

Министерство науки и высшего образования Российской Федерации

Томский государственный университет
систем управления и радиоэлектроники

Н. В. Пермякова

ИНФОРМАТИКА

Методические указания к лабораторным работам,
практическим занятиям и организации самостоятельной работы
для студентов направления «Программная инженерия»
(уровень бакалавриата)

Томск
2018

Пермякова, Наталья Викторовна

Информатика: методические указания к лабораторным работам, практическим занятиям и организации самостоятельной работы для студентов направления «Программная инженерия» (уровень бакалавриата) Н.В. Пермякова. – Томск : Томск. гос. ун-т. систем упр. и радиоэлектроники, 2018. – 36 с.

Настоящее учебно-методическое пособие по выполнению лабораторных работ, подготовке к практическим занятиям и организации самостоятельной работы составлено с учетом требований федерального государственного образовательного стандарта высшего образования (ФГОС ВО).

Учебно-методическое пособие содержит краткое описание теоретического материала, необходимого для выполнения указанных видов занятий. Для каждой лабораторной работы определена цель работы, порядок ее выполнения, приведены примеры выполнения типовых заданий. Для практических занятий описан регламент проведения занятия. Для организации самостоятельной работы перечислены темы домашних заданий и контрольных работ, перечислены дополнительные литературные источники.

© Пермякова Н.В., 2018

© Томск. гос. ун-т. систем упр. и радиоэлектроники, 2018

Оглавление

1	МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ПРОВЕДЕНИЮ ЛАБОРАТОРНЫХ РАБОТ	5
1.1	Общие положения	5
1.2	Лабораторная работа «Создание консольного приложения в среде Dev-C++. Ввод-вывод информации»	5
1.3	Лабораторная работа «Проверка ошибок ввода в языке программирования Си»	8
1.4	Лабораторная работа «Проверка условий. Геометрия на плоскости»	10
1.5	Лабораторная работа «Вычисление суммы бесконечного ряда»	14
1.6	Лабораторная работа «Обработка статического одномерного массива»	15
1.7	Лабораторная работа «Обработка двумерных массивов»	17
1.8	Лабораторная работа «Функции»	17
2	МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ПРОВЕДЕНИЮ ПРАКТИЧЕСКИХ ЗАНЯТИЙ	19
2.1	Практическое занятие «Введение в структурное программирование»	19
2.2	Практическое занятие «Структурное программирование»	19
2.3	Практическое занятие «Вычислительная сложность алгоритмов»	20
2.4	Практическое занятие «Разработка циклических алгоритмов»	20
2.5	Практическое занятие «Числовые алгоритмы»	21
2.6	Практическое занятие «Одномерные массивы»	21
2.7	Практическое занятие «Сортировка одномерных массивов»	21
2.8	Практическое занятие «Двумерные массивы»	22
2.9	Практическое занятие «Генерация элементов двумерных массивов»	22
3	МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ОРГАНИЗАЦИИ САМОСТОЯТЕЛЬНОЙ РАБОТЫ	24
3.1	Общие положения	24
3.2	Проработка лекционного материала, подготовка к контрольным работам, лабораторным работам и практическим занятиям	24
3.3	Выполнение домашних заданий и контрольных работ	25
3.4	Самостоятельное изучение тем теоретической части курса	28
3.5	Подготовка к экзамену	31
4	РЕКОМЕНДУЕМЫЕ ИСТОЧНИКИ	33
	ПРИЛОЖЕНИЕ А	34

ВВЕДЕНИЕ

В методических указаниях к проведению лабораторных работ, практических занятий и организации самостоятельной работы по дисциплине «Информатика» собраны материалы для методической поддержки аудиторных занятий и самостоятельной работы студентов.

Целью проведения лабораторных работ, практических занятий и самостоятельной работы является формирование и развитие практических навыков структурного программирования. Для достижения заданной цели во время проведения занятий последовательно решаются следующие задачи:

- изучение основ и приемов алгоритмизации;
- изучение языка программирования языка Си и его применение на практике;
- изучение принципов структурного программирования и применение их на практике;
- изучение структур данных, алгоритмов их обработки и их программная реализация;
- изучение вопросов построения эффективных алгоритмов;
- решение прикладных задач программирования на примере решения задач дискретной математики и вычислительной математики.

