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
по выполнению практических работ

ОП.09 ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ

«Общепрофессиональные дисциплины»
программы подготовки специалистов среднего звена
специальности 09.02.01 Компьютерные системы и комплексы
(базовой подготовки)

Магнитогорск, 2019

ОДОБРЕНО:

Предметно-цикловой комиссией
«Информатики и вычислительной техники»
Председатель *И.Г.Зорина*
Протокол № 6 от 20 февраля.2019г.

Методической комиссией МпК
Протокол №5 от «21» февраля 2019г

Разработчик (и):
преподаватель МпК
ФГБОУ ВПО «МГТУ» Людмила Александровна Фетисова

Методические указания по выполнению практических работ разработаны на основе рабочей программы учебной дисциплины «ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ».

Содержание практических работ ориентировано на подготовку студентов к освоению профессионального модуля основной программы подготовки специалистов среднего звена 09.02.01 Компьютерные системы и комплексы базового уровня и овладению профессиональными компетенциями.

Содержание

1 ВВЕДЕНИЕ

Важную часть теоретической и профессиональной практической подготовки студентов составляют практические занятия. Являясь частью изучения учебной дисциплины, они призваны, экспериментально подтвердить теоретические положения и формировать общие и профессиональные компетенции, практические умения.

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

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

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

В результате их выполнения, обучающийся должен:

уметь:

- формализовать поставленную задачу;
- применять полученные знания к различным предметным областям;
- составлять и оформлять программы на языках программирования;
- тестировать и отлаживать программы;

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

ПК 2.1. Создавать программы на языке ассемблера для микропроцессорных систем.

ПК 2.2. Производить тестирование и отладку микропроцессорных систем.

ПК 3.3. Принимать участие в отладке и технических испытаниях компьютерных систем и комплексов: инсталляции, конфигурировании и настройке операционной системы, драйверов, резидентских программ.

А также формированию общих компетенций:

ОК 1. Понимать сущность и социальную значимость своей будущей профессии, проявлять к ней устойчивый интерес.

ОК 2. Организовывать собственную деятельность, определять методы и способы выполнения профессиональных задач, оценивать их эффективность и качество.

ОК 3. Решать проблемы, оценивать риски и принимать решения в нестандартных ситуациях.

ОК 4. Осуществлять поиск, анализ и оценку информации, необходимой для постановки и решения профессиональных задач, профессионального и личностного развития.

ОК 5. Использовать информационно-коммуникационные технологии в профессиональной деятельности.

ОК 6. Работать в коллективе и команде, обеспечивать ее сплочение, эффективно общаться с коллегами, руководством, потребителями.

ОК 7. Ставить цели, мотивировать деятельность подчиненных, организовывать и контролировать их работу с принятием на себя ответственности за результат выполнения заданий.

ОК 8. Самостоятельно определять задачи профессионального и личностного развития, заниматься самообразованием, осознанно планировать повышение

квалификации.

ОК 9. Быть готовым к смене технологий в профессиональной деятельности.

Выполнение студентами 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.

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

2 МЕТОДИЧЕСКИЕ УКАЗАНИЯ
ПЕРЕЧЕНЬ ПРАКТИЧЕСКИХ ЗАНЯТИЙ

Разделы/темы	Темы практических/практических занятий	Количество часов
Раздел 1. Принципы машинной обработки данных		8
Тема 1.1. Основные понятия алгоритмизации. Основные алгоритмические конструкции.	Практическая работа № 1,2,3,4 Построение блок схем основных алгоритмических конструкций.	8
Раздел 2. Структурное программирование		32
Тема 2.2. Ввод и вывод данных	Практическая работа № 5,6 Операции ввода - вывода	4
Тема 2.3. Базовые конструкции языков программирования	Практические работы №7,8 Оператор условия Практическая работа №9 Оператор цикла с предусловием Практическая работа № 10 Оператор цикла с постусловием Практическая работа № 11 Оператор цикла с параметром	4 2 2 2
Тема 2.4. Массивы	Практическая работа № 12 Работа со строками. Практическая работа № 13,14,15 Алгоритмы поиска, сортировки и замены	2 6
Тема 2.5. Функции	Практическая работа № 16 Параметры функции Практическая работа № 17 Рекурсивные функции Практическая работа № 18 Многофайловые проекты	2 2 2
Тема 2.6. Функции	Практическая работа №19 Работа с указателями. Практическая работа №20 Динамическое распределение памяти.	2 2
Раздел 3 Основы программирования на Ассемблере		10
Тема 3.2. Директивы и операторы ассемблера	Практическая работа №21 Работа и использование отладчика AFDP: Основные команды Практическая работа №22 Работа и использование отладчика AFDP: Команды передачи данных Практическая работа №23 Работа и использование отладчика AFDP: Арифметические команды Практическая работа №24 Работа и использование отладчика AFDP:	2 2 2 2

	Логические операторы и команды сдвига Практическая работа №25 Работа и использование отладчика AFDP: Команды передачи управления	2
ИТОГО		**50**

Цель работы:

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.

