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

Многопрофильный колледж

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ЛАБОРАТОРНЫХ ЗАНЯТИЙ УЧЕБНОЙ ДИСЦИПЛИНЫ

ОП.06 ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ

для обучающихся специальности
09.02.01 Компьютерные системы и комплексы

Магнитогорск,2023

ОДОБРЕНО

Предметно-цикловой комиссией «Информатики и вычислительной техники»
Председатель Т.Б. Ремез
Протокол № 6 от 25.01.2023 г.

Методической комиссией МпК
Протокол № 4 от 08.02.2023 г.

Разработчик (и):
преподаватель отделения № 2 «Информационных технологий и транспорта»
Многопрофильного колледжа ФГБОУ ВО «МГТУ им. Г.И. Носова» Р.Н. Урманова
преподаватель отделения № 2 «Информационных технологий и транспорта»
Многопрофильного колледжа ФГБОУ ВО «МГТУ им. Г.И. Носова» С.М. Утралинова

Методические указания по выполнению лабораторных работ разработаны на основе рабочей программы учебной дисциплины / профессионального модуля «ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ».

Содержание практических занятий ориентировано на подготовку обучающихся к освоению профессионального(ых) модуля(ей) программы подготовки специалистов среднего звена по специальности 09.02.01 Компьютерные системы и комплексы и овладению профессиональными компетенциями.

Содержание

1 ВВЕДЕНИЕ

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

Состав и содержание практических занятий направлены на реализацию Федерального государственного образовательного стандарта среднего профессионального образования.

Ведущей дидактической целью практических занятий является формирование профессиональных практических умений (умений выполнять определенные действия, операции, необходимые в последующем в профессиональной деятельности), необходимых в последующей учебной деятельности.

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

В соответствии с рабочей программой учебной дисциплины «Основы алгоритмизации и программирования» предусмотрено проведение практических занятий.

В результате их выполнения, обучающийся должен:

уметь:

- применять стандартные алгоритмы в соответствующих областях;
- применять выбранные языки программирования для написания программного кода;
- использовать выбранную среду программирования;
- выявлять дефекты и отклонения в функционировании программного обеспечения компьютерных систем и комплексов.

Содержание практических занятий ориентировано на подготовку обучающихся к освоению профессионального модуля программы подготовки специалистов среднего звена по специальности и овладению ***профессиональными компетенциями:***

ПК 2.1 Проектировать, разрабатывать и отлаживать программный код модулей управляющих программ.

ПК 2.2 Владеть методами командной разработки программных продуктов.

ПК 3.2. Проверять работоспособность, выполнять обнаружение и устранять дефекты программного кода управляющих программ компьютерных систем и комплексов.

А также формированию ***общих компетенций***:

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

ОК 02 - Использовать современные средства поиска, анализа и интерпретации информации и информационные технологии для выполнения задач профессиональной деятельности.

ОК 04 - Эффективно взаимодействовать и работать в коллективе и команде.

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

ОК 09 - Пользоваться профессиональной документацией на государственном и иностранном языках.

Выполнение обучающихся практических работ по учебной дисциплине «Основы алгоритмизации и программирования» 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.*

Практические занятия проводятся в рамках соответствующей темы, после освоения дидактических единиц, которые обеспечивают наличие знаний, необходимых для ее выполнения.

2 МЕТОДИЧЕСКИЕ УКАЗАНИЯ

Тема 1.1. Основные понятия алгоритмизации. Основные алгоритмические конструкции.

Практическое занятие № 1,2,3,4
Построение блок схем основных алгоритмических конструкций.

Цель:

- Научиться применять базовые понятия теории приближенных методов решения задач на ЭВМ.
- Сформировать представление об алгоритмах решения прикладных задач, научиться написанию и отладки программного кода.
- Выбрать и обосновать наиболее рациональный метод и алгоритм решения задачи;
- Разработать алгоритм для решения прикладных задач;
- Написать программный код и отладить программу.

Выполнив работу, Вы будете:

уметь:

- применять стандартные алгоритмы в соответствующих областях;
- применять выбранные языки программирования для написания программного кода;
- использовать выбранную среду программирования;
- выявлять дефекты и отклонения в функционировании программного обеспечения компьютерных систем и комплексов.

Материальное обеспечение:

- Персональные компьютеры;
- Комплекты роботехнические "ПервоРобот NXT";
- Комплект робототехнический "LEGO";
- Контроллер 500995 ROBO TX;
- Набор аккумуляторный Accu Set;
- Наборы конструкторские ROBO TX;
- Датчик цвета для микрокомпьютера NXT;
- Программное обеспечение общего и профессионального назначения, в том числе включающее в себя следующее ПО:

MS Windows 7 (подписка Imagine Premium)
MS Office 2007
7 Zip
MS Visual Studio (подписка Imagine Premium)
Visual Studio Code
Calculate Linux Desktop

Задание:

1 Составить различные формы представления алгоритмов на примере вычисления корней квадратного уравнения $ax^2 + bx + c = 0$.

Порядок выполнения работы:

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

– Написать и отладить программу.

Ход работы:

Алгоритмы классифицируются по форме представления (рис. 2.1).

Рис. 2.1. Классификация алгоритмов по форме представления

Словесный способ записи алгоритмов представляет собой описание последовательных этапов обработки данных. Алгоритм задается в произвольном изложении на естественном языке.

Словесный алгоритм вычисления корней квадратного уравнения $ax^2 + bx + c = 0$ будет иметь следующий вид:

1. Задаем коэффициенты уравнения a, b, c;
2. Если значение коэффициента $a \neq 0$, то вычисляем дискриминант по формуле $D = b^2 - 4ac$;
3. Если выполняется условие $D > 0$, то корни квадратного уравнения x_1 и x_2 вычисляем по формуле $x_{1,2} = \frac{-b \pm \sqrt{D}}{2a}$;
4. Если выполняется условие $D = 0$, то вычисляем один корень квадратного уравнения x_1 по формуле $x_1 = \frac{-b}{2a}$;
5. Если выполняется условие $D < 0$, то выводим сообщение «Нет действительных корней».

Табличные алгоритмы оформляются в виде таблицы и используются для развития культуры оформления решения задач, для отображения связи с другими предметами и для формирования точности, аккуратности и пунктуальности В табл.2.1 приведена табличная форма вычисления корней квадратного уравнения $ax^2 + bx + c = 0$.

Таблица 2.1

Табличная форма вычисления корней квадратного уравнения $ax^2 + bx + c = 0$

Шаг	Алгоритм	Образец выполнения
1	Внимательно прочитайте текст задачи	Найдите корни квадратного уравнения вида $x^2 + 5x - 6 = 0$
2	Запишите в «Дано» буквенное обозначение и	Дано:

Шаг	Алгоритм	Образец выполнения
	числовое значение известных по тексту коэффициентов	$a = 1, b = 5,$ $c = -6$
3	Под горизонтальной чертой запишите буквенное обозначение неизвестной величины со знаками «=» и «?»	x_1 = ? x_2 = ?
4	Если значение коэффициента $a \neq 0$, то по формуле $D = b^2 - 4ac$ вычислите дискриминант.	Так как $a \neq 0$ $(a = 1)$, $D = 5^2 - 4 \cdot 1 \cdot (-6)$ $D = 49$
5	– если выполняется условие $D > 0$, то корни квадратного уравнения x_1 и x_2 вычисляем по формуле $x_{1,2} = \frac{-b \pm \sqrt{D}}{2a}$, – если выполняется условие $D = 0$, то вычисляем один корень квадратного уравнения x_1 по формуле $x_1 = \frac{-b}{2a}$, – если выполняется условие $D < 0$, то выводим сообщение «Нет действительных корней».	Так как $D > 0$ $(D = 49)$, $x_1 = \frac{-5+\sqrt{49}}{2 \cdot 1} = 1$ $x_2 = \frac{-5-\sqrt{49}}{2 \cdot 1} = -6$
6	Запишите ответ	Ответ: $x_1 = 1$ и $x_2 = -6$.

Графический способ представления алгоритмов (в виде блок-схемы) наиболее распространенная форма записи алгоритмов. При этом представлении алгоритм изображается в виде последовательности связанных между собой функциональных блоков, каждый из которых соответствует выполнению одного или нескольких действий. На рис.2.2 представлена блок-схема вычисления корней квадратного уравнения $ax^2 + bx + c = 0$.

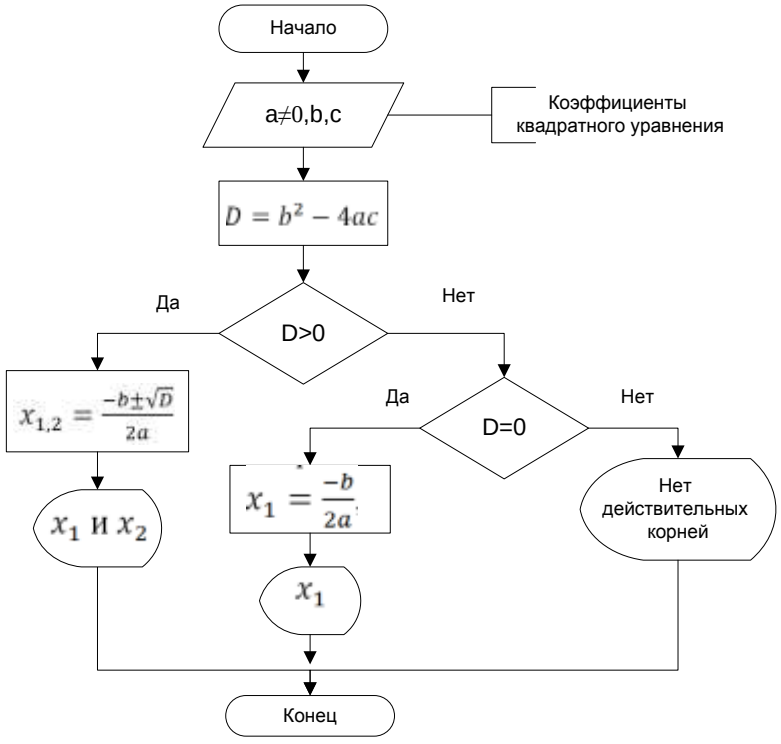


Рис.2.2. Блок-схема алгоритма решения квадратного уравнения

Форма записи алгоритма в виде псевдокодов представляет собой полуформализованные описания алгоритмов на условном алгоритмическом языке, включающие в себя как элементы языка программирования, так и фразы естественного языка, общепринятые математические обозначения и др.

Алгоритмический язык – формальный язык, используемый для записи, реализации или изучения алгоритмов.

Также понятием алгоритмический язык иногда называют:

- семейство языков программирования Алгол;
- учебный алгоритмический язык (школьный алгоритмический язык, русский алгоритмический язык);
- ДРАКОН – Дружелюбный Русский Алгоритмический язык, Который Обеспечивает Наглядность.

Во второй половине 1980-х годов под руководством академика А.П.Ершова была разработана методика обучения программированию в средней и высшей школе и, непосредственно, сам школьный алгоритмический язык КуМир (Комплект Учебных МИРов). Это простой алголоподобный язык с русской лексикой и встроенными командами управления программными исполнителями (Робот, Чертёжник).

При вводе программы КуМир осуществляет постоянный полный контроль ее правильности, сообщая на полях программы обо всех обнаруженных ошибках.

При выполнении программы в пошаговом режиме КуМир выводит на поля результаты операций присваивания и значения логических выражений. Это позволяет ускорить процесс освоения азов программирования.

КуМир работает в операционных системах Windows или Linux.

Ниже представлен пример алгоритма решения квадратного уравнения с применением алгоритмического языка КуМир.

```
алг Квад_кор (арг вещ a,b,c, рез вещ x1,x2)
нач
| ввод a,b,c
    вещ D
    D:=b*b-4*a*c
    Выбор
    при D> 0: x1:= (-b+sqrt(D))/2*a;x2:= (-b-sqrt(D))/2*a
    при D=0: x1:= -b/2*a
    иначе вывод "Нет действительный корней"
    все
кон
```

Все ранее рассмотренные алгоритмы могут допускать неточности при изображении команд, что, в свою очередь, может привести к неполному пониманию процесса. Как было сказано выше, основным исполнителем алгоритмов в современном мире является компьютер, что требует записи алгоритма на понятном ему языке.

Следовательно, язык для записи алгоритмов должен быть формализован. Такой язык принято называть языком программирования, а запись алгоритма на этом языке - программой для компьютера.

Программа, создаваемая человеком-программистом, представляет собой текст, состоящий из знаков, как правило, букв, цифр и специальных знаков. Знаки в тексте программы часто объединены в последовательности – ключевые слова, слова объединены в предложения языка программирования – операторы. Каждый оператор, как правило, записывается в отдельную строку текста программы.