По окончании обучения дисциплины «Информатика» студент должен:

- **знать** основные принципы и конструкции структурного программирования, графические способы представления алгоритмов, основные приемы алгоритмизации, синтаксис и алфавит языка Си;
- **уметь** разрабатывать алгоритмы поставленных перед ним задач, представлять алгоритмы в графическом виде и в виде программ на языке высокого уровня, использовать базовые алгоритмы для решения прикладных задач;
- **владеть** навыками реализации, отладки и тестирования программ на алгоритмических языках программирования, навыками использования различных структур данных при решении задач.

Цикл лабораторных работ и практических занятий по дисциплине, выполняемых студентами, направлен на изучение и закрепление навыков использования конструкций структурного программирования.

Согласно действующему Федеральному Государственному стандарту высшего образования (ФГОС ВО) значительная доля курса отведена на самостоятельную работу студентов. При изучении дисциплины «Информатика» самостоятельная работа студентов включает несколько видов деятельности – проработка лекционного материала, подготовка к контрольным работам, выполнение домашних заданий, самостоятельное изучение тем, подготовка к экзамену. В методическом пособии предложены рекомендации по организации перечисленных видов деятельности, включающие порядок выполнения отдельных видов самостоятельной работы, краткие теоретические сведения, перечень литературных источников по темам дисциплины, вынесенным на самостоятельное изучение.

1 МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ПРОВЕДЕНИЮ ЛАБОРАТОРНЫХ РАБОТ

1.1 Общие положения

Целью проведения лабораторных работ является формирование и развитие навыков структурного программирования.

Основной формой проведения лабораторных работ является разработка алгоритма решения индивидуальной задачи и его программная реализация на языке Си. Процесс программной реализации включает в себя написание программы, отладку программы и тестирование программы.

К основным способам контроля формирования компетенций при выполнении лабораторных работ относятся индивидуальная защита выполненной работы, организация опроса студентов по теоретическому материалу дисциплины, практическое применение которого осуществляется в ходе выполнения лабораторной работы.

Для получения максимальной оценки за лабораторную работу необходимо выполнить и защитить работу во время, отведенное для ее выполнения, согласно расписанию занятий. Допускается досрочное выполнение лабораторной работы по предварительной договоренности с преподавателем.

Выполнение всех лабораторных работ, предусмотренных рабочей программой дисциплины, является условием допуска к итоговому контролю изучения дисциплины – экзамену.

1.2 Лабораторная работа «Создание консольного приложения в среде Dev-C++. Ввод-вывод информации»

Цель работы: ознакомиться с интегрированной средой Dev-C++, изучить основные типы данных языка Си, функции ввода и вывода информации, получить навыки написания программ на языке Си.

Форма проведения: выполнение индивидуального задания.

Подготовка к выполнению лабораторной работы: для выполнения лабораторной работы необходимо изучить теоретический материал, изложенный в [1]. Описание процесса создания проекта в IDE Dev-C++ рассматривается в главе 2 пособия (стр. 33 – 35). Описание структуры простой программы (стр. 81 – 82) и универсальных функций ввода-вывода информации (стр. 77 – 81) в главе 5 пособия. Для реализации индивидуального задания необходимо ознакомиться с операторами языка Си (стр. 52 – 55) и основными типами данных (стр. 59 – 60).

Порядок выполнения работы

1. Получить индивидуальный вариант;
2. создать проект в Dev-C++;
3. написать программу на языке Си, выполняемые функции которой могут быть описаны следующей последовательностью шагов:
 - 3.1. описать входные и выходные данные;
 - 3.2. ввести данные с клавиатуры;
 - 3.3. вычислить значение функции;
 - 3.4. вывести полученное значение на экран;
 - 3.5. вывести личные данные.
4. выполнить компиляцию проекта;
5. выполнить тестирование проекта;
6. защитить работу.

