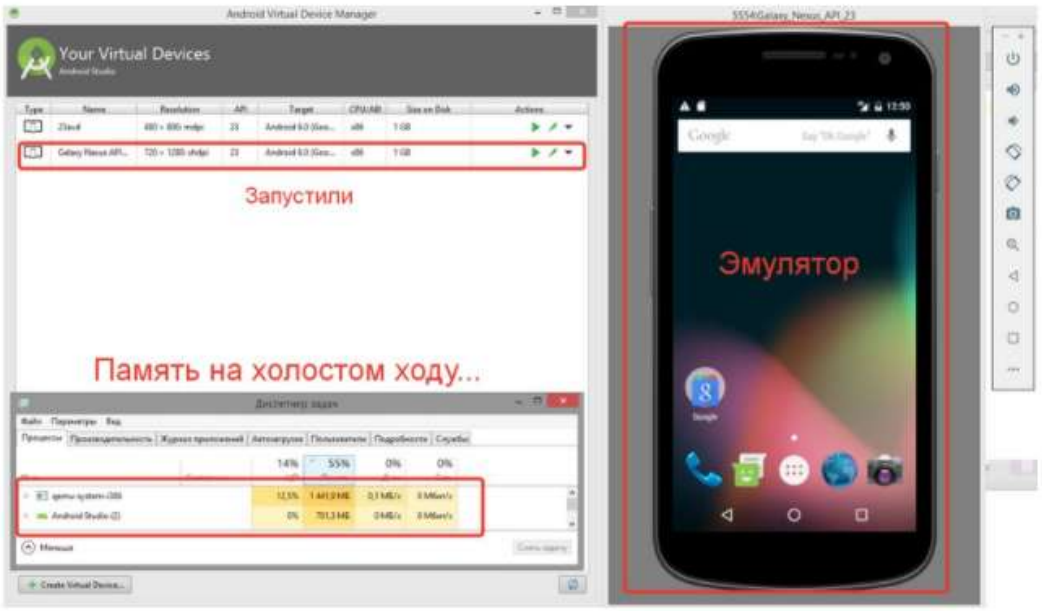
4. Настроить дополнительно Show Advances Setting, на которой будут показаны элементы настройки камеры, сети, можно регулировать объем оперативной памяти, размер SD CARS.



5. При создании нового AVD, создается новое виртуальное устройство.



6. Запустить виртуальное устройство.



Форма предоставления результата

Отчет о проделанной работе.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

–

Лабораторная работа №14. Создание нового проекта.
Лабораторная работа №15. Изучение и комментирование кода.
Лабораторная работа №16. Изменение элементов дизайна

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Разрабатывать программные модули в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода;
- Разрабатывать модули программного обеспечения для мобильных платформ.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.
- Разрабатывать модули программного обеспечения для мобильных платформ.

Материальное обеспечение:

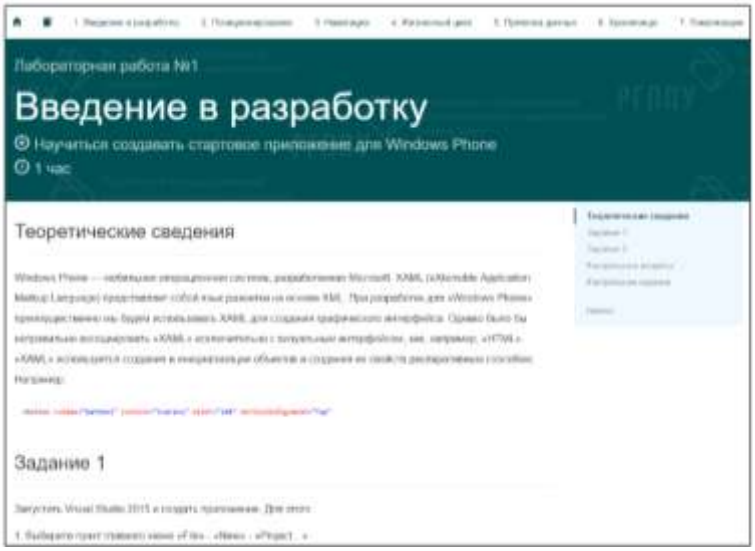
- Мультимедийные средства хранения, передачи и представления информации.