Количество часов: 50

Материальное обеспечение:

- Автоматизированные рабочие места на 12-15 обучающихся (процессор не ниже Core i3, оперативная память объемом не менее 8 Гб) или аналоги;
- Автоматизированное рабочее место преподавателя (процессор не ниже Core i3, оперативная память объемом не менее 8 Гб) или аналоги;
- Сервер в лаборатории (8-х ядерный процессор с частотой не менее 3 ГГц, оперативная память объемом не менее 16 Гб, жесткие диски общим объемом не менее 1 Тб, программное обеспечение: WindowsServer 2012 или более новая версия) или выделение аналогичного по характеристикам виртуального сервера из общей фермы серверов
- Проектор и экран;
- Маркерная доска;
- Программное обеспечение общего и профессионального назначения, в том числе включающее в себя следующее ПО:
  EclipseIDEforJavaEEDevelopers,
  .NETFrameworkJDK 8,
  MicrosoftSQLServerExpressEdition,
  MicrosoftVisioProfessional,
  MicrosoftVisualStudio,
  MySQLInstallerforWindows,
  NetBeans,
  SQLServerManagementStudio,
  MicrosoftSQLServerJavaConnector,
  AndroidStudio,
  IntelliJIDEA.

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 предоставления результата

1. Алгоритм программы.
2. Код программы.

Практическая работа № 1,2,3,4 Построение блок схем основных алгоритмических конструкций.

Алгоритмы классифицируются по форме представления (рис. 2.1).

Рис. 2.1. Классификация алгоритмов по форме представления

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

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 $ax^2 + bx + c = 0$.

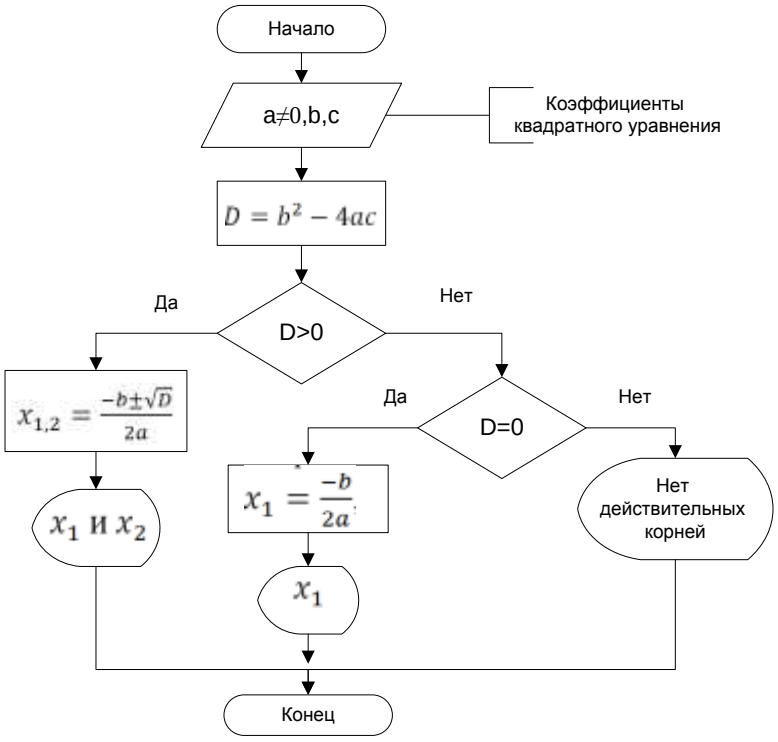


Рис.2.2. Блок-схема алгоритма решения квадратного уравнения

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 C++ (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.