Таким образом, текстовое программирование представляет собой иерархическую последовательность знаков, слов, операторов, записываемых и читаемых последовательно, как обычный текст человеческой письменности. Ниже приведены примеры программного

способа записи алгоритмов на языках программирования С++ (среда разработки Microsoft Visual Studio).

```
#include "stdafx.h"
#include <stdio.h>
#include <math.h>
int _tmain(int argc, _TCHAR* argv[])
{
	float a, b, c,d;
	printf("Input a, b, c: ");
	scanf("%f%f%f",&a,&b,&c);
	if (a!=0)
	{

	  //Вычисляем дискриминант}
	d = b*b - 4*a*c;
		printf("\nd =%5.2f",d);
		    // Если дискриминант больше 0,
		    //то вычисляем корни и выводим на экран
		if (d > 0)
		{
			float x1 = (-b - d) /2 /a;
			float x2 = (-b + d) /2 /a;
			printf("\nx1 =%5.2f",x1);
			printf("\tx2 =%5.2f",x2);
		}
		    // Если дискриминант равен 0,
		    //то вычисляем один корень и выводим на экран
		else if (d == 0)
		{
			float x = (-b) /2 /a;
		printf("\nx =%5.2f",x);
		}
		    // Если дискриминант меньше 0,
		    //то выводим сообщение
		else if (d < 0)
			printf("\nNo valid roots");

		getchar();getchar();
		return 0;
	}
}
```

Алгоритмы классифицируются по структуре (рис. 2.3).

Линейный алгоритм – алгоритм, все этапы которого выполняются однократно и строго последовательно.

Разветвленный алгоритм – это алгоритм, включающий выбор тех или иных действий в зависимости от какого-либо условия.

Циклический алгоритм – это алгоритм, в котором предусмотрено многократное выполнение одной и той же последовательности действий.

Вспомогательный алгоритм – это алгоритм, созданный ранее и вызываемый из основного алгоритма как команда.

Комбинированный алгоритм – это алгоритм, в котором встречаются два вида алгоритма циклический и разветвляющий.

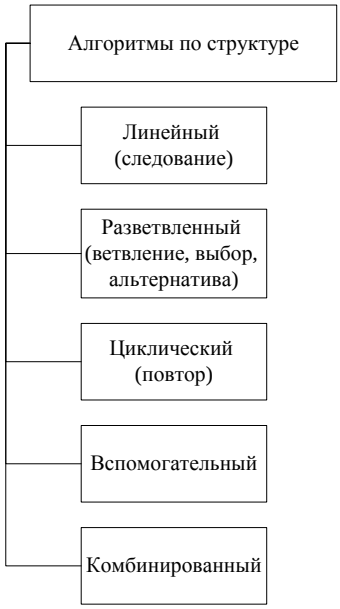


Рис.2.3. Классификация алгоритмов по структуре

Форма представления результата:

- Алгоритм программы.
- 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 2.2. Ввод и вывод данных

Практическое занятие № 5,6
Операции ввода – вывода

Цель:

1. Научиться применять базовые понятия теории приближенных методов решения задач на ЭВМ.
2. Сформировать представление об алгоритмах решения прикладных задач, научится написанию и отладки программного кода.
3. Выбрать и обосновать наиболее рациональный метод и алгоритм решения задачи;
4. Разработать алгоритм для решения прикладных задач;
5. Написать программный код и отладить программу.

Выполнив работу, Вы будете:

уметь:

- применять стандартные алгоритмы в соответствующих областях;
- применять выбранные языки программирования для написания программного кода;
- использовать выбранную среду программирования;
- выявлять дефекты и отклонения в функционировании программного обеспечения компьютерных систем и комплексов.

Материальное обеспечение:

- Персональные компьютеры;
- Комплекты роботехнические "ПервоРобот NXT";
- Комплект робототехнический "LEGO";
- Контроллер 500995 ROBO TX;
- Набор аккумуляторный Accu Set;
- Наборы конструкторские ROBO TX;
- Датчик цвета для микрокомпьютера NXT;
- Программное обеспечение общего и профессионального назначения, в том числе включающее в себя следующее ПО:

MS Windows 7 (подписка Imagine Premium)
MS Office 2007
7 Zip
MS Visual Studio (подписка Imagine Premium)
Visual Studio Code
Calculate Linux Desktop

Задание:

1. Какого типа будет результат деления 15 на 4?
2. Какие из приведенных ниже операторов присвоения являются правильными?
   а. x:=I+J-B
   б. I:=I+K/J
   если известно, что I,J,K int; x,B float?
3. Написать программу вычисления выражения

$$\frac{5.23 + 7.6^2 + \sin\frac{\pi}{7}}{\sin\frac{2\pi}{7} + 3.1}$$

Дать протокол программы.

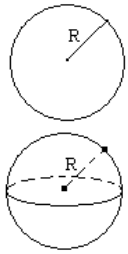
Задача

1. Формулировка задачи

Составить алгоритм вычисления длины окружности, площади круга, площади сферы и объема шара по заданному радиусу окружности.

Ход работы:

2. Математическая постановка задачи



Для расчета перечисленных характеристик воспользуемся формулами:

длина окружности – $L = 2\pi R$;

площадь круга – $S_{кр} = \pi R^2$;

площадь сферы – $S_{сф} = 4\pi R^2$;

объем шара – $V = \frac{4}{3}\pi R^3$,

где π – число Пи, математическая константа, которая выражает отношение длины окружности к её диаметру $\pi \approx 3{,}14159265358979323846264 3\ldots$, *R* – радиус окружности.

3. Выбор переменных программы

Из приведенного выше решения определяем следующие переменные:

исходные данные – радиус окружности (*R*);

справочные данные – число π (*Pi*);

результат – длина окружности (*L*), площадь круга ($S_{кр}$), площадь сферы ($S_{сф}$) и объем шара (*V*).

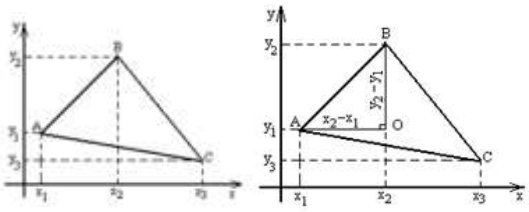
4. Блок-схема алгоритма

5. Код программы.

Задача

1. Формулировка задачи

Треугольник задается координатами своих вершин на плоскости: *A(x1,y1), B(x2,y2), C(x3,y3).* Требуется составить алгоритм программы, которая вычисляет площадь треугольника *ABC*.



Ход работы:

2. Математическая постановка задачи

Для решения задачи можно использовать формулу Герона: $S = \sqrt{p(p-A)(p-B)(p-C)}$, где *p* – полупериметр. Для вычисления длины сторон по координатам вершин необходимо воспользоваться формулой $|AB| = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$, где $|AB|$ – длина сторона треугольника; (x_1,y_1), (x_2,y_2) – координаты вершин *A и B*.

3. Выбор переменных программы

Из приведенного выше решения определяем следующие переменные:
исходные данные – координаты вершин на плоскости (x_1,y_1), (x_2,y_2), (x_3,y_3);
промежуточные данные – длины сторон (A, B, C), p – полупериметр.
результат – площадь треугольника (S).

4. Блок-схема алгоритма

5. . Код программы.

Порядок выполнения работы:

1. Составить математическую модель задачи.
2. Выбрать и обосновать наиболее рациональный метод решения задачи;
3. Разработать алгоритм для решения задачи.
4. Написать и отладить программу.

Форма представления результата:

1. Алгоритм программы.
2. 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 2.3. Базовые конструкции языков программирования

Практические работы №7,8
Оператор условия

Цель:

1. Научиться применять базовые понятия теории приближенных методов решения задач на ЭВМ.
2. Сформировать представление об алгоритмах решения прикладных задач, научится написанию и отладки программного кода.
3. Выбрать и обосновать наиболее рациональный метод и алгоритм решения задачи;
4. Разработать алгоритм для решения прикладных задач;
5. Написать программный код и отладить программу.

Выполнив работу, Вы будете:

уметь:

- применять стандартные алгоритмы в соответствующих областях;
- применять выбранные языки программирования для написания программного кода;
- использовать выбранную среду программирования;
- выявлять дефекты и отклонения в функционировании программного обеспечения компьютерных систем и комплексов.

Материальное обеспечение:

- Персональные компьютеры;
- Комплекты роботехнические "ПервоРобот NXT";
- Комплект робототехнический "LEGO";
- Контроллер 500995 ROBO TX;
- Набор аккумуляторный Accu Set;
- Наборы конструкторские ROBO TX;
- Датчик цвета для микрокомпьютера NXT;
- Программное обеспечение общего и профессионального назначения, в том числе включающее в себя следующее ПО:

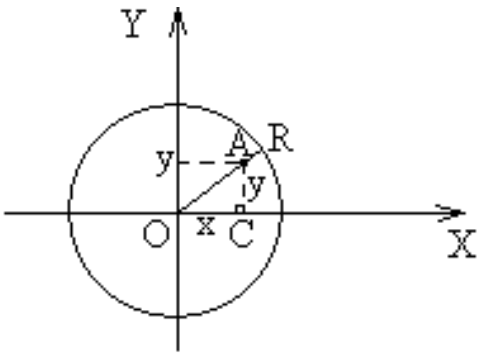
MS Windows 7 (подписка Imagine Premium)
MS Office 2007
7 Zip
MS Visual Studio (подписка Imagine Premium)
Visual Studio Code
Calculate Linux Desktop

Задание:

1. Составить алгоритм решения задачи, который определяет по введенным координатам, попадает ли заданная точка в окружность с центром в точке O (0;0) и заданным радиусом R.

Ход работы:

2. Математическая постановка задачи



Заданная точка имеет координаты x, y. Рассмотрим треугольник AOC. ∠OCA=90º, OC=x, AC=y, следовательно, по теореме Пифагора:
AO^2 (гипотенуза)= OC^2+AC^2 ($AO^2=x^2+y^2$).
Поэтому условие принадлежности точки окружности можно записать в виде:$x^2+y^2 \le R^2$.

3. Выбор переменных программы

Из приведенного выше решения определяем следующие переменные:

исходные данные – радиус окружности (*R*) и координаты точки (*x* и *y*);

результат – сообщение «В окружности» или «Вне окружности».

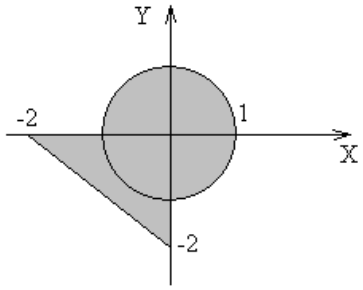
Так как радиус окружности и координаты точки могут принимать любые значения (2; 2,5; 3,75), все переменные для данной задачи определены как действительные числа.

3. Блок-схема алгоритма.
4. Код программы.

Задание:

1. Формулировка задачи

Составить алгоритм решения задачи, который по введенным координатам точки определяет, попадает ли эта точка в заштрихованную область.



Ход работы:

2. Математическая постановка задачи

Запишем условия попадания точки в область в виде формул. Область можно описать как круг, пересекающийся с треугольником. Точка может попадать либо в круг, либо в треугольник, либо в их общую часть. Условие попадание точки в круг запишем как $x^2 + y^2 \le 1^2$, т.к. радиус окружности в соответствии с графической интерпретацией равен 1.

Так как треугольник расположен в третьей четверти координатной плоскости, следовательно, для него выполняется условие $\begin{cases} x \le 0 \\ y \le 0 \end{cases}$. Выведем уравнение прямой линии, ограничивающей треугольник снизу.

Известно, что прямая линия проходит через две точки: (-2;0) и (0;-2). Требуется составить уравнение прямой линии.

Общий вид уравнения прямой $y = kx + b$, где $k, x,$ – фиксированные числа.

Искомая прямая $y = kx + b$ с пока неизвестными коэффициентами $k, x,$ проходит через точки (-2;0) и (0;-2), а, значит, выполняются равенства

$0 = k \cdot (-2) + b$ и $-2 = k \cdot 0 + b$, что можно записать в виде системы: $\begin{cases} 0 = k \cdot (-2) + b \\ -2 = k \cdot 0 + b \end{cases}$ или $\begin{cases} 0 = k \cdot (-2) + b \\ -2 = b \end{cases}$.

Решив систему относительно неизвестных *k и b* , мы найдем уравнение прямой.

Подставим значение $b = -2$ в уравнение $0 = k \cdot (-2) + b$:

$-2k = -b$

$-2k = -(-2)$

$-2k = 2$

$k = -1$

Таким образом, искомое уравнение прямой имеет вид $y = (-1) \cdot x - 2$. Так как заштрихованная область расположена выше прямой линии, ограничивающей треугольник снизу, то попадание точки в треугольник запишем как $\begin{cases} x \leq 0 \\ y \leq 0 \\ y \geq -x - 2 \end{cases}$. Следовательно, заштрихованная область определена следующими условиями $x^2 + y^2 \leq 1^2$ или $\begin{cases} x \leq 0 \\ y \leq 0 \\ y \geq -x - 2 \end{cases}$.

3. Выбор переменных программы

Из приведенного выше решения определяем следующие переменные:

исходные данные – координаты точки (*x* и *y*);

результат – сообщение «Принадлежит» или «Не принадлежит».

Так как координаты точки могут принимать любые значения (2; 2,5; 3,75), все переменные определяем как действительные числа.

4. Блок-схема алгоритма.

5. Код программы.

Порядок выполнения работы:

1. Составить математическую модель задачи.
2. Выбрать и обосновать наиболее рациональный метод решения задачи;
3. Разработать алгоритм для решения задачи.
4. Написать и отладить программу.