Задание:

1.

– Создать новый проект.
– Создать новое приложение.
– Добавить элементы управления в приложение.
– Запустить приложение.

Лабораторная работапредставлена на рисунке.



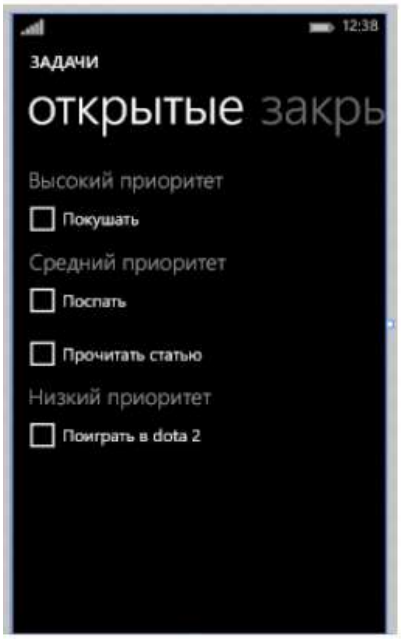
Пример итогового результата выполнения лабораторной работы изображен на рисунке

Рисунок – Вид итогового результата выполнения лабораторной работы N1

Задание 2

– Создать новый проект.
– Изучить разметку.
– Изучить расположение элементов на экране.
– Изучить новые элементы управления.
– Наполнить приложение элементами управления с использованием компоновки.

Пример итогового результата выполнения лабораторной работы изображен на рисунке.



Порядок выполнения работы

- Написать код программ для решения поставленной задачи на языке программирования, выбранном на этапе эскизного проектирования.
- Отладить программный модуль.
- Получить результаты работы.
- Оформить документацию к разработанному программному обеспечению.

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Лабораторная работа №17. Обработка событий: подсказки
Лабораторная работа №18. Обработка событий: цветовая индикация
Лабораторная работа №19. Подготовка стандартных модулей
Лабораторная работа № 20. Обработка событий: переключение между экранами

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Разрабатывать программные модули в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода;
- Разрабатывать модули программного обеспечения для мобильных платформ.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.
- Разрабатывать модули программного обеспечения для мобильных платформ.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Задание:

1.

- Добавить вторую страницу для приложения.
- Изучить новые элементы управления.
- Добавить кнопки, которые будут осуществлять навигацию.
- Создать навигацию между страницами.

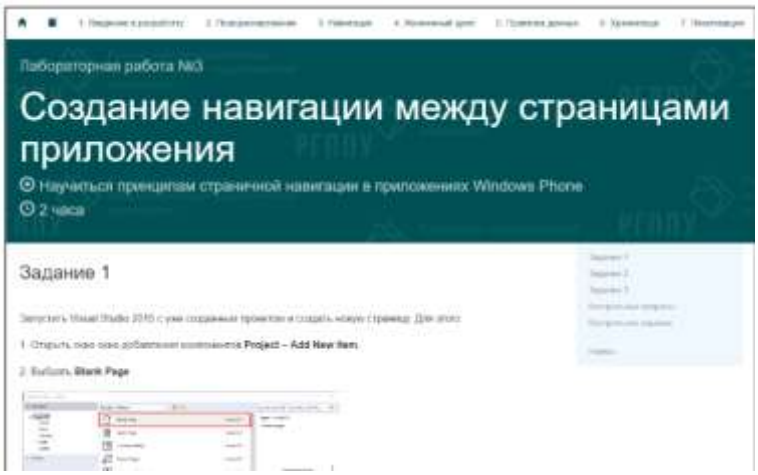
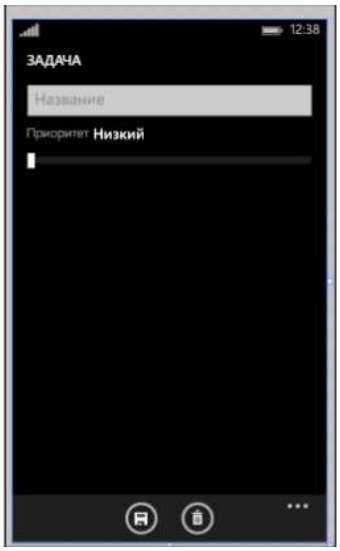


Рисунок – Вид страницы лабораторной работы

Результатом выполнения данной работы является новая страница в приложении с элементами управления и новые кнопки на обеих страницах, которые осуществляют навигацию между этими страницами

.Пример итогового результата выполнения лабораторной работы изображен на рисунке



Порядок выполнения работы

- Написать код программ для решения поставленной задачи на языке программирования, выбранном на этапе эскизного проектирования.
- Отладить программный модуль.
- Получить результаты работы.
- Оформить документацию к разработанному программному обеспечению.