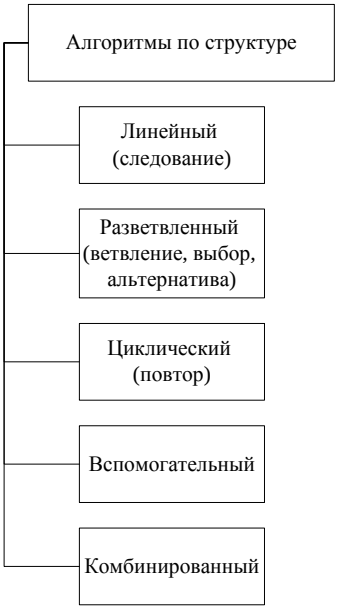


Рис.2.3. Классификация алгоритмов по структуре

Практическая работа № 5,6 Операции ввода - вывода
Задача

1. Какие операции определены над переменными булевого типа?
2. Записать на языке С++ следующие выражения:

   а. $\frac{x^2 + 3x - y}{a\sin x + e^y}$ б. $\frac{ab^{-2}}{2c}$
3. Организовать ввод и вывод данных заданных типов, снабдив распечатки соответствующими заголовками

   x=’*’ y=’/ ’ z=0,75 J=0.0 π

Дать протокол программы.

Задача

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

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

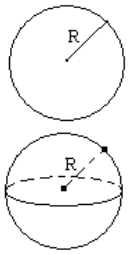
Дать протокол программы.

Задача

1. Формулировка задачи

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

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 $\pi \approx 3{,}14159265358979323846264338\ldots$, 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).

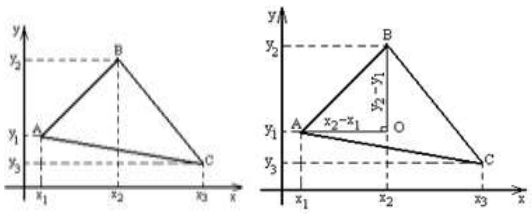
4. Блок-схема алгоритма

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*.



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

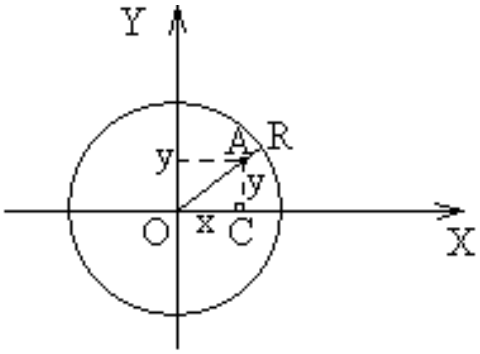
5. . Код программы.

Практические работы №7,8 Оператор условия

Задача

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

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».

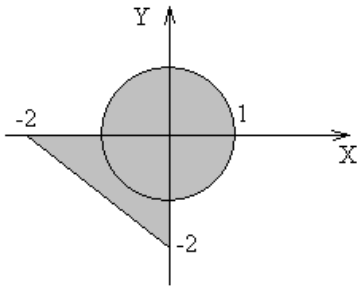
Так как радиус окружности и координаты точки могут принимать любые значения (2; 2,5; 3,75), все переменные для данной задачи определены как действительные числа.

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

Задача

1. Формулировка задачи

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



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 $\begin{cases} x \le 0 \\ y \le 0 \\ y \ge -x - 2 \end{cases}$.

Следовательно, заштрихованная область определена следующими условиями $x^2 + y^2 \le 1^2$ или $\begin{cases} x \le 0 \\ y \le 0 \\ y \ge -x - 2 \end{cases}$.

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.

Практическая работа №9 Оператор цикла с предусловием

Задача

1. Формулировка задачи

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

2. Математическая постановка задачи

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

$$N!=1 \cdot 2 \cdot 3 \cdot \ldots \cdot N$$

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.

Задача

1. Формулировка задачи

Составить алгоритм вычисления суммы ряда:

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

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.

Практическая работа № 10 Оператор цикла с постусловием

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.

Задача

1. Формулировка задачи

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

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.

Практическая работа № 11Оператор цикла с параметром
Задача

1. Формулировка задачи

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

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.

Практическая работа № 12 Работа со строками.

Задача

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.

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

Задача

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.

2.Выбор переменных программы

3. Блок-схема алгоритма.

4.Код программы

Практическая работа № 13,14,15 Алгоритмы поиска, сортировки и замены

Задача

1. Формулировка задачи:

Составить алгоритм и написать программу:

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

2. Выбор переменных программы

3. Блок-схема алгоритма.

4.Код программы

Задача

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.

2.Выбор переменных программы

3. Блок-схема алгоритма.

4.Код программы

Практическая работа № 16 Параметры функции

Задача

1. Формулировка задачи:

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

2. Выбор переменных программы

3. Блок-схема алгоритма.