Форма представления результата:

1. Алгоритм программы.
2. 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 2.3. Базовые конструкции языков программирования

Практическое занятие №9
Оператор цикла с предусловием

Цель:

1. Научиться применять базовые понятия теории приближенных методов решения задач на ЭВМ.
2. Сформировать представление об алгоритмах решения прикладных задач, научится написанию и отладки программного кода.
3. Выбрать и обосновать наиболее рациональный метод и алгоритм решения задачи;
4. Разработать алгоритм для решения прикладных задач;
5. Написать программный код и отладить программу.

Выполнив работу, Вы будете:

уметь:

- применять стандартные алгоритмы в соответствующих областях;
- применять выбранные языки программирования для написания программного кода;
- использовать выбранную среду программирования;
- выявлять дефекты и отклонения в функционировании программного обеспечения компьютерных систем и комплексов.

Материальное обеспечение:

- Персональные компьютеры;
- Комплекты роботехнические "ПервоРобот NXT";
- Комплект робототехнический "LEGO";
- Контроллер 500995 ROBO TX;
- Набор аккумуляторный Accu Set;
- Наборы конструкторские ROBO TX;
- Датчик цвета для микрокомпьютера NXT;
- Программное обеспечение общего и профессионального назначения, в том числе включающее в себя следующее ПО:

MS Windows 7 (подписка Imagine Premium)
MS Office 2007
7 Zip
MS Visual Studio (подписка Imagine Premium)
Visual Studio Code
Calculate Linux Desktop

Задание:

1. Формулировка задачи

Составить алгоритм расчета факториала N! с использованием различных видов цикла.

Ход работы:

2. Математическая постановка задачи

Факториал числа N (обозначается N!) – произведение всех натуральных чисел до N включительно:

$$N!=1\cdot2\cdot3\cdot\ldots\cdot N$$

По определению полагают 0! = 1. Факториал определён только для целых неотрицательных чисел.

3. Выбор переменных программы

Из приведенного выше решения определяем следующие переменные:

исходные данные – значение *N*;

результат – значение *factorial*.

Выбор переменных: аргумент и значение функции могут принимать целые неотрицательные значения.

4. Блок-схема алгоритма.

5. Код программы.

Задание:

1. Формулировка задачи

Составить алгоритм вычисления суммы ряда:

$y = x - \frac{x^2}{2} + \frac{x^3}{3} + \text{K} + \frac{(-1)^{n-1} x^n}{n}$ с точностью *e=0.0001* с использованием цикла с постусловием и предусловием.

Ход работы:

2. Математическая постановка задачи

Ряд представляет собой сумму слагаемых, в котором перед каждым слагаемым чередуется знак (перед первым слагаемым знак «+», перед вторым – «-», перед третьим – «+», перед четвертым – «-», и т.д.). Поэтому для его учета в алгоритм накопления суммы вводим переменную, отвечающую за знак – z.

При накоплении суммы, начальное значение суммы обнуляем, т.е. S=0. Первое слагаемое можно представить в виде $\frac{x^1}{1}$, т.е. начальное значение степени x и знаменатель равны 1. Шаг изменения степени и знаменателя совпадает и равен 1. Точность вычисления определяется значением слагаемого.

3. Выбор переменных программы

Из приведенного выше описания определяем следующие переменные:

исходные данные – значение *x*;

начальное значение суммы S равно 0 (S=0);

начальное значение степени x и знаменатель равны 1 (i=1);

знак перед первым слагаемым – «+» (z=1);

результат – значение S.

4. Блок-схема алгоритма.

5. Код программы.

Порядок выполнения работы:

1. Составить математическую модель задачи.
2. Выбрать и обосновать наиболее рациональный метод решения задачи;
3. Разработать алгоритм для решения задачи.
4. Написать и отладить программу.

Форма представления результата:

1. Алгоритм программы.
2. 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 2.3. Базовые конструкции языков программирования

Практическое занятие № 10
Оператор цикла с постусловием

Цель:

1. Научиться применять базовые понятия теории приближенных методов решения задач на ЭВМ.
2. Сформировать представление об алгоритмах решения прикладных задач, научится написанию и отладки программного кода.
3. Выбрать и обосновать наиболее рациональный метод и алгоритм решения задачи;
4. Разработать алгоритм для решения прикладных задач;
5. Написать программный код и отладить программу.

Выполнив работу, Вы будете:

уметь:

- применять стандартные алгоритмы в соответствующих областях;
- применять выбранные языки программирования для написания программного кода;
- использовать выбранную среду программирования;
- выявлять дефекты и отклонения в функционировании программного обеспечения компьютерных систем и комплексов.

Материальное обеспечение:

- Персональные компьютеры;
- Комплекты роботехнические "ПервоРобот NXT";
- Комплект робототехнический "LEGO";
- Контроллер 500995 ROBO TX;
- Набор аккумуляторный Accu Set;
- Наборы конструкторские ROBO TX;
- Датчик цвета для микрокомпьютера NXT;
- Программное обеспечение общего и профессионального назначения, в том числе включающее в себя следующее ПО:

MS Windows 7 (подписка Imagine Premium)
MS Office 2007
7 Zip
MS Visual Studio (подписка Imagine Premium)
Visual Studio Code
Calculate Linux Desktop

Заполнение массива с помощью генератора случайных чисел

Для генерации чисел в диапазоне *[A, B]* можно использовать следующее выражение: *random(B-A+1)+A.*

Например, выражение A[i]=random(100) генерирует числа от 0 до 99, при этом 100 не входит в диапазон.

Записать выражение для генерации чисел в диапазоне *[–15; 38]:*
A[i]=random(38-(-15)+1)+(-15), т.е. A[i]=random(54)-15.

Задание:

1. Формулировка задачи

Заполнить массив *n* целыми случайными числами в диапазоне [–10; 15]. Составить алгоритм вычисления суммы четных чисел.

Ход работы:

2. Математическая постановка задачи

При накоплении суммы, начальное значение суммы обнуляем, т.е. Sum=0.
Выражение для генерации чисел в диапазоне *[–10; 15]:*
A[i]=random(15-(-10)+1)+(-10), т.е. запишем в следующем виде A[i]=random(26)-10.
Для четного значения элемента выполняется условие A[i] mod 2=0.

3. Выбор переменных программы

Из приведенного выше описания определяем следующие переменные:
исходные данные – количество элементов массива n;
начальное значение суммы равно 0 (Sum=0);
результат – значение накопленной суммы Sum.

4. Блок-схема алгоритма.

5. Код программы.

Порядок выполнения работы:

1. Составить математическую модель задачи.
2. Выбрать и обосновать наиболее рациональный метод решения задачи;
3. Разработать алгоритм для решения задачи.
4. Написать и отладить программу.

Форма представления результата:

1. Алгоритм программы.
2. 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 2.3. Базовые конструкции языков программирования

Практическое занятие № 11
Оператор цикла с параметром

Цель:

1. Научиться применять базовые понятия теории приближенных методов решения задач на ЭВМ.
2. Сформировать представление об алгоритмах решения прикладных задач, научится написанию и отладки программного кода.
3. Выбрать и обосновать наиболее рациональный метод и алгоритм решения задачи;
4. Разработать алгоритм для решения прикладных задач;
5. Написать программный код и отладить программу.

Выполнив работу, Вы будете:

уметь:

- применять стандартные алгоритмы в соответствующих областях;
- применять выбранные языки программирования для написания программного кода;
- использовать выбранную среду программирования;
- выявлять дефекты и отклонения в функционировании программного обеспечения компьютерных систем и комплексов.

Материальное обеспечение:

- Персональные компьютеры;
- Комплекты роботехнические "ПервоРобот NXT";
- Комплект робототехнический "LEGO";
- Контроллер 500995 ROBO TX;
- Набор аккумуляторный Accu Set;
- Наборы конструкторские ROBO TX;
- Датчик цвета для микрокомпьютера NXT;
- Программное обеспечение общего и профессионального назначения, в том числе включающее в себя следующее ПО:

MS Windows 7 (подписка Imagine Premium)
MS Office 2007
7 Zip
MS Visual Studio (подписка Imagine Premium)
Visual Studio Code
Calculate Linux Desktop

Задание:

1. Формулировка задачи

Заполнить массив n целыми случайными числами в диапазоне [–5; 10]. Составить алгоритм вычисления произведения положительных элементов.

Ход работы:

2. Математическая постановка задачи

При накоплении произведения, начальное значение произведения равно 1, т.е. pr=1.
Выражение для генерации чисел в диапазоне [–5; 10]:
A[i]=random(10-(-5)+1)+(-5), т.е. запишем в следующем виде A[i]=random(16)-5.
Для положительного значения элемента выполняется условие A[i] >0.

3. Выбор переменных программы

Из приведенного выше описания определяем следующие переменные:

исходные данные – количество элементов массива n;

начальное значение произведения равно 1 (pr=1);

результат – значение произведения pr.

4. Блок-схема алгоритма.

5. Код программы.

Порядок выполнения работы:

1. Составить математическую модель задачи.
2. Выбрать и обосновать наиболее рациональный метод решения задачи;
3. Разработать алгоритм для решения задачи.
4. Написать и отладить программу.

Форма представления результата:

3. Алгоритм программы.
4. 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 2.4. Массивы

Практическое занятие № 12
Работа со строками

Цель:

1. Научиться применять базовые понятия теории приближенных методов решения задач на ЭВМ.
2. Сформировать представление об алгоритмах решения прикладных задач, научится написанию и отладки программного кода.
3. Выбрать и обосновать наиболее рациональный метод и алгоритм решения задачи;
4. Разработать алгоритм для решения прикладных задач;
5. Написать программный код и отладить программу.

Выполнив работу, Вы будете:

уметь:

- применять стандартные алгоритмы в соответствующих областях;
- применять выбранные языки программирования для написания программного кода;
- использовать выбранную среду программирования;
- выявлять дефекты и отклонения в функционировании программного обеспечения компьютерных систем и комплексов.

Материальное обеспечение:

- Персональные компьютеры;
- Комплекты роботехнические "ПервоРобот NXT";
- Комплект робототехнический "LEGO";
- Контроллер 500995 ROBO TX;
- Набор аккумуляторный Accu Set;
- Наборы конструкторские ROBO TX;
- Датчик цвета для микрокомпьютера NXT;
- Программное обеспечение общего и профессионального назначения, в том числе включающее в себя следующее ПО:

MS Windows 7 (подписка Imagine Premium)
MS Office 2007
7 Zip
MS Visual Studio (подписка Imagine Premium)
Visual Studio Code
Calculate Linux Desktop

Задание:

1. Формулировка задачи

- Дана строка, заканчивающаяся точкой. Подсчитать, сколько слов в строке.
- *Лишние пробелы.* Дана строка, состоящая из слов, разделенных пробелами. Напишите программу, удаляющую лишние пробелы. Пробел считается лишним, если он: стоит в начале строки.

Ход работы:

2. Блок-схема алгоритма.
3. Код программы.

Задание:

1. Формулировка задачи:

- Дана строка. Подсчитать, сколько в ней букв *R, K,T.*
- *Лишние пробелы.* Дана строка, состоящая из слов, разделенных пробелами. Напишите программу, удаляющую лишние пробелы. Пробел считается лишним, если он: следует за пробелом.

Ход работы:

2. Выбор переменных программы
3. Блок-схема алгоритма.
4. Код программы

Порядок выполнения работы:

1. Составить математическую модель задачи.
2. Выбрать и обосновать наиболее рациональный метод решения задачи;
3. Разработать алгоритм для решения задачи.
4. Написать и отладить программу.

Форма представления результата:

1. Алгоритм программы.
2. 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 2.4. Массивы

Практическое занятие № 13,14,15
Алгоритмы поиска, сортировки и замены

Цель:

1. Научиться применять базовые понятия теории приближенных методов решения задач на ЭВМ.
2. Сформировать представление об алгоритмах решения прикладных задач, научится написанию и отладки программного кода.
3. Выбрать и обосновать наиболее рациональный метод и алгоритм решения задачи;
4. Разработать алгоритм для решения прикладных задач;
5. Написать программный код и отладить программу.

Выполнив работу, Вы будете:

уметь:

- применять стандартные алгоритмы в соответствующих областях;
- применять выбранные языки программирования для написания программного кода;
- использовать выбранную среду программирования;
- выявлять дефекты и отклонения в функционировании программного обеспечения компьютерных систем и комплексов.

Материальное обеспечение:

- Персональные компьютеры;
- Комплекты роботехнические "ПервоРобот NXT";
- Комплект робототехнический "LEGO";
- Контроллер 500995 ROBO TX;
- Набор аккумуляторный Accu Set;
- Наборы конструкторские ROBO TX;
- Датчик цвета для микрокомпьютера NXT;
- Программное обеспечение общего и профессионального назначения, в том числе включающее в себя следующее ПО:

MS Windows 7 (подписка Imagine Premium)
MS Office 2007
7 Zip
MS Visual Studio (подписка Imagine Premium)
Visual Studio Code
Calculate Linux Desktop

Задание:

1. Формулировка задачи:

Составить алгоритм и написать программу:

Вычислить сумму и число положительных элементов матрицы A[N, N], находящихся над главной диагональю.

Ход работы:

2. Выбор переменных программы
3. Блок-схема алгоритма.
4. Код программы

Задание:

1. Формулировка задачи:

Составить алгоритм и написать программу:

Дана матрица B[N, M].. Найти в каждой строке матрицы максимальный и минимальный элементы и поменять их местами с первым и последним элементом строки соответственно.