Контрольные вопросы

1. Какое имя носит исполняемая функция Си?
2. Дайте определение понятия «переменная».
3. Дайте определение понятия «идентификатор».
4. Сколько переменных требуется описать в программе, если необходимо решить следующую задачу – «С клавиатуры вводятся три числа, необходимо вывести на экран значение минимального из этих трех чисел»?
5. Какая функция используется в Си для ввода информации?
6. Какая функция используется в Си для вывода информации?
7. Какой тип данных Си соответствует спецификатору «%d»?
8. Какой тип данных Си соответствует спецификатору «%f»?
9. Переменная j описана в программе следующим образом: `int j`; Запишите функцию `scanf` для считывания значения в переменную j .
10. Переменная k описана в программе следующим образом: `float k`; Запишите функцию `printf` для вывода значения переменной k .

Пример выполнения индивидуального варианта

Вариант 1. Ввести с клавиатуры целое число x . Вывести на экран значение функции $x^2 + 3.1x + 7.5$ и сообщение вида: «Программу выполнил ФИО».

В таблице 1.1 представлена программа, реализующая индивидуальное задание.

Таблица 1.1 – Этапы выполнения лабораторной работы

№	Название этапа	Описание результатов выполнения этапа
2	Создание проекта	Запустить Dev-C++, создать новый проект, дополнить код программы вызовом функции <code>system ("chcp 1251");</code> – смена кодировки страницы.
3	Реализация программы	
3.1	Описание переменных	<code>int x;</code> <code>float y;</code>
3.2	Ввод данных с клавиатуры	<code>printf("Введите значение переменной x: ");</code> <code>scanf("%d",&x);</code>
3.3	Вычисление значения функции	<code>y = x*x+3.1*x + 7.5;</code>
4	Вывод результата	<code>printf("Значение функции: %7.2f\n",y);</code>
5	Вывод личных данных	<code>printf("Программу выполнил Иванов Андрей Сергеевич\n");</code>

После набора представленной выше программы в шаблоне функции `main ()` (рис. 1.1) выполните компиляцию проекта.

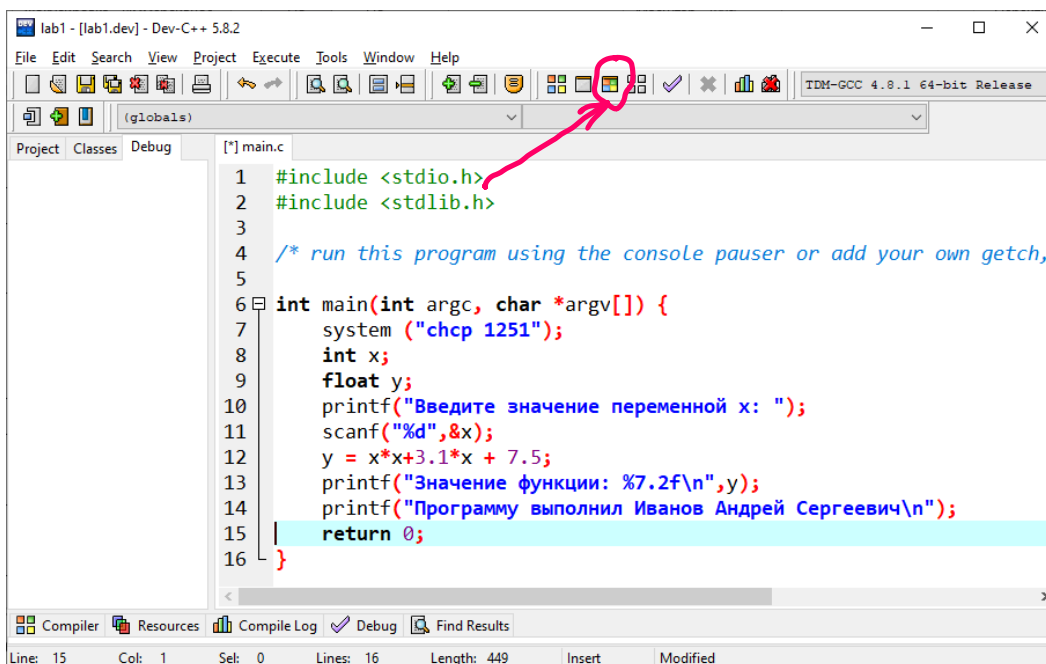


Рисунок 1.1 – Проект выполнения индивидуального задания

Определите имя файла, содержащего функцию `main()`. В рассматриваемом примере имя файла определено как `lab1` (рис. 1.2).