4. Код программы

Задача

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.

2. Выбор переменных программы

3. Блок-схема алгоритма.

4. Код программы

Практическая работа № 17 Рекурсивные функции

Задача

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'.

2. Выбор переменных программы

3. Блок-схема алгоритма.

4. Код программы

Задача

1.Формулировка задачи:

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

2. Выбор переменных программы

3. Блок-схема алгоритма.

4. Код программы

Практическая работа № 18 Многофайловые проекты

Задача

1. Формулировка задачи:

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

2. Выбор переменных программы

3. Блок-схема алгоритма.

4. Код программы

Задача

1. Формулировка задачи:

Записать в файл последовательного доступа произвольных натуральных чисел. Переписать в другой файл последовательного доступа те элементы, которые кратны *K*. Вывести полученный файл на печать.

2. Выбор переменных программы

3. Блок-схема алгоритма.

4. Код программы

Практическая работа №19 Работа с указателями.

Задача

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!».

2. Выбор переменных программы

3. Блок-схема алгоритма.

4. Код программы

Задача

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).

2. Выбор переменных программы

3. Блок-схема алгоритма.

4. Код программы

Практическая работа №20 Динамическое распределение памяти.
Задача

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.

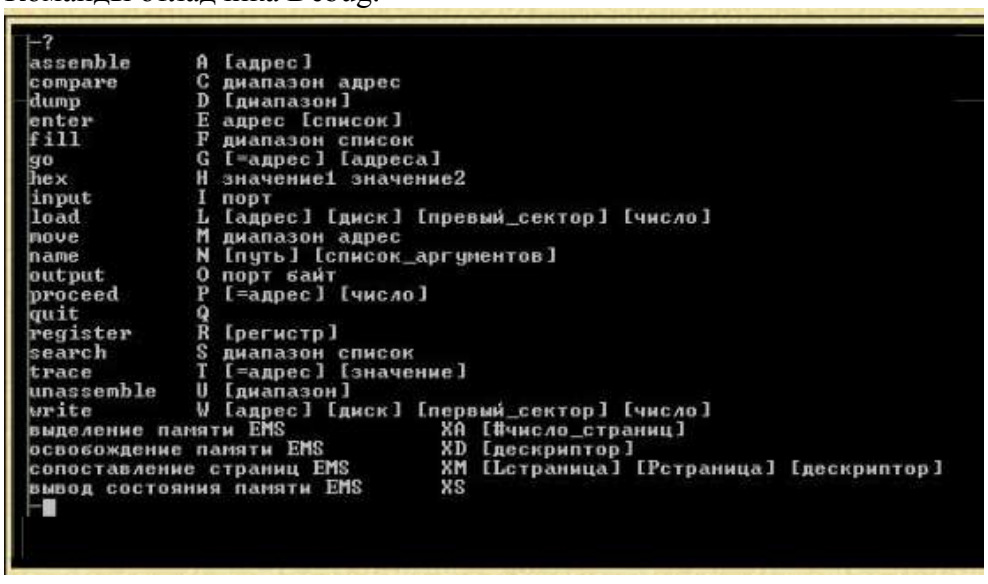
2. Выбор переменных программы

3. Блок-схема алгоритма.

4. Код программы

Практическая работа №21 Работа и использование отладчика AFDP: Основные команды

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/
Выполнить следующие команды и прокомментировать результаты их выполнения:

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

Практическая работа №22 Работа и использование отладчика AFDP: Команды передачи данных
Задания.
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:

	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

Вставить эти 8 команд после 9-й строки.

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.

Методические рекомендации по выполнению 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.

Практическая работа №23 Работа и использование отладчика AFDP: Арифметические команды

Цель работы: Отработка навыков применения арифметических команд при создании программ.
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	-	-	-	-	-	-	-	-	-

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

- Получите исполняемый модуль.
- Проверьте правильность работы программы, запустив ее из отладчика и отслеживая результаты в нужных окнах.

Задача 2. Написать программу для вычисления выражения:
(365/5+915)-(31*11-309)

- Получите исполняемый модуль.
- Проверьте правильность работы программы, запустив ее из отладчика и отслеживая результаты в нужных окнах.

Практическая работа №24 Работа и использование отладчика AFDP: Логические операторы и команды сдвига

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.

Например, если R1=31_{16}, R2=32_{16} и R3=33_{16}, 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`	

Практическая работа №25 Работа и использование отладчика AFDP: Команды передачи управления

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

Варианты заданий

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

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