Ход работы:

2. Выбор переменных программы
3. Блок-схема алгоритма.
4. Код программы

Порядок выполнения работы:

1. Составить математическую модель задачи.
2. Выбрать и обосновать наиболее рациональный метод решения задачи;
3. Разработать алгоритм для решения задачи.
4. Написать и отладить программу.

Форма представления результата:

1. Алгоритм программы.
2. 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 2.5. Функции

Практическое занятие № 16
Параметры функции

Цель:

1. Научиться применять базовые понятия теории приближенных методов решения задач на ЭВМ.
2. Сформировать представление об алгоритмах решения прикладных задач, научится написанию и отладки программного кода.
3. Выбрать и обосновать наиболее рациональный метод и алгоритм решения задачи;
4. Разработать алгоритм для решения прикладных задач;
5. Написать программный код и отладить программу.

Выполнив работу, Вы будете:

уметь:

- применять стандартные алгоритмы в соответствующих областях;
- применять выбранные языки программирования для написания программного кода;
- использовать выбранную среду программирования;
- выявлять дефекты и отклонения в функционировании программного обеспечения компьютерных систем и комплексов.

Материальное обеспечение:

- Персональные компьютеры;
- Комплекты роботехнические "ПервоРобот NXT";
- Комплект робототехнический "LEGO";
- Контроллер 500995 ROBO TX;
- Набор аккумуляторный Accu Set;
- Наборы конструкторские ROBO TX;
- Датчик цвета для микрокомпьютера NXT;
- Программное обеспечение общего и профессионального назначения, в том числе включающее в себя следующее ПО:

MS Windows 7 (подписка Imagine Premium)
MS Office 2007
7 Zip
MS Visual Studio (подписка Imagine Premium)
Visual Studio Code
Calculate Linux Desktop

Задание:

1. Формулировка задачи:

Написать функцию, выводящую в порядке возрастания элементы одномерного массива. В главной программе вызвать функцию для двух разных массивов.

Ход работы:

2. Выбор переменных программы
3. Блок-схема алгоритма.
4. Код программы

Задание:

1. Формулировка задачи:

Написать функцию вычисления произведения прямоугольной матрицы A размера k x m на прямоугольную матрицу B размера m x n. В главной программе обратиться к этой функции.

Ход работы:

2. Выбор переменных программы
3. Блок-схема алгоритма.
4. Код программы

Порядок выполнения работы:

1. Составить математическую модель задачи.
2. Выбрать и обосновать наиболее рациональный метод решения задачи;
3. Разработать алгоритм для решения задачи.
4. Написать и отладить программу.

Форма представления результата:

1. Алгоритм программы.
2. 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 2.5. Функции

Практическое занятие № 17
Рекурсивные функции

Цель:

1. Научиться применять базовые понятия теории приближенных методов решения задач на ЭВМ.
2. Сформировать представление об алгоритмах решения прикладных задач, научится написанию и отладки программного кода.
3. Выбрать и обосновать наиболее рациональный метод и алгоритм решения задачи;
4. Разработать алгоритм для решения прикладных задач;
5. Написать программный код и отладить программу.

Выполнив работу, Вы будете:

уметь:

- применять стандартные алгоритмы в соответствующих областях;
- применять выбранные языки программирования для написания программного кода;
- использовать выбранную среду программирования;
- выявлять дефекты и отклонения в функционировании программного обеспечения компьютерных систем и комплексов.

Материальное обеспечение:

- Персональные компьютеры;
- Комплекты роботехнические "ПервоРобот NXT";
- Комплект робототехнический "LEGO";
- Контроллер 500995 ROBO TX;
- Набор аккумуляторный Accu Set;
- Наборы конструкторские ROBO TX;
- Датчик цвета для микрокомпьютера NXT;
- Программное обеспечение общего и профессионального назначения, в том числе включающее в себя следующее ПО:

MS Windows 7 (подписка Imagine Premium)
MS Office 2007
7 Zip
MS Visual Studio (подписка Imagine Premium)
Visual Studio Code
Calculate Linux Desktop

Задание:

1. Формулировка задачи:

Ввести с клавиатуры целое число N. Используя рекурсию, распечатать сначала последовательность, состоящую из N букв 'A', а затем из N букв 'B'.

Ход работы:

2. Выбор переменных программы
3. Блок-схема алгоритма.
4. Код программы

Задание:

1.Формулировка задачи:

Напечатать в обратном порядке последовательность чисел, признаком конца которой является 0.

Ход работы:

2. Выбор переменных программы
3. Блок-схема алгоритма.
4. Код программы

Порядок выполнения работы:

1. Составить математическую модель задачи.
2. Выбрать и обосновать наиболее рациональный метод решения задачи;
3. Разработать алгоритм для решения задачи.
4. Написать и отладить программу.

Форма представления результата:

1. Алгоритм программы.
2. 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 2.5. Функции

Практическое занятие № 18
Многофайловые проекты

Цель:

1. Научиться применять базовые понятия теории приближенных методов решения задач на ЭВМ.
2. Сформировать представление об алгоритмах решения прикладных задач, научится написанию и отладки программного кода.
3. Выбрать и обосновать наиболее рациональный метод и алгоритм решения задачи;
4. Разработать алгоритм для решения прикладных задач;
5. Написать программный код и отладить программу.

Выполнив работу, Вы будете:

уметь:

- применять стандартные алгоритмы в соответствующих областях;
- применять выбранные языки программирования для написания программного кода;
- использовать выбранную среду программирования;
- выявлять дефекты и отклонения в функционировании программного обеспечения компьютерных систем и комплексов.

Материальное обеспечение:

- Персональные компьютеры;
- Комплекты роботехнические "ПервоРобот NXT";
- Комплект робототехнический "LEGO";
- Контроллер 500995 ROBO TX;
- Набор аккумуляторный Accu Set;
- Наборы конструкторские ROBO TX;
- Датчик цвета для микрокомпьютера NXT;
- Программное обеспечение общего и профессионального назначения, в том числе включающее в себя следующее ПО:

MS Windows 7 (подписка Imagine Premium)
MS Office 2007
7 Zip
MS Visual Studio (подписка Imagine Premium)
Visual Studio Code
Calculate Linux Desktop

Задание:

1. Формулировка задачи:

Записать в файл прямого доступа ряд вещественных чисел. Найти наибольшее из значений модулей компонентов с нечетными номерами.

Ход работы:

2. Выбор переменных программы
3. Блок-схема алгоритма.
4. Код программы

Задание:

1. Формулировка задачи:

Записать в файл последовательного доступа произвольных натуральных чисел. Переписать в другой файл последовательного доступа те элементы, которые кратны К.

Вывести полученный файл на печать.

Ход работы:

2. Выбор переменных программы
3. Блок-схема алгоритма.
4. Код программы

Порядок выполнения работы:

1. Составить математическую модель задачи.
2. Выбрать и обосновать наиболее рациональный метод решения задачи;
3. Разработать алгоритм для решения задачи.
4. Написать и отладить программу.

Форма представления результата:

1. Алгоритм программы.
2. 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 2.5. Функции

Практическое занятие № 19
Работа с указателями.

Цель:

1. Научиться применять базовые понятия теории приближенных методов решения задач на ЭВМ.
2. Сформировать представление об алгоритмах решения прикладных задач, научится написанию и отладки программного кода.
3. Выбрать и обосновать наиболее рациональный метод и алгоритм решения задачи;
4. Разработать алгоритм для решения прикладных задач;
5. Написать программный код и отладить программу.

Выполнив работу, Вы будете:

уметь:

- применять стандартные алгоритмы в соответствующих областях;
- применять выбранные языки программирования для написания программного кода;
- использовать выбранную среду программирования;
- выявлять дефекты и отклонения в функционировании программного обеспечения компьютерных систем и комплексов.

Материальное обеспечение:

- Персональные компьютеры;
- Комплекты роботехнические "ПервоРобот NXT";
- Комплект робототехнический "LEGO";
- Контроллер 500995 ROBO TX;
- Набор аккумуляторный Accu Set;
- Наборы конструкторские ROBO TX;
- Датчик цвета для микрокомпьютера NXT;
- Программное обеспечение общего и профессионального назначения, в том числе включающее в себя следующее ПО:

MS Windows 7 (подписка Imagine Premium)
MS Office 2007
7 Zip
MS Visual Studio (подписка Imagine Premium)
Visual Studio Code
Calculate Linux Desktop

Задание:

1. Формулировка задачи:

Дана последовательность целых чисел, оканчивающаяся нулем. Записать все числа последовательности в типизированный файл.

В конец существующего типизированного файла записать:

а) число 0;

б) фразу «До свидания!».

Ход работы:

2. Выбор переменных программы
3. Блок-схема алгоритма.
4. Код программы

Задание:

1. Формулировка задачи:

Создать типизированный файл, элементами которого являются двенадцать первых членов последовательности Фибоначчи (последовательности, в которой первые два члена равны 1, а каждый следующий равен сумме двух предыдущих).

Ход работы:

2. Выбор переменных программы
3. Блок-схема алгоритма.
4. Код программы

Порядок выполнения работы:

1. Составить математическую модель задачи.
2. Выбрать и обосновать наиболее рациональный метод решения задачи;
3. Разработать алгоритм для решения задачи.
4. Написать и отладить программу.

Форма представления результата:

1. Алгоритм программы.
2. 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 2.5. Функции

Практическое занятие № 20
Динамическое распределение памяти.

Цель:

1. Научиться применять базовые понятия теории приближенных методов решения задач на ЭВМ.
2. Сформировать представление об алгоритмах решения прикладных задач, научится написанию и отладки программного кода.
3. Выбрать и обосновать наиболее рациональный метод и алгоритм решения задачи;
4. Разработать алгоритм для решения прикладных задач;
5. Написать программный код и отладить программу.

Выполнив работу, Вы будете:

уметь:

- применять стандартные алгоритмы в соответствующих областях;
- применять выбранные языки программирования для написания программного кода;
- использовать выбранную среду программирования;
- выявлять дефекты и отклонения в функционировании программного обеспечения компьютерных систем и комплексов.

Материальное обеспечение:

- Персональные компьютеры;
- Комплекты роботехнические "ПервоРобот NXT";
- Комплект робототехнический "LEGO";
- Контроллер 500995 ROBO TX;
- Набор аккумуляторный Accu Set;
- Наборы конструкторские ROBO TX;
- Датчик цвета для микрокомпьютера NXT;
- Программное обеспечение общего и профессионального назначения, в том числе включающее в себя следующее ПО:

MS Windows 7 (подписка Imagine Premium)
MS Office 2007
7 Zip
MS Visual Studio (подписка Imagine Premium)
Visual Studio Code
Calculate Linux Desktop

Задание:

1. Формулировка задачи:

- Разработать программу, используя модульный подход.

-Сформировать по одному или несколько заголовочных модулей и файлов реализации, которые будут содержать разработанные функции.

-Написать главную функцию main, используя правила структурного подхода: она должна содержать в основном только вызов функций, все вычисления должны быть внутри разработанных функций.

-Выполнить отладку и компиляцию программы, получить исполняемый файл.

-Выполнить тестирование программы несколькими наборами входных данных.

-Составить диаграмму модулей, дерево вызова функций и спецификации функций

Даны целые числа a1, ... , an, b1, ..., bm, k.

Если в последовательности a1, ... , an нет ни одного члена со значением k, то первый по порядку20 член этой последовательности, не меньший всех остальных членов, заменить на значение k.

По такому же правилу преобразовать последовательность b1, ..., bm применительно к 10.

Ход работы:

2. Выбор переменных программы
3. Блок-схема алгоритма.
4. Код программы

Порядок выполнения работы:

1. Составить математическую модель задачи.
2. Выбрать и обосновать наиболее рациональный метод решения задачи;
3. Разработать алгоритм для решения задачи.
4. Написать и отладить программу.

Форма представления результата:

1. Алгоритм программы.
2. 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 3.2. Директивы и операторы ассемблера

Практическое занятие № 21
Работа и использование отладчика AFDP: Основные команды

Цель:

1. Научиться применять базовые понятия теории приближенных методов решения задач на ЭВМ.
2. Сформировать представление об алгоритмах решения прикладных задач, научится написанию и отладки программного кода.
3. Выбрать и обосновать наиболее рациональный метод и алгоритм решения задачи;
4. Разработать алгоритм для решения прикладных задач;
5. Написать программный код и отладить программу.

Выполнив работу, Вы будете:

уметь:

- применять стандартные алгоритмы в соответствующих областях;
- применять выбранные языки программирования для написания программного кода;
- использовать выбранную среду программирования;
- выявлять дефекты и отклонения в функционировании программного обеспечения компьютерных систем и комплексов.