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Лабораторная работа №21. Передача данных между модулями. Лабораторная работа № 22,23. Тестирование и оптимизация мобильного приложения

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Разрабатывать программные модули в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода;
- Разрабатывать модули программного обеспечения для мобильных платформ.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.
- Разрабатывать модули программного обеспечения для мобильных платформ.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Задание:

- Настроить сохранение данных.
- Изменить классы в соответствии с заданием.
- Настроить отображение сохраненных данных.
- Настроить чтение данных.
- Протестировать и оптимизировать приложение.

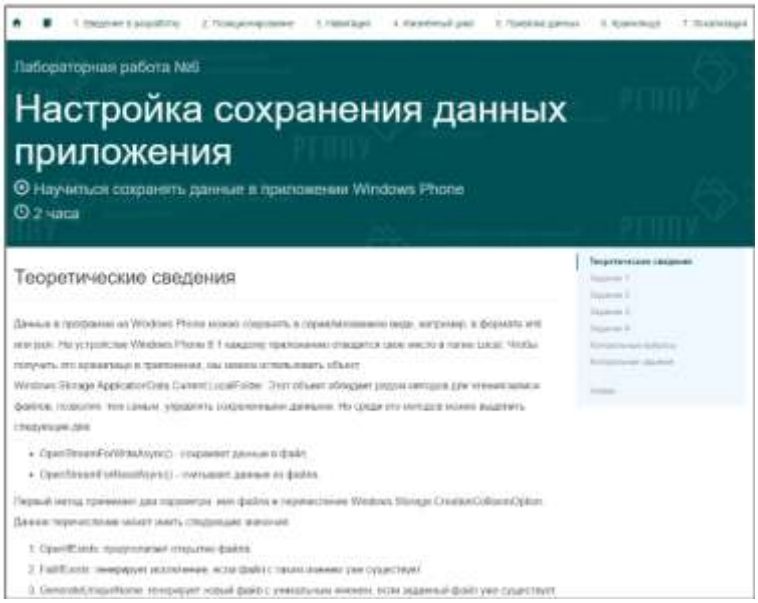
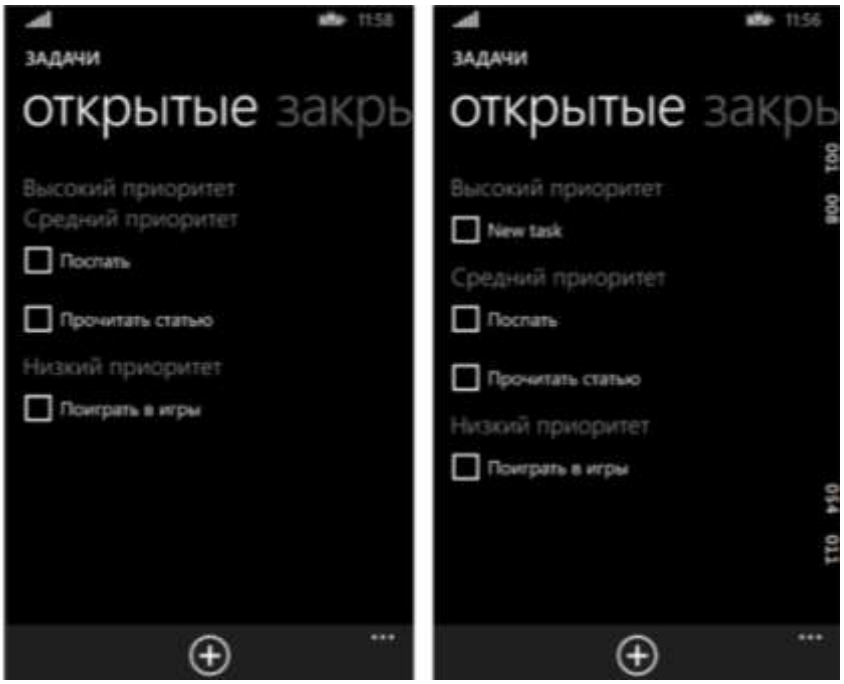


Рисунок – Вид страницы лабораторной работы

Пример итогового результата выполнения лабораторной работы изображен на рисунке.