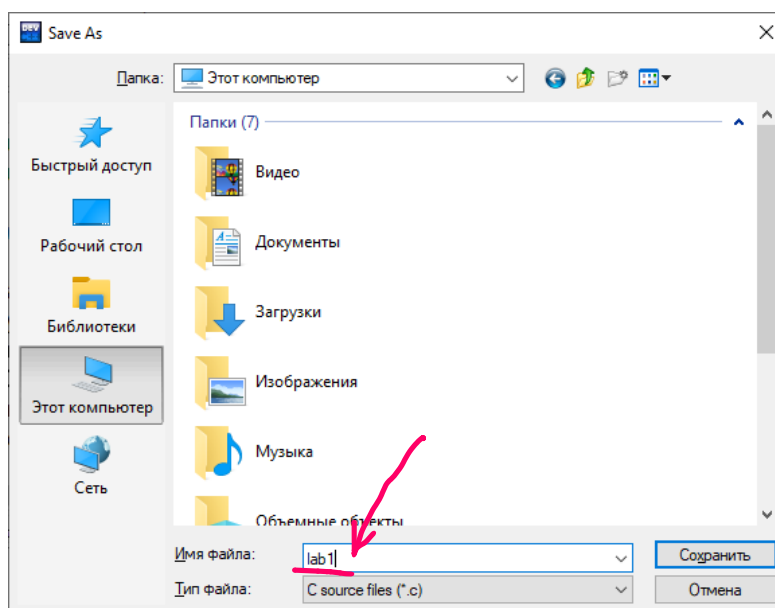


Рисунок 1.2 – Определение имени файла

Если этап компиляции прошел успешно, программа автоматически выполнится. Для смены кодировки страницы зайдите в свойства консольного окна, выберите вкладку «Шрифт» и выберите шрифт `Lucida Console`.

Результат работы программы представлен на рисунке 1.3.

```
C:\Users\pсрыпш\шльсюёютэр\Downloads\lab1.exe
Текущая кодовая страница: 1251
Введите значение переменной x: 12
Значение функции: 188.70
Программу выполнил Иванов Андрей Сергеевич
-----
Process exited after 15.93 seconds with return value 0
для продолжения нажмите любую клавишу . . .
```

Рисунок 1.3 – Тестирование программы при $x = 12$

При выполнении лабораторной работы в консольном окне выведено полученное значение переменной и личные данные студента, выполняющего работу.

1.3 Лабораторная работа «Проверка ошибок ввода в языке программирования Си»

Цель работы: ознакомиться с возможностями функции `scanf()`. Научиться составлять условные алгоритмы на примере алгоритма проверки ошибок ввода данных. Реализовать алгоритм на языке Си.

Форма проведения: выполнение индивидуального задания.

Рекомендации по подготовке к лабораторной работе: для выполнения лабораторной работы необходимо изучить теоретический материал, изложенный в [1]. Основные конструкции структурного программирования рассматриваются в главе 1 пособия (стр. 13 – 15). Описание синтаксиса конструкции проверки условия в языке Си (стр. 87 – 90) в главе 6 пособия. Возможности функции `scanf` для организации проверки корректности ввода данных описаны на стр. 80 – 81.

Порядок выполнения работы

1. Получить индивидуальный вариант;
2. по индивидуальному варианту определить типы и значения данных, являющиеся некорректными для задачи;
3. составить и записать алгоритм решения задачи;
4. составить программу, реализующую алгоритм:
 - 4.1. описать входные и выходные данные;
 - 4.2. ввести данные с клавиатуры;
 - 4.3. проверить входные данные;
 - 4.4. вычислить значение функции;
 - 4.5. вывести полученное значение на экран;
 - 4.6. вывести личные данные;
 - 4.7. выполнить компиляцию проекта;
5. защитить работу.