Материальное обеспечение:

- Персональные компьютеры;
- Комплекты роботехнические "ПервоРобот NXT";
- Комплект робототехнический "LEGO";
- Контроллер 500995 ROBO TX;
- Набор аккумуляторный Accu Set;
- Наборы конструкторские ROBO TX;
- Датчик цвета для микрокомпьютера NXT;
- Программное обеспечение общего и профессионального назначения, в том числе включающее в себя следующее ПО:

MS Windows 7 (подписка Imagine Premium)
MS Office 2007
7 Zip
MS Visual Studio (подписка Imagine Premium)
Visual Studio Code
Calculate Linux Desktop

Задание:

Выполнить следующие команды и прокомментировать результаты их выполнения:

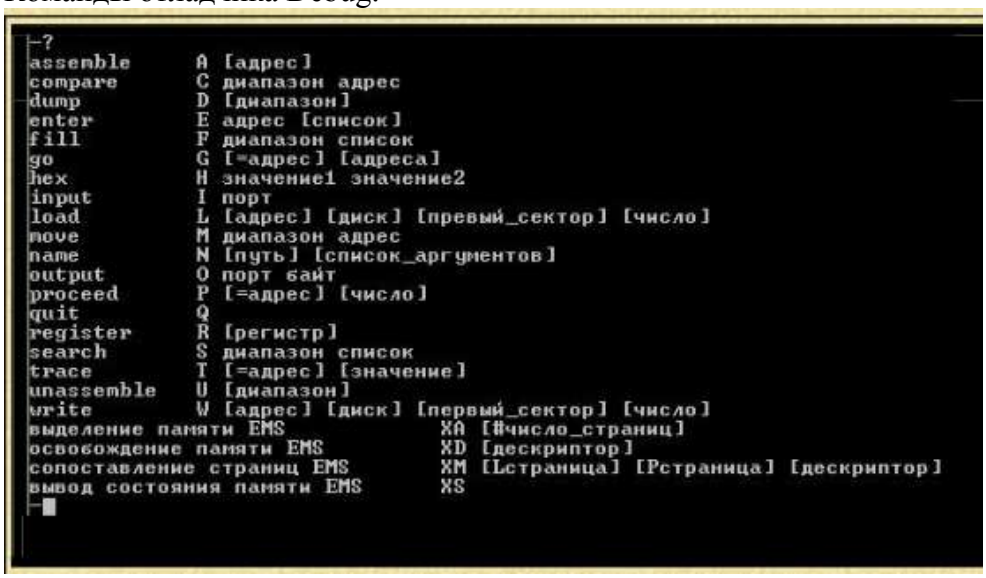
```
-G= 100
-D140 L40
-A118
-U100,11B
-E140 41 42
-E140 'ABCD'
-E140
-F140 L40 0
```

Ход работы:

Работа с файлами в DEBUG может начинаться с момента загрузки самого отладчика. Запуск отладчика DEBUG: Запуск отладчика можно осуществлять двумя способами:1-ый способ: Debug Запуск отладчика2-ой способ: Debugprim.com Запуск отладчика с последующей загрузкой файла prim.comc адреса смещения 0100(при этом, в пару регистров BX:CX заносится размер

загружаемого файла)Загрузка и запись файлов в отладчике DEBUG: Загрузка и запись файлов в отладчике осуществляется командами отладчика: L [адрес] -команда загрузки файла с указанного адреса смещения W[адрес] -команда записи файла с указанного адреса смещения Перед выполнением этих команд следует выполнить два действия:1. Командой Nзадать полное имя файла и путь к нему.2. В пару регистров BX:CX занести размер загружаемого / записываемого файла.

Команды отладчика Debug:



Пример:

сохранить диапазон адресов 100..11Bв файле PRIMER.COM.-NC:\TEMP\PRIMER.COM–ввод имени файла PRIMER.COM(расположен в папке TEMPдиска C)-R–просмотр содержимого регистров процессора (интерес представляет пара регистров BX:CX)-RCX–вход в режим изменения содержимого регистра CX(с целью занесения в регистр числа 1C-количества сохраняемых ячеек диапазона адресов 100..11B)-W100 –запись в файл PRIMER.COMсодержимого 1Cячеек памяти, начиная с адреса 100/

Порядок выполнения работы:

1. Составить математическую модель задачи.
2. Выбрать и обосновать наиболее рациональный метод решения задачи;
3. Разработать алгоритм для решения задачи.
4. Написать и отладить программу.

Форма представления результата:

1. Алгоритм программы.
2. 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 3.2. Директивы и операторы ассемблера

Практическое занятие № 22
Работа и использование отладчика AFDP: Команды передачи данных

Цель:

1. Научиться применять базовые понятия теории приближенных методов решения задач на ЭВМ.
2. Сформировать представление об алгоритмах решения прикладных задач, научится написанию и отладки программного кода.
3. Выбрать и обосновать наиболее рациональный метод и алгоритм решения задачи;
4. Разработать алгоритм для решения прикладных задач;
5. Написать программный код и отладить программу.

Выполнив работу, Вы будете:

уметь:

- применять стандартные алгоритмы в соответствующих областях;
- применять выбранные языки программирования для написания программного кода;
- использовать выбранную среду программирования;
- выявлять дефекты и отклонения в функционировании программного обеспечения компьютерных систем и комплексов.

Материальное обеспечение:

- Персональные компьютеры;
- Комплекты роботехнические "ПервоРобот NXT";
- Комплект робототехнический "LEGO";
- Контроллер 500995 ROBO TX;
- Набор аккумуляторный Accu Set;
- Наборы конструкторские ROBO TX;
- Датчик цвета для микрокомпьютера NXT;
- Программное обеспечение общего и профессионального назначения, в том числе включающее в себя следующее ПО:

MS Windows 7 (подписка Imagine Premium)
MS Office 2007
7 Zip
MS Visual Studio (подписка Imagine Premium)
Visual Studio Code
Calculate Linux Desktop

Задание:

Написать программу.

1. Вывод сообщения на экран с начала строки.
2. Вывод сообщения на экран с середины экрана.
3. Вывод сообщения на экран с начала строки и в рамке, построенной из любых символов псевдографики.
4. Вывод сообщения в рамке на середину экрана.
5. Вывод на экран символа с помощью функции 2h, для этого запишите в сегменте кодов:

	mov	ah, 2h	; функция вывода символа на экран
	mov	dl, 'A'	;ASCII
	int	21h	;прерывание DOS

6.Вывод сообщения на экран. Перед выдачей сообщения очистить экран функцией 6 int 10h:Вставить эти 8 команд после 9-й строки.

	mov	ah, 6h	; функция очистки экрана
	mov	al, 0	; 0 - весь экран
	mov	ch, 0	; номер строки левого верхнего угла
	mov	cl, 0	; номер столбца левого верхнего угла
	mov	dh, 24	; номер строки правого нижнего угла
	mov	dl, 79	; номер столбца правого нижнего угла
	mov	bh, 30h	; байт атрибут (на бирюзовом фоне черные символы)
	int	10h	; прерывание BIOS

7. Вывод сообщения на экран. Перед выдачей сообщения установить курсор функцией 2 int 10h:

	mov	ah, 2h	; функция установки курсора
	mov	bh, 0	; текущая видеостраница
	mov	dh, 5	; номер строки –5
	mov	dl, 10	; номер столбца -10
	int	10h	; прерывание BIOS

Вставить эти команды перед выдачей символа или сообщения.

8. Вывод сообщения на экран. Перед выдачей сообщения нарисовать цветное окно функцией 6 int 10h и установить курсор функцией 2 int 10h.

Ход работы:

При составлении программы на языке ассемблера можно выделить следующие этапы:

1) составление блок – схемы;

2) создание исходной программы NAME.ASM, где NAME – любое допустимое в С имя DOS файла;

3) создание объектной программы NAME.OBJ;

4) создание исполняемой программы NAME.EXE;

5) выполнение EXE – программы;

6) проверка результатов.

Порядок выполнения работы:

1. Составить математическую модель задачи.
2. Выбрать и обосновать наиболее рациональный метод решения задачи;
3. Разработать алгоритм для решения задачи.
4. Написать и отладить программу.

Форма представления результата:

3. Алгоритм программы.

4. 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 3.2. Директивы и операторы ассемблера

Практическое занятие № 23
Работа и использование отладчика AFDP: Арифметические команды

Цель:

1. Научиться применять базовые понятия теории приближенных методов решения задач на ЭВМ.
2. Сформировать представление об алгоритмах решения прикладных задач, научится написанию и отладки программного кода.
3. Выбрать и обосновать наиболее рациональный метод и алгоритм решения задачи;
4. Разработать алгоритм для решения прикладных задач;
5. Написать программный код и отладить программу.

Выполнив работу, Вы будете:

уметь:

- применять стандартные алгоритмы в соответствующих областях;
- применять выбранные языки программирования для написания программного кода;
- использовать выбранную среду программирования;
- выявлять дефекты и отклонения в функционировании программного обеспечения компьютерных систем и комплексов.

Материальное обеспечение:

- Персональные компьютеры;
- Комплекты роботехнические "ПервоРобот NXT";
- Комплект робототехнический "LEGO";
- Контроллер 500995 ROBO TX;
- Набор аккумуляторный Accu Set;
- Наборы конструкторские ROBO TX;
- Датчик цвета для микрокомпьютера NXT;
- Программное обеспечение общего и профессионального назначения, в том числе включающее в себя следующее ПО:

MS Windows 7 (подписка Imagine Premium)
MS Office 2007
7 Zip
MS Visual Studio (подписка Imagine Premium)
Visual Studio Code
Calculate Linux Desktop

Задание:

1.Написать программу для вычисления выражения:
(3456-2501+1099-794)-(384+101)

Ход работы:

Арифметические команды можно подразделить на пять подгрупп:
Команды сложения
Команды вычитания
Команды умножения
Команды деления
Команды расширения знака

- флаг не изменяется

+ флаг изменяется
? неопределенное значение

Мнемокод	Формат команды	Флаги								
		OF	DF	IF	TF	SF	ZF	AF	PF	CF
Команды сложения										
Add	Add приемник,источник	+	-	-	-	+	+	+	+	+
Adc	Adc приемник,источник	+	-	-	-	+	+	+	+	+
Inc	Inc приемник	+	-	-	-	+	+	+	+	-
Команды вычитания										
Sub	Sub приемник,источник	+	-	-	-	+	+	+	+	+
Sbb	Sbb приемник,источник	+	-	-	-	+	+	+	+	+
Dec	Dec приемник	+	-	-	-	+	+	+	+	-
Cmp	Cmp приемник,источник	+	-	-	-	+	+	+	+	+
Neg	Neg приемник	+	-	-	-	+	+	+	+	+
Команды умножения										
Mul	Mul источник	+	-	-	-	?	?	?	?	+
Imul	Imul источник	+	-	-	-	?	?	?	?	+
Команды деления										
Div	Div источник	?	-	-	-	?	?	?	?	?
Idiv	Idiv источник	?	-	-	-	?	?	?	?	?
Команды расширения знака										
Cbw	Cbw	-	-	-	-	-	-	-	-	-
Cwd	Cwd	-	-	-	-	-	-	-	-	-

2.Получите исполняемый модуль.

3.Проверьте правильность работы программы, запустив ее из отладчика и отслеживая результаты в нужных окнах.

Задание:

1.Написать программу для вычисления выражения:
(365/5+915)-(31*11-309)

Ход работы:

2.Получите исполняемый модуль.

3.Проверьте правильность работы программы, запустив ее из отладчика и отслеживая результаты в нужных окнах.

Порядок выполнения работы:

1. Составить математическую модель задачи.
2. Выбрать и обосновать наиболее рациональный метод решения задачи;
3. Разработать алгоритм для решения задачи.
4. Написать и отладить программу.

Форма представления результата:

1. Алгоритм программы.
2. 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 3.2. Директивы и операторы ассемблера

Практическое занятие № 24
Работа и использование отладчика AFDP: Логические операторы и команды сдвига. Команды передачи управления

Цель:

1. Научиться применять базовые понятия теории приближенных методов решения задач на ЭВМ.
2. Сформировать представление об алгоритмах решения прикладных задач, научится написанию и отладки программного кода.
3. Выбрать и обосновать наиболее рациональный метод и алгоритм решения задачи;
4. Разработать алгоритм для решения прикладных задач;
5. Написать программный код и отладить программу.

Выполнив работу, Вы будете:

уметь:

- применять стандартные алгоритмы в соответствующих областях;
- применять выбранные языки программирования для написания программного кода;
- использовать выбранную среду программирования;
- выявлять дефекты и отклонения в функционировании программного обеспечения компьютерных систем и комплексов.

Материальное обеспечение:

- Персональные компьютеры;
- Комплекты роботехнические "ПервоРобот NXT";
- Комплект робототехнический "LEGO";
- Контроллер 500995 ROBO TX;
- Набор аккумуляторный Accu Set;
- Наборы конструкторские ROBO TX;
- Датчик цвета для микрокомпьютера NXT;
- Программное обеспечение общего и профессионального назначения, в том числе включающее в себя следующее ПО:

MS Windows 7 (подписка Imagine Premium)
MS Office 2007
7 Zip
MS Visual Studio (подписка Imagine Premium)
Visual Studio Code
Calculate Linux Desktop

Задание 1:

Написать программу, которая решает следующую задачу, используя логические операции:

В регистрах R1, R2 и R3 записаны коды трех десятичных цифр, составляющих трехзначное число (соответственно сотни, десятки и единицы). Построить в регистре R0 это число.

Ход работы:

Например, если R1=3116, R2=3216 и R3=3316, в регистре R0 должно получиться десятичное число 123.

Определите и запишите в таблицу значения регистра R0 после выполнения каждой из следующих команд:

	Команда	R0
1	`MOV 1234, R0`	
2	`XOR ABCD, R0`	
3	`XOR ABCD, R0`	

Порядок выполнения работы:

1. Составить математическую модель задачи.
2. Выбрать и обосновать наиболее рациональный метод решения задачи;
3. Разработать алгоритм для решения задачи.
4. Написать и отладить программу.