Порядок выполнения работы

- Написать код программ для решения поставленной задачи на языке программирования, выбранном на этапе эскизного проектирования.
- Отладить программный модуль.
- Получить результаты работы.
- Оформить документацию к разработанному программному обеспечению.

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

МДК.01.04 Системное программирование

Тема 1.4.1 Программирование на языке низкого уровня
Лабораторная работа№ 1,2,3,4,5,6,7. Использование потоков.

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Разрабатывать программные модули в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода;
- Разрабатывать модули программного обеспечения для мобильных платформ.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.
- Разрабатывать модули программного обеспечения для мобильных платформ.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Задание:

1 Должны быть реализованы три потока, каждый из которых осуществляет передвижение собственной надписи по главному окну. Все надписи должны быть различны и двигаться с разной скоростью.

2. Главное окно должно быть разделено на четыре части. Также должны быть созданы четыре потока, каждый из которых раз в секунду изменяет цвет фона в своей части окна.

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Написать и отладить программу.

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Лабораторная работа№ 8,9,10,11,12,13,14. Обмен данными.

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Разрабатывать программные модули в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода;
- Разрабатывать модули программного обеспечения для мобильных платформ.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.
- Разрабатывать модули программного обеспечения для мобильных платформ.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Задание:

1.Запустить несколько заданий (например, команд просмотра файлов less), возвращаясь в командную строку комбинацией клавиш Ctrl-Z и изучить действие команд ps, jobs, fg, bg, kill, killall.

2.Обеспечить синхронизацию процессов и передачу данных между ними на примере двух приложений «клиент» и «сервер», создав два процесса (два исполняемых файла) – процесс «клиент» (первый исполняемый файл) и процесс «сервер» (второй исполняемый файл). С помощью механизмов межпроцессного взаимодействия обеспечить передачу

информации от «клиента» к «серверу» и наоборот. В качестве типа передаваемой информации можно использовать: данные, вводимые с клавиатуры; данные, считываемые из файла; данные, генерируемые случайным образом и
т.п.
3.Обмен данными между процессами «клиент»-«сервер» осуществить следующим образом:
•с использованием программных каналов (именованных либо неименованных, по указанию преподавателя);
•с использованием (по указанию преподавателя) одного из перечисленных вариантов:
•разделяемая память (обязательна синхронизация процессов, например с помощью семафоров);
•очередь сообщений.

1. Создать приложение, состоящее из двух процессов:

−первый процесс записывает текстовый файл и растровый рисунок в буфер обмена;
−второй процесс считывает информацию из буфера обмена и отображает в окне процесса.
Текстовый файл и файл с растровым рисунком взять из предыдущих лабораторных работ 2

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Написать и отладить программу.

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

−«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

−«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

−«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Лабораторная работа№ 15,16,17,18,19,20,21. Сетевое программирование сокетов.

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Разрабатывать программные модули в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;

– Осуществлять рефакторинг и оптимизацию программного кода;
– Разрабатывать модули программного обеспечения для мобильных платформ.

Выполнив работу, Вы будете:

уметь:

– Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.
– Оценка сложности алгоритма.
– Создавать программу по разработанному алгоритму как отдельный модуль.
– Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
– Выполнять отладку и тестирование программы на уровне модуля.
– Оформлять документацию на программные средства.
– Применять инструментальные средства отладки программного обеспечения.
– Выполнять оптимизацию и рефакторинг программного кода.
– Работать с системой контроля версий.
– Разрабатывать модули программного обеспечения для мобильных платформ.

Материальное обеспечение:

– Мультимедийные средства хранения, передачи и представления информации.

Краткие теоретические сведения:

Изучить:

- возможности реализации архитектуры клиент-сервер на основе интерфейса сокетов Windows Sockets API;
- типы сокетов TCP/IP;
- основные методики и API-функции для разработки сетевых приложений с использованием Winsock.