Пример выполнения индивидуального варианта

Вариант 1. Составить и записать алгоритм решения индивидуального задания с проверкой корректности данных. По составленному алгоритму написать программу на языке Си.

Даны x, y, z . Вычислить a, b , если $a = \left(\frac{2y}{z-x} - |x|\right) \left(\sqrt[4]{x - \frac{y}{\sqrt{x}}}\right)$, $b = a - \frac{x}{2a} + \frac{x^2}{a-z}$. Значения x, y, z вводить с клавиатуры.

В таблице 1.2 описано поэтапное выполнение лабораторной работы.

Таблица 1.2 – Этапы выполнения лабораторной работы

№	Название этапа	Описание результата выполнения этапа
2.	Определение некорректных данных	Символьные данные; $z - x = 0$ – ошибка деления на 0; $x \leq 0$ – ошибка извлечения квадратного корня, ошибка деления на 0; $x - \frac{y}{\sqrt{x}} < 0$ – ошибка извлечения корня четвертой степени; $a = 0$ – ошибка деления на 0; $a = z$ – ошибка деления на 0.
3.	Алгоритм	<u>алг</u> Лабораторная работа 2 <u>нач</u> <u>ввод</u> x,y,z <u>если</u> x или y или z – не число, <u>то</u> <u>рез</u> “Введены не числа” <u>если</u> z-x=0 или x = 0, <u>то</u> <u>рез</u> “Ошибка деления на 0” <u>если</u> $x - \frac{y}{\sqrt{x}} < 0$ или $x < 0$ <u>то</u> <u>рез</u> “Ошибка извлечения корня” $a = \left(\frac{2y}{z-x} - x \right) \left(\sqrt[4]{x - \frac{y}{\sqrt{x}}} \right)$ <u>рез</u> a <u>если</u> a = 0 или a = z <u>то</u> <u>рез</u> “Ошибка деления на 0” $b = a - \frac{x}{2a} + \frac{x^2}{a-z}$ <u>рез</u> b <u>конец</u>
4.	Составление программы	Для использования математических функций и констант подключите библиотеку математических функций: <code>#include <math.h></code>
4.6.	Вывод личных данных	<code>printf(“Программу выполнил Иванов Андрей Сергеевич\n”);</code>
4.1.	Описание переменных	<code>float x,y,z,a,b;</code>
4.2.	Ввод данных с клавиатуры	<code>printf(“Введите значения переменных: ”);</code> <code>int m = scanf(“%f%f%f”,&x,&y,&z);</code>
4.3.	Проверка входных данных	<code>if (m!=3) {</code> <code>printf (“Введены не числа\n”);</code> <code>system(“pause”);</code> <code>return 0;}</code> <code>if (x==0 z-x==0) {</code> <code>printf (“Ошибка деления на 0\n”);</code> <code>system(“pause”);</code> <code>return 0;}</code>

		float temp = x - y/sqrt(x); if (temp<0 x<0) { printf("Ошибка извлечения корня\n"); system("pause"); return 0;}
4.4	Вычисление значения функции	a = 2*y/(z-x)-fabs(x); a = a*pow(temp,1/4.);
4.5	Вывод результата	printf("Значение a = %9.3f\n",a);
4.3.	Проверка входных данных	if (a==0 a==z) { printf ("Ошибка деления на 0\n"); system("pause"); return 0;}
4.4	Вычисление значения функции	b = a - x/(2*a)+ x*x/(a-z);
4.5	Вывод результата	printf("Значение b = %9.3f\n",b);

Контрольные вопросы

1. Что возвращает функция *scanf()*?
2. Запишите функцию *scanf()* для ввода трех переменных.
3. Что Вы понимаете под некорректными данными?
4. Какие данные будут некорректными для решения следующей задачи – “Даны длины трех сторон треугольника. Найдите площадь треугольника.”
5. Как в Си реализована условная конструкция структурного программирования?
6. Опишите синтаксис конструкции *if else* языка Си.
7. Какое значение примет переменная *m* после выполнения следующего фрагмента программы:

```
...
float i;
int j;
int m = scanf("%f%d",&i, &j);
...
```

если с клавиатуры были введены значения 3 2?