Форма представления результата:

3. Алгоритм программы.
4. Код программы.

Задание 2:

Составить программу арифметических и логических действий над целыми переменными и константами.

Варианты заданий

1. $M=\begin{cases}(K+1)*4, & N<2\\ N-3*K+1/2, & 2\le N<4\\ 2*N+1, & N\ge 4\end{cases}$

2. $M=\begin{cases}4*I-7, & I>3\\ I*I+4*I-7, & I<1\\ (I*I*I)/(I*I+2), & 1\le I\le 3\end{cases}$

3. $M=\begin{cases}I+4*J, & I=J\\ 2*I-(J+J/2), & I>J\\ I*4+J, & I<J\end{cases}$

4. $M=\begin{cases}K*4, & N<7\\ (N-K)/2, & N>20\\ N+1, & 7\le N\le 20\end{cases}$

5. $M=\begin{cases}(J+I)*3, & I<-2\\ I*2+J*J/4, & I>1\\ 2*N+1, & -2\le I\le 1\end{cases}$

6. $M=\begin{cases}I*K+K/2, & I\ge 2\\ 2*I*I+K, & I<0\\ 3*I*I*I-K, & 0\le I\le 2\end{cases}$

7. $M=\begin{cases}2*I*I+I, & I*I<5\\ 5*I*I-1, & 5\le I*I\le 16\\ 3*I/5-2, & I*I\ge 16\end{cases}$

8. $M=\begin{cases}J+I*I/J, & I>J\\ J*(I+1), & I+J\\ 2*J-I, & I<J\end{cases}$

9. $M=\begin{cases}2*J*J+3, & J\ge 5\\ 7*J+8, & 2\le J<5\\ -2*J*J+2, & J<2\end{cases}$

10. $M=\begin{cases}3*I+J/2, & I\ge 7\\ 4*I-6, & 1\le I<7\\ 2*I*I*J, & I<1\end{cases}$

11. $M=\begin{cases}3*I-J+3, & (J+I)\ge 5\\ I*J, & 2\le(J+I)<5\\ J+2, & (J+I)<2\end{cases}$

12. $M=\begin{cases}K, & J>K\\ 5*J+K, & J<K\\ 7*K+J, & J=K\end{cases}$

Ход работы:

Для ввода исходных данных и вывода результата использовать буферы в виде символьных строк, в которых каждое значение переменной размещается в трех байтах:<знак числа> <старшая цифра> <младшая цифра>

Перевод чисел из символьной формы в числовую и обратно выполнить с помощью подпрограмм PSN и PNS соответственно.

Порядок выполнения работы:

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

4. Написать и отладить программу.

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 3.3. Java — язык программирования общего назначения.

Практическое занятие №25
Разработка консольного приложения для изучения типов данных и операторов. Документирование кода.

Цель:
Получить общее представление о создании программ на языке Java и познакомиться с его основными понятиями. Изучить синтаксические единицы, основные операторы и структуру кода программы. Освоить способы компиляции исходного кода и запуска программы.

Выполнив работу, Вы будете:
уметь:
- применять выбранные языки программирования для написания программного кода;
- использовать выбранную среду программирования;
- выявлять дефекты и отклонения в функционировании программного обеспечения компьютерных систем и комплексов.

Материальное обеспечение:
- Персональные компьютеры;
- Комплекты роботехнические "ПервоРобот NXT";
- Комплект робототехнический "LEGO";
- Контроллер 500995 ROBO TX;
- Набор аккумуляторный Accu Set;
- Наборы конструкторские ROBO TX;
- Датчик цвета для микрокомпьютера NXT;
- Программное обеспечение общего и профессионального назначения, в том числе включающее в себя следующее ПО:

MS Windows 7 (подписка Imagine Premium)
MS Office 2007
7 Zip
MS Visual Studio (подписка Imagine Premium)
Visual Studio Code
Calculate Linux Desktop

Теоретический материал

Язык Java ворвался в Интернет в конце 1995 года и немедленно завоевал популярность. Он обещал стать универсальным средством, обеспечивающим связь пользователей с любыми источниками информации, независимо от того, где она расположена – на Web-сервере, в базе данных, хранилище данных и т.д. Этот хорошо разработанный объектно-ориентированный язык программирования поддержали все производители программного обеспечения. Он имеет встроенные средства, позволяющие решать задачи повышенной сложности такие как: работа с сетевыми ресурсами, управление базами данных, динамическое наполнение web-страниц, многопоточность приложений.

Инсталляция набора инструментальных средств Java Software Development Kit. JDK долгое время был базовым средством разработки приложений. Он не содержит никаких текстовых редакторов, а оперирует только с уже существующими java-файлами. Компилятор представлен утилитой javac (java compiler), виртуальная машина реализована программой java. Для тестовых запусков аплетов есть специальная утилита appletviewer.

Пакет Java Software Development Kit можно загрузить с web-страницы: http://java.sun.com/javase/downloads/index.jsp. Способы инсталляции на разных платформах (Solaris, Windows, Linux) отличаются друг от друга.

После инсталляции пакета JSDK нужно добавить имя каталога jdk\bin в список путей, по которым операционная система может найти выполняемые файлы. Правильность установки пакета можно проверить, набрав команду java–version. На экране должно появиться, примерно, следующее:

java version "1.5.0_01"

Java(TM) 2 Runtime Environment, Standard Edition (build 1.5.0_01-b08) Java HotSpot(TM) Client VM (build 1.5.0_01-b08, mixed mode, sharing)

Среда разработки программ

Для написания программ на языке Java достаточно использовать самый простой текстовый редактор, однако применение специализированных средств разработки (Eclipse, Java WorkShop, Java Studio и др.) предоставляет большой набор полезных и удобных функций. Существует несколько способов компиляции и запуска на выполнение программ, написанных на языке Java: из командной строки или из другой программы, например, интегрированной среды разработки. Для компиляции программы из командной строки необходимо вызвать компилятор, набрав команду javac и указав через пробел имена компилируемых файлов:

javac file1.java file2.java file3.java

При успешном выполнении этапа компиляции в директории с исходными кодами появятся файлы с расширением .class, которые являются java байт-кодом. Виртуальная Java-машина (JVM) интерпретирует байт-код и выполняет программу. Для запуска программы необходимо в JVM загрузить основной класс, т.е. класс, который содержит функцию main(String s[]).

Например, если в файле file1.java есть функция main(), которая располагается в классе file1, то для запуска программы после этапа компиляции необходимо набрать следующее:

java file1

Компиляцию и запуск программ из интегрированных сред разработки необходимо осуществлять в соответствии с документацией на программный продукт.

Анализ программы

Технология Java, как платформа, изначально спроектированная для Глобальной сети Internet, должна быть многоязыковой, а значит, обычный набор символов ASCII (American Standard Code for Information Interchange, Американский стандартный код обмена информацией), включающий в себя лишь латинский алфавит, цифры и простейшие специальные знаки (скобки, знаки препинания, арифметические операции и т.д.), недостаточен. Поэтому для записи текста программы применяется более универсальная кодировка Unicode. Например, если в программу нужно вставить знак с кодом 6917, необходимо его представить в шестнадцатеричном формате (1B05) и записать: \u1B05.

Компилятор, анализируя программу, сразу разделяет ее на:

- пробелы (white spaces);
- комментарии (comments);
- основные лексемы (tokens).

Пробелами в данном случае называют все символы, разбивающие текст программы на лексемы. Это как сам символ пробела (space, \u0020, десятичный код 32), так и знаки табуляции и перевода строки. Они используются для разделения лексем, а также для оформления кода, чтобы его было легче читать. Например, следующую часть программы (вычисление корней квадратного уравнения):

```
double a = 1, b = 5,
c = 6; double D = b
* b - 4 * a * c;if (D
>= 0) {
double x1 = (-b + Math.sqrt (D)) /
(2 * a);double x2 = (-b - Math.sqrt
(D)) / (2 * a);
```

}
можно записать и в таком виде:
*double a=1,b=5,c=6;double D=b*b-4*a*c;if(D>=0)*
{double x1=(-
*b+Math.sqrt(D))/(2*a);doublex2=(-*
*b-Math.sqrt(D))/(2*a);}*

В обоих случаях компилятор сгенерирует абсолютно одинаковый код. Единственное соображение, которым должен руководствоваться разработчик, -

легкость чтения и дальнейшей поддержки такого кода.

Комментарии не влияют на результирующий бинарный код и используются только для ввода пояснений к программе. В Java комментарии бывают двух видов: строчные и блочные. Строчные комментарии начинаются с ASCII-символов // и длятся до конца текущей строки, например:

int y=1994; // год рождения

Блочные комментарии располагаются между ASCII-символами /* и */, могут занимать произвольное количество строк. Кроме этого, существует особый вид блочного комментария – комментарий разработчика (/**комментарии*/). Он применяется для автоматического создания документации кода [1].

Лексика языка

Лексика описывает, из чего состоит текст программы, каким образом он записывается и на какие простейшие слова (лексемы) компилятор разбивает программу при анализе. Лексемы (или tokens в английском варианте) – это основные "кирпичики", из которых строится любая программа на языке Java [1]. Ниже перечислены все виды лексем в Java:

- идентификаторы (identifiers);
- ключевые слова (key words);
- литералы (literals);
- разделители (separators);
- операторы (operators).

Идентификаторы – это имена, которые даются различным элементам языка для упрощения доступа к ним. Имена имеют пакеты, классы, интерфейсы, поля, методы, аргументы и локальные переменные. Длина имени не ограничена. Идентификатор состоит из букв и цифр. Имя не может начинаться с цифры.

Ключевые слова – специальные лексемы, зарегистрированные в системе для внутреннего использования, такие как abstract, default, if, private, this, boolean, implements, protected, static, try, void, native и др.

Литералы позволяют задать в программе значения для числовых, символьных и строковых выражений, а также null-литералов. В Java определены следующие виды литералов:

- целочисленный (integer);
- дробный (floating-point);
- булевский (boolean);
- символьный (character);
- строковый (string);
- null-литерал (null-literal).

Целочисленные (тип int занимает 4 байта, тип long – 8) литералы позволяют задавать целочисленные значения в десятеричном, восьмеричном и шестнадцатеричном виде. Запись нуля можно осуществить следующими способами:

- 0 (10-ричная система)
- 00 (8-ричная)
- 0x0 (16-ричная)

Если в конце литерала не стоит указателя на тип, то литерал по умолчанию имеет тип int.

Дробные литералы (тип float занимает 4 байта, тип double – 8) представляют собой числа с плавающей десятичной точкой. Дробный литерал состоит из следующих составных частей (по умолчанию имеет тип double):

- целая часть;
- десятичная точка (используется ASCII-символ точка);
- дробная часть;
- показатель степени (состоит из латинской ASCII-буквы «E» в произвольном регистре и целого числа с опциональным знаком «+» или «–»);
- окончание-указатель типа (D или F).

Символьные литералы. Представляют собой один символ и заключаются в одинарные кавычки ‘s’, ‘a’. Допускается запись через Unicode ‘\u0041’ – латинская буква “A”.

Строковые литералы состоят из набора символов и записываются в двойных кавычках: “символьный литерал”. Null литерал может принимать всего одно значение: null. Это литерал ссылочного типа, причем эта ссылка никуда не ссылается.

Разделители – специальные символы, используемые в конструкциях языка “()”, “[]”, “{}”, “;”, “,”, “.”.

Операторы используются в различных операциях – арифметических, логических, битовых, операциях сравнения и присваивания.

Пример простой программы “Hello, world!” выглядит следующим образом:

```
public class Test {
/**
*Основной метод, с которого начинается
*выполнение любой Java программы.
*/
public static void main(String args[])
{
System.out.println("Hello, world!");
}
}
```

Типы данных

Java является строго типизированным языком. Это означает, что любая переменная и любое выражение имеют известный тип еще на момент компиляции. Такое строгое правило позволяет выявлять многие ошибки уже во время компиляции. Компилятор, найдя ошибку, указывает точную строку и причину ее возникновения, а динамические «баги» необходимо сначала выявить тестированием, а затем найти место в коде, которое их породило.

Все типы данных разделяются на две группы. Первую составляют 8 простых или примитивных (от английского primitive) типов данных [1-3]. Они подразделяются на три подгруппы:

- целочисленные: byte, short, int, long, char;
- дробные: float, double;
- булевский: boolean.

Булевский тип представлен всего одним типом boolean, который может хранить всего два возможных значения – true и false. Величины именно этого типа получаются в результате операций сравнения.

Вторую группу составляют объектные или ссылочные (от английского reference) типы данных. Это все классы, интерфейсы и массивы.

Переменные

Переменные используются в программе для хранения данных. Любая переменная имеет три базовых характеристики: имя, тип, значение. Имя уникально идентифицирует переменную и позволяет к ней обращаться в программе. Тип описывает, какие величины может хранить переменная. Значение – текущая величина, хранящаяся в переменной на данный момент. Значение может быть указано сразу (инициализация), а в большинстве случаев задание начальной величины можно и отложить.