1. Постановка задачи

2.Изучить методические указания к лабораторной работе, материалы лекций и рекомендуемую литературу.

3.Разработать консольное клиент-серверное приложение, демонстрирующее взаимодействие на основе потоковых сокетов.

4.Разработать консольное клиент-серверное приложение, демонстрирующее взаимодействие на основе дейтаграммных сокетов.

Задание 1.

Разработать TCP-сервер, создающий сокет, привязывающий его к локальному IP-адресу и порту и прослушивающий соединения клиентов. Номер порта и IP-адрес вводить с клавиатуры. IP-адрес задавать в десятично-точечной нотации.

Учесть, что функция accept возвращает новый дескриптор сокета, соответствующий принятому клиентскому соединению. Для всех последующих операций с данным клиентским соединением должен применяться новый сокет. Исходный прослушивающий сокет используется для приема других клиентских соединений и продолжает находиться в режиме прослушивания. При получении от клиента запроса на установление соединения вывести на экран IP-адрес и номер порта клиентского сокета.

Разработать приложение–клиент, соединяющееся с заданным TCP-сервером. Все отправленные и полученные по сокету данные вывести на экран.

При вызове API-функций выполнять проверку и обработку ошибок.

Нарисовать блок-схему серверной и клиентской части программы.

Задание 2.

Разработать серверное приложение, выполняющее получение данных через сокет без установления соединения по протоколу UDP.

Разработать клиентское приложение, выполняющее отправку UDP-дейтаграмм. IP-адрес и порт удаленного получателя должен задаваться пользователем.

При вызове API-функций выполнять проверку и обработку ошибок.

Адреса сокетов вывести на экран.

Нарисовать блок-схему серверной и клиентской части программы.

2. Написать программные коды
3. Запустить и отладить программу

Варианты заданий:

1) Удаленный калькулятор (+,-,*,/);
2) Работа с массивом чисел (количество элементов в массиве, получение значения элемента массива по номеру, найти номер элемента в массиве по значению, увеличить значения элементов на заданное число);

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Написать и отладить программу.

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.

Лабораторная работа№22 23,24,25,26,27. Работы с буфером экрана.

Цель работы:

- Научиться формировать алгоритмы разработки программных модулей в соответствии с техническим заданием;
- Разрабатывать программные модули в соответствии с техническим заданием;
- Выполнять отладку программных модулей с использованием специализированных программных средств;
- Выполнять тестирование программных модулей;
- Осуществлять рефакторинг и оптимизацию программного кода;
- Разрабатывать модули программного обеспечения для мобильных платформ.

Выполнив работу, Вы будете:

уметь:

- Формировать алгоритмы разработки программных модулей в соответствии с техническим заданием.

- Оценка сложности алгоритма.
- Создавать программу по разработанному алгоритму как отдельный модуль.
- Осуществлять разработку кода программного модуля на языках низкого уровня и высокого уровней в том числе для мобильных платформ.
- Выполнять отладку и тестирование программы на уровне модуля.
- Оформлять документацию на программные средства.
- Применять инструментальные средства отладки программного обеспечения.
- Выполнять оптимизацию и рефакторинг программного кода.
- Работать с системой контроля версий.
- Разрабатывать модули программного обеспечения для мобильных платформ.

Материальное обеспечение:

- Мультимедийные средства хранения, передачи и представления информации.

Задание1:

Создайте сценарий WMI, выполняющий запись в файл сведений о количестве процессоров и скорости процессора, о размерах КЭШей различных уровней.

Задание2:

Использовать классы - Win32_Processor, Win32_CacheMemory

Создайте сценарий WMI, выполняющий запись в файл сведений о скринсейвере и разрешении экрана, наименование клавиатуры и количество функциональных клавиш.

Использовать классы - Клавиатура (Win32_Keyboard), Монитор (Win32_DesktopMonitor)

Порядок выполнения работы

- Составить математическую модель задачи.
- Выбрать и обосновать наиболее рациональный метод решения задачи;
- Разработать алгоритм для решения задачи.
- Написать и отладить программу.

Форма предоставления результата

- Блок – схема.
- Код программы.

Критерии оценки:

«Отлично» - теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко.

–«Хорошо» - теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками.

–«Удовлетворительно» - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки.

–«Неудовлетворительно» - теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.