Int a;
int b=0,
c=2+3;
double
d=e=5.5;

Ниже приведены данные по всем целым и дробным типам:

Название типа	**Длина (байт)**	**Область значений**
Целые типы		
byte	1	-128..127
short	2	-32768..32767
int	4	-2147483648..2147483647
long	8	-9223372036854775808..9223372036854775807
char	2	0..65535
Дробные типы		
float	4	3.40282347e+38f; 1.40239846e-45f
double	8	1.79769313486231570e+308; 4.94065645841246544e-324

```
/* # 1 # Пример. Типы данных, литералы и операции над ними */
public class Primer1 {
   public static void main(String[]
      args) {byte b = 1, b1 = 1 +
      2;
      final byte B = 1 + 2;
      // b = b1 + 1; // ошибка приведения типов int в byte
      /*
       * переменная b1 на момент выполнения кода b = b1 + 1; может измениться, и
       * выражение b1 + 1 может превысить допустимый размер byte- типа
       */
      b = (byte) (b1 + 1);
      b = B + 1; // работает
      /*
       * B - константа, ее значение определено, компилятор вычисляет значение
       * выражения B + 1, и если его размер не превышает допустимого для byte типа, то
       * ошибка не возникает
       */
      // b = -b; // ошибка приведения типов
      b = (byte) -b;
      // b = +b; // ошибка приведения типов
      b = (byte) +b;
```

```
i               // b = i; // ошибка приведения типов, int больше, чем byte
n               b = (byte) i;
t               final int D = 3;
                b = D; // работает
i               /*
                 * D –константа. Компилятор проверяет, не превышает ли ее значение
=

3
;
допустимый
                 * размер для типа byte, если не превышает, то ошибка не возникает
                 */
                final int D2 = 129;
                // b=D2; // ошибка приведения типов, т.к. 129 больше, чем допустимое 127
                b = (byte) D2;
                b += i++; // работает
                b += 1000; // работает
                b1 *= 2; // работает
```

```
        float f = 1.1f;
        b /= f; // работает
        /*
         *  все сокращенные операторы автоматически преобразуют результат выражения к
         * типу переменной, которой присваивается это значение. Например, b /= f;
         * равносильно b = (byte)(b / f);
         */
    }
}
```

Документирование кода

В языке Java используются блочные и однострочные комментарии /* */ и //, аналогичные комментариям, применяемым в C++. Введен также новый вид комментария

Документирование кода
В языке Java используются блочные и однострочные комментарии /* */ и //, аналогичные комментариям, применяемым в C++. Введен также новый вид комментария /** */, который может содержать описание документа с помощью дескрипторов вида:

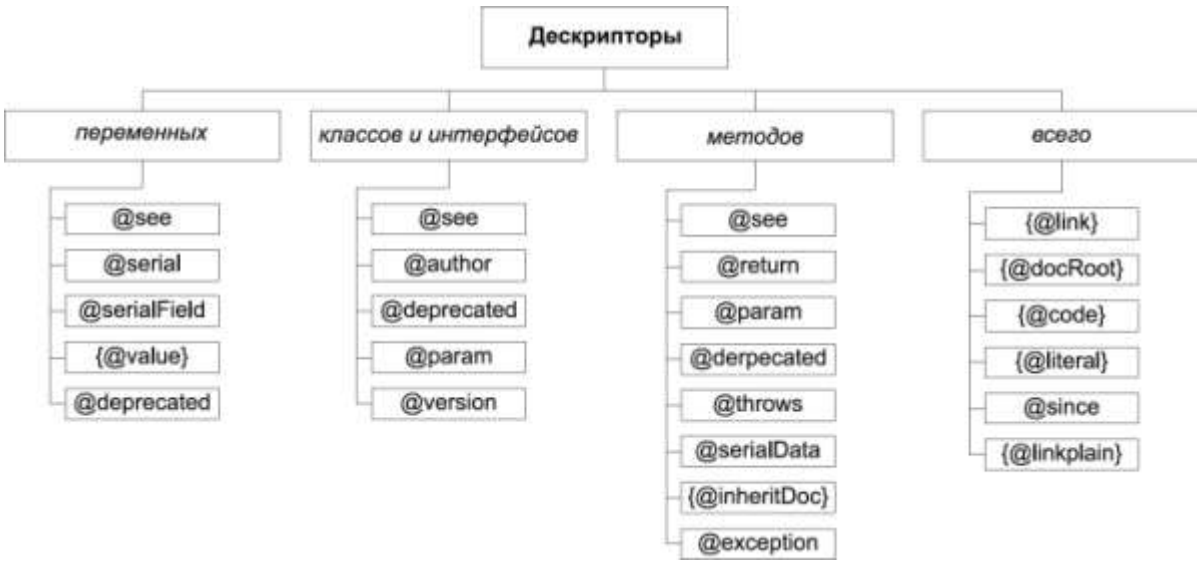


Рисунок 1.1. Дескрипторы документирования кода

@author – задает сведения об авторе;
@version – задает номер версии класса;
@exception – задает имя класса исключения;
@param – описывает параметры, передаваемые методу;
@return – описывает тип, возвращаемый методом;
@deprecated – указывает, что метод устаревший и у него есть более совершенный аналог;
@since – определяет версию, с которой метод (член класса, класс) присутствует;
@throws – описывает исключение, генерируемое методом;
@see – что следует посмотреть дополнительно.

Из *java*-файла, содержащего такие комментарии, соответствующая утилита javadoc.exe может извлекать информацию для документирования классов и сохранения ее в виде *html*-документа. В качестве примера и образца можно рассматривать исходный код языка *Java* и документацию, сгенерированную на его основе.

/* # 2 # фрагмент класса Object с дескрипторами документирования # Object.java */
package java.lang;

```
/**
* Class {@code Object} is the root of the class hierarchy.
* Every class has {@code Object} as a superclass. All objects,
* including arrays, implement the methods of this class.
*
* @author unascribed
* @see java.lang.Class
* @since JDK1.0
*/
public class Object {
/**
* Indicates whether some other object is "equal to" this one.
* <p>
* MORE COMMENTS HERE
* @param obj the reference object with which to compare.
* @return {@code true} if this object is the same as the obj
* argument; {@code false} otherwise.
* @see #hashCode()
* @see java.util.HashMap
*/
public boolean equals(Object obj)
{return (this == obj);
}
/**
* Creates and returns a copy of this object.
* MORE COMMENTS HERE
* @return a clone of this instance.
* @exception CloneNotSupportedException if the object's class does not
* support the {@code Cloneable} interface. Subclasses
* that override the {@code clone} method can also
* throw this exception to indicate that an instance cannot
* be cloned.
* @see java.lang.Cloneable
*/
protected native Object clone() throws CloneNotSupportedException;
// more code here
}
```

Всегда следует помнить, что точные названия классов, их полей и методов улучшают восприятие кода и уменьшают размер комментариев. Наличие комментария должно еще больше облегчить скорость восприятия разработанного кода.

Код системы будет читаться чаще и больше по времени, чем требуется на его создание. Комментарии помогут программисту, сопровождающему код, быстрее разобраться в нем и грамотнее использовать или изменять его.

Классы–оболочки

Кроме базовых типов данных, в языке Java широко используются соответствующие классы-оболочки (wrapper-классы) из пакета java.lang: Boolean,Character, Integer, Byte, Short, Long, Float, Double. Объекты этих классов могут хранить те же значения, что и соответствующие им базовые типы.

Объект любого из этих классов представляет собой полноценный экземпляр в динамической памяти, в котором хранится его неизменяемое значение.

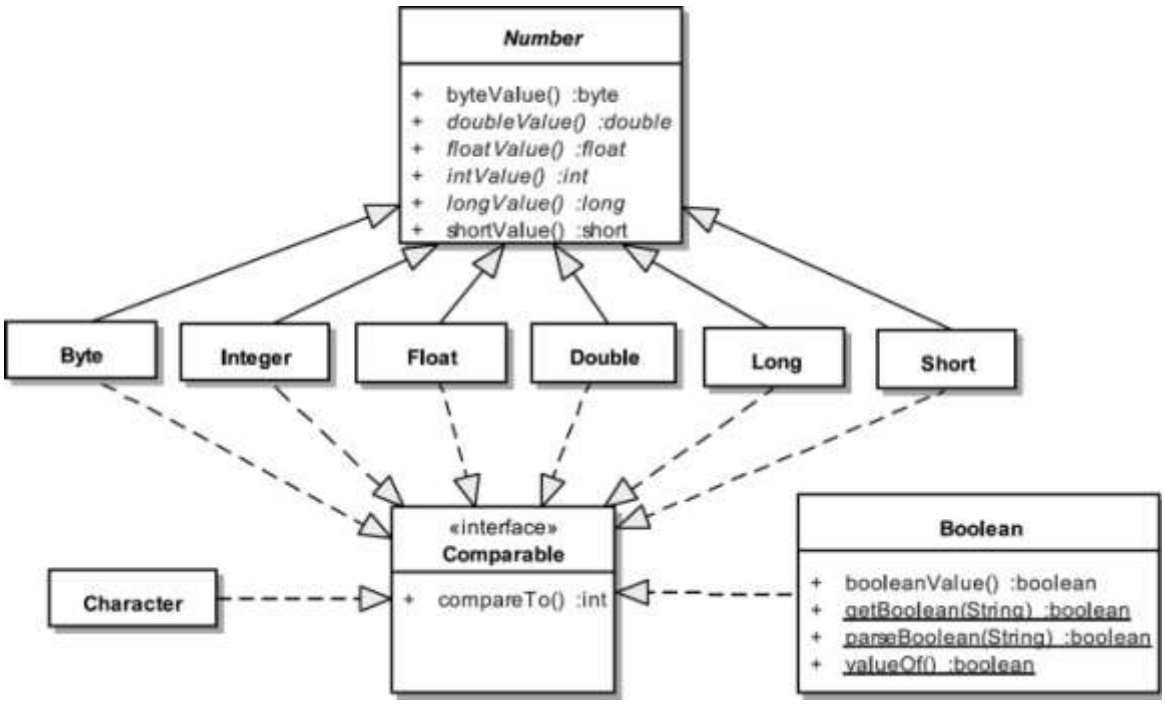


Рисунок 1.2 Иерархия классов-оболочек

Значения базовых типов хранятся в стеке и не являются объектами. Классы, соответствующие числовым базовым типам, находятся в библиотеке java.lang, являются наследниками абстрактного класса Number и реализуют интерфейс Comparable<T>. Этот интерфейс определяет возможность сравнения объектов одного типа между собой с помощью метода int compareTo(T ob). Объекты классов-оболочек по умолчанию получают значение null.

Создаются экземпляры интегральных или числовых классов с помощью одного из двух конструкторов с параметрами типа String и соответствующего базового типа.

Объект класса-оболочки может быть преобразован к базовому типу методом типValue() или обычным присваиванием.

Класс Character не является подклассом Number, этому классу нет необходимости поддерживать интерфейс классов, предназначенных для хранения результатов арифметических операций. Вместо этого класс Character имеет целый ряд специфических методов для обработки символьной информации.

У класса Character, в отличие от других классов оболочек, не существует конструктора с параметром типа String.

Float ft = new Float(1.7); // double в Float Short s = new Short((short)5); // int в Short Short sh = new Short("5"); // String в Short double d = s.doubleValue(); // Short в double byte b = (byte)(float)ft; // Float в byte Character ch = new Character('3');

int i = Character.digit(ch.charValue(), 10); /* Character в int */

Конструкторы классов-оболочек с параметром типа String и их методы valueOf(String str), decode(String str) и parseТип(String str) выполняют действия по преобразованию значения, заданного в виде строки, к значению соответствующего объектного типа данных. Исключение составляет класс Character. При преобразовании строки к конкретному типу может возникнуть ошибка формата данных, если строка не соответствует этому типу данных. Для устойчивой работы приложения все операции по преобразованию строки в типизированные значения желательно заключать в блок try-catch для перехвата и обработки возможного исключения.

Четыре стандартных способа преобразования строки в число:

```
/* # 3 # преобразование строки в целое число # StringToInt.java */ package by.bsac.transformation;
public class StringToInt {
public static void main(String[ ] args) {
String arg = "71"; // 071 или 0x71или 0b1000111 try {
int value1 = Integer.parseInt(arg); // возвращает int
int value2 = Integer.valueOf(arg); // возвращает Integer int value3 = Integer.decode(arg); // возвращает Integer int value4 = new Integer(arg); /* создает Integer,
для преобразования применяется редко */
} catch (NumberFormatException e) { System.err.println("Неверный формат числа " + e);
}
}
}
```

У приведенных способов есть определенные различия при использовании разных систем счисления и представления чисел.

Обратное преобразование из типизированного значения (в частности int) в строку можно выполнить следующими способами:

```
int value = 71;
String arg1 = Integer.toString(value); // хороший способ String arg2 = String.valueOf(value); // хороший способ String arg3 = "" + value; // плохой способ
```

Существует два класса для работы с высокоточной арифметикой – java.math.BigInteger и java.math.BigDecimal, которые поддерживают целые числа и числа с фиксированной точкой произвольной длины.

Начиная с версии 5.0, введен процесс автоматической инкапсуляции данных базовых типов в соответствующие объекты оболочки и обратно (автоупаковка/авторапаковка). При этом нет необходимости в явном создании соответствующего объекта с использованием оператора new:

```
Integer iob = 71; // эквивалентно Integer iob = new Integer(71);
```

При инициализации объекта класса-оболочки значением базового типа преобразование типов в некоторых ситуациях необходимо указывать явно, то есть код

```
Float f = 7; // правильно будет (float)7 или 7F вместо 7
```

вызывает ошибку компиляции. С другой стороны, справедливо:

```
Float f = new Float("7");
```

Авторапаковка – процесс извлечения из объекта-оболочки значения базового типа. Вызовы методов intValue(), doubleValue() и им подобных для преобразования объектов в значения базовых типов становятся излишними.

Допускается участие объектов в арифметических операциях, однако не следует этим злоупотреблять, поскольку упаковка/распаковка является ресурсоемким процессом:

```
// autoboxing & unboxing:
Integer i = 71; // создание объекта+упаковка

++i; // распаковка+операция+создание объекта+упаковка
int j = i; // распаковка
```

Однако следующий код генерирует исключительную ситуацию NullPointerException при попытке присвоить базовому типу значение null объекта класса Integer, литерал null – не объект и не может быть преобразован к значению «ноль»:

```
Integer j = null; // объект не создан! Это не ноль!
int i = j; // генерация исключения во время выполнения
```

Несмотря на то, что значения базовых типов могут быть присвоены объектам классов-оболочек, сравнение объектов между собой происходит по ссылкам. Для сравнения значений объектов следует использовать метод equals().

```
int i = 127;
Integer a = i; // создание объекта+упаковка
```

```
Integer b = i;
System.out.println("a==i " + (a == i)); // true – распаковка и сравнение значений
System.out.println("b==i " + (b == i)); // true
System.out.println("a==b " + (a == b)); /* false(ссылки на разные объекты) */
System.out.println("equals ->" + a.equals(i)
+ b.equals(i)
+ a.equals(b)); // true, true, true
```

Метод equals() сравнивает не значения объектных ссылок, а значения объектов, на которые установлены эти ссылки. Поэтому вызов a.equals(b) возвращает значение true.

Значение базового типа может быть передано в метод equals(). Однако ссылка на базовый тип не может вызывать методы:

```
i.equals(a); // ошибка компиляции
```

Стало возможным создавать объекты и массивы, сохраняющие различные базовые типы без взаимных преобразований, с помощью ссылки на класс Number, а именно:

```
Number n1 = 1; // идентично new Integer(1) Number n2 = 7.1; // идентично new Double(7.1)
```

Практическое применение таких объектов крайне ограничено.

При автоупаковке значения базового типа возможны ситуации с появлением некорректных значений и непроверяемых ошибок.

Переменная базового типа всегда передается в метод по значению, а переменная класса-оболочки – по ссылке.

Операторы управления

В языке Java, как и в любом другом языке программирования, есть условные операторы и циклы для управления потоком. Блок, или составной оператор, произвольное количество простых операторов языка Java, заключенных в фигурные скобки. Блоки определяют область видимости своих переменных. Блоки могут быть вложенными один в другой. Однако невозможно объявить одинаково названные переменные в двух вложенных блоках.

```
Public static void main(String [] args)
{ int n;
…
{
int k;
int n; // Ошибка – невозможно переопределить переменную n во вложенном цикле
} //переменная k определена только в этом блоке
}
```

Условные операторы

Условный оператор в языке Java имеет вид:

```
if (условие) оператор
// или
if (условие) { оператор1; оператор2; }
```

Все операторы, заключенные в фигурные скобки, будут выполнены, если значение условия истинно. Общий случай условного оператора выглядит так:

```
if (условие) оператор1 else оператор2 if (yourSale >= target)
{ performance=”Удовлетворительно”; Bonus = 100 + 0.01*( yourSale – target);
}
else
{ performance=”Неудовлетворительно”; Bonus =0;
}
```

Многовариантное ветвление представлено в виде повторяющихся операторов

```
if … else if…
```

```
if (sale >=2*target)
{ performance=”Отлично”;
}
else if (sale >=1.5*target)
{ performance=”Удовлетворительно”;
}
else {System.out.println(“Вы уволены”);}
```

Многовариантное ветвление – оператор switch

Конструкция if/else может оказаться неудобной, если необходимо сделать выбор из многих вариантов. Например, создавая систему меню из трех альтернатив, можно использовать следующий код.

```
String input = “1”;
int choice = Integer.parseInt(input); switch (choice){
case 1:
… break; case 2:
… break; case 3:
… break;
default: // неверный выбор
… break; }
```

Выполнение начинается с метки case, соответствующей значению переменной choice, и продолжается до следующего оператора break или конца оператора switch. Если ни одна метка не совпадает со значением переменной, выполняется раздел default. Метка case должна быть целочисленной!

Неопределенные циклы

Существует два вида повторяющихся циклов, которые лучше всего подходят, если вы точно не знаете, сколько повторений должно быть выполнено. Первый из них, цикл while, выполняет тело цикла, только пока выполняется его условие.

```
while (условие) {операторы;}
```

Условие цикла while проверяется в самом начале. Следовательно, возможна ситуация, когда код, содержащийся в блоке, не будет выполнен ни разу. Если необходимо, чтобы блок выполнялся хотя бы один раз, проверку условия нужно перенести в конец. Это можно сделать с помощью цикла

do/while.

```
do оператор while (условие);
```

Определенные циклы

Цикл for – распространенная конструкция для выполнения повторений, количество которых контролируется счетчиком, обновляемым на каждой итерации.

```
for (int i = 1; i <= 10; i++){ System.out.println(i);
}
```

Первый элемент оператора for обычно выполняет инициализацию счетчика, второй формулирует условие выполнения тела цикла, а третий определяет способ обновления счетчика.

```
for (int i = 10; i > 0; - i){ System.out.println(“Обратный отсчет …” + i);
}
```

Цикл полного перебора используется для перебора элементов массива или коллекции:

```
for (ТипДанных имя : имяОбъекта) { /* операторы */ }
```

Оператор продолжения continue

Позволяет начать новую итерацию цикла не доходя до конца текущей итерации. По умолчанию итерация относится к телу цикла, в котором вызывается оператор. Также можно указать метку блока, с которой начать новую итерацию.

continue;

Оператор выхода из блока break

Позволяет выйти из текущего блока в операторе выбора или из тела цикла. Если указана метка блока, то выходит из того блока.

break;

Пакеты

Программа на Java представляет собой набор пакетов (packages).

Каждый пакет может включать вложенные пакеты, а так же может содержать классы и интерфейсы. Каждый пакет имеет свое пространство имен, что позволяет создавать одноименные классы в различных пакетах.

Имена бывают простыми (simple), состоящими из одного идентификатора, и составными (qualified), состоящими из последовательности идентификаторов, разделенных точкой. Составное имя любого элемента пакета составляется из составного имени этого пакета и простого имени элемента.

Простейшим способом организации пакетов и типов является обычная файловая структура. Например, исходный код класса space.sunsystem.Moon хранится в файле space\sunsystem\Moon.java

Запуск программы на JAVA стоит производить из директории, в которой содержатся пакеты. Было бы ошибкой запускать Java прямо из папки space\sunsystem и пытаться обращаться к классу Moon, несмотря на то, что файл-описание лежит именно в ней. Необходимо подняться на два уровня директорий выше, чтобы Java, построив путь из имени пакета, смогла обнаружить нужный файл.

Модуль компиляции

Модуль компиляции (complication unit)-хранится в текстовом .java-файле и является единичной порцией входных данных для компилятора. Состоит из трех частей:

–Объявление пакета;

–Import-выражения;

–Объявления верхнего уровня;

Объявление пакета указывает, какому пакету будут принадлежать все объявляемые ниже типы. Используется ключевое слово package, после которого указывается полное имя пакета. Например, в файле java/lang/Object.java идет: package java.lang; что служит одновременно объявлением пакета lang, вложенного в пакет java, и указанием, что объявляемый ниже класс Object, находится в этом пакете. Так складывается полное имя класса java.lang.Object.

Область видимости типа – пакет, в котором он располагается. Внутри этого пакета допускается обращение к типу по его простому имени. Из всех других пакетов необходимо обращаться по составному имени.

Для решения этой проблемы вводятся import-выражения, позволяющие импортировать типы в модуль компиляции и далее обращаться к ним по простым именам. Существует два вида таких выражений:

- импорт одного типа: import java.net.URL;
- импорт пакета: import java.awt.*;

2. Порядок выполнения работы

- изучить теоретический материал;
- напишите программы на языке Java:

1. *Программа, в которой перебираются числа от 1 до 500 и выводятся на экран. Если число делится на 5, то вместо него выводится слово fizz, если на 7, то buzz. Если число*

делится на 5 и на 7, то выводить слово fizzbuzz. Примечание: остаток от деления в Java обозначается через символ %.*

2. *Программа, в которой все переданные во входную строку аргументы выводятся на экран в обратной порядке. Например, если было передано 2 аргумента – make install, то на экран должно вывестись llatsni ekam. Примечание*: для разбора слова по буквам необходимо использовать функцию charAt(). Например, str.charAt(i) вернет символ с позиции i в слове, записанном в строковую переменную str. Команда str.length() возвращает длину слова str.*
3. *Создайте программу, вычисляющую числа Фибоначчи. Числа Фибоначчи – последовательность чисел, в котором каждое следующее число равно сумме двух предыдущих. Начало этой последовательности – числа 1, 1, 2, 3, 5, 8, 13…*
4. *Создайте программу, вычисляющую факториал целого числа.*

- *выполните индивидуальные задания.*

5. *Индивидуальные задания*

1. *Ввести с консоли 3 целых числа. На консоль вывести: Четные и нечетные числа.*
2. *Ввести с консоли 3 целых числа. На консоль вывести: Наибольшее число.*
3. *Ввести с консоли 3 целых числа. На консоль вывести: Числа, которые делятся на 3 или на 9.*
4. *Ввести с консоли 3 целых числа. На консоль вывести: Числа, которые делятся на 5 и на 7.*
5. *Ввести с консоли 3 целых числа. На консоль вывести: Определить среднее значение наибольшего и наименьшего числа.*
7. *Ввести с консоли 3 целых числа. На консоль вывести: Наибольший общий делитель этих чисел.*
8. *Ввести с консоли 3 целых числа. На консоль вывести: Простые числа.*
9. *Ввести с консоли 3 целых числа. На консоль вывести: Отсортированные числа в порядке возрастания и убывания.*
10. *Ввести с консоли 3 целых числа. На консоль вывести: Наименьшее общее кратное этих чисел.*
11. *Ввести с консоли 3 целых числа. На консоль вывести: Наименьшее число.*
12. *Ввести с консоли 3 целых числа. На консоль вывести: Числа, входящие в заданный промежуток.*
13. *Ввести с консоли 3 целых числа. На консоль вывести: Найти количество положительных чисел.*
14. *Ввести с консоли 3 целых числа. На консоль вывести: Найти среднее из них (то есть число, расположенное между наименьшим и наибольшим).*
15. *Ввести с консоли 3 целых числа. На консоль вывести: Если их значения упорядочены по возрастанию, то удвоить их; в противном случае заменить значение каждой переменной на противоположное. Вывести новые значения переменных.*
16. *: Даны координаты точки, не лежащей на координатных осях OX и OY. Определить номер координатной четверти, в которой находится данная точка.*
17. *Даны три целых числа, одно из которых отлично от двух других, равных между собой. Определить порядковый номер числа, отличного от остальных.*
18. *Ввести с консоли 3 целых числа. На консоль вывести: Найти сумму двух наибольших из них.*
19. *Дано целое число. Вывести его строку-описание вида «отрицательное четное число», «нулевое число», «положительное нечетное число».*
20. *Даны три числа. Найти сумму двух наименьших из них.*
21. *Даны четыре целых числа, одно из которых отлично от трех других, равных между собой. Определить порядковый номер числа, отличного от остальных.*

22. *Ввести с консоли 3 целых числа. На консоль вывести: Определить среднее значение наибольшего и среднего числа.*
23. *Ввести с консоли 3 целых числа. На консоль вывести: Определить среднее значение среднего и наименьшего числа.*
24. *Дано целое число. Если оно является отрицательным, то прибавить к нему 1; в противном случае не изменять его. Вывести полученное число.*
25. *Дано целое число. Если оно равно 0, то прибавить к нему 1; в противном случае не изменять его. Вывести полученное число.*
26. *Дано целое число. Если оно является положительным, то прибавить к нему 1; в противном случае не изменять его. Вывести полученное число.*
27. *Даны две переменные целого типа: A и B. Если их значения не равны, то присвоить каждой переменной большее из этих значений, а если равны, то присвоить переменным нулевые значения. Вывести новые значения переменных A и B.*
28. *Даны две переменные вещественного типа: A, B. Перераспределить значения данных переменных так, чтобы в A оказалось меньшее из значений, а в B – большее. Вывести новые значения переменных A и B.*
29. *Дано целое число. Если оно является положительным, то прибавить к нему 1; в противном случае не изменять его. Вывести полученное число.*
30. *Пользователь вводит порядковый номер пальца руки. Необходимо вывести его название на экран.*

3. Контрольные вопросы

1. В чем разница JDK и JRE?
2. Назовите типы данных Java и классы-оболочки. Чем они отличаются?
3. Для чего применяют документирование кода с помощью дескрипторов? Перечислите основные дескрипторы.
4. Перечи

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.