8. Какое значение примет переменная *m* после выполнения следующего фрагмента программы:

```
...
float i;
int j;
int m = scanf("%f%d",&i, &j);
...
```

если с клавиатуры были введены значения 3 d?

9. Какое значение примет переменная *x* после выполнения следующего фрагмента программы, представленного ниже?

```
...
int x = 10;
int k = 12,z = 74;
if (k<z) x = 1; else x = 0;
...
```

1.4 Лабораторная работа «Проверка условий. Геометрия на плоскости»

Цель работы: закрепление навыков построения разветвляющихся алгоритмов.

Форма проведения: выполнение индивидуального задания.

Проверка расположения точки с координатами (x, y) относительно прямой: пусть уравнение прямой задано в каноническом виде $y = ax + b$. Тогда все точки, лежащие на линии прямой (рис. 1.4), подчиняются условию $y = ax + b$. На рисунке это условие выполняется для точки с координатами (x_3, y_3) . Все точки, лежащие левее линии прямой, подчиняются условию $y < ax + b$, это условие выполняется для точки с координатами (x_1, y_1) . Все точки, лежащие правее линии прямой, подчиняются условию $y > ax + b$. Это условие является истинным для точки с координатами (x_2, y_2) . Тогда для выбранных трех точек являются истинными условия: $y_3 = ax_3 + b$; $y_2 > ax_2 + b$; $y_1 < ax_1 + b$.

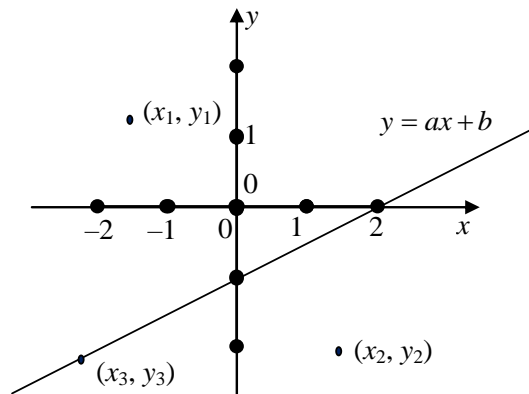


Рисунок 1.4 – Расположение точки относительно прямой

Для прямой, изображенной на рис. 4, составим уравнение прямой по двум заданным точкам: прямая проходит через точки с координатами $(0, -1)$ и $(2, 0)$. Найдем коэффициенты уравнения a и b . Для этого решим систему уравнений:

$$\begin{cases} -1 = a \cdot 0 + b \\ 0 = a \cdot 2 + b \end{cases} \Rightarrow \begin{cases} b = -1 \\ a = \frac{-b}{2} \end{cases} \Rightarrow \begin{cases} b = -1 \\ a = 0.5 \end{cases} \Rightarrow y = 0.5x - 1.$$

Таким образом, проверить местоположение произвольной точки с координатами (x, y) можно, написав следующий код:

...

```
if (y < 0.5*x - 1)
printf("Точка расположена левее прямой");
else if (y > 0.5*x - 1)
printf("Точка расположена правее прямой");
else printf("Точка расположена на прямой");
```

...

Проверка расположения точки относительно окружности с заданным центром известного радиуса: каноническое уравнение окружности выглядит следующим образом:

$$R^2 = (x - x_1)^2 + (y - y_1)^2, \quad (1)$$

где R – радиус окружности,

(x_1, y_1) – координаты центра окружности.

Тогда (рис. 1.5) для точки с координатами (x_4, y_4) , выполняется равенство:

$$R^2 = (x_4 - x_1)^2 + (y_4 - y_1)^2.$$

Выражение (1) истинно для всех точек, лежащих на линии окружности. Для точки с координатами (x_2, y_2) и для всех точек, лежащих за окружностью, выполняется неравенство:

$$R^2 < (x_2 - x_1)^2 + (y_2 - y_1)^2.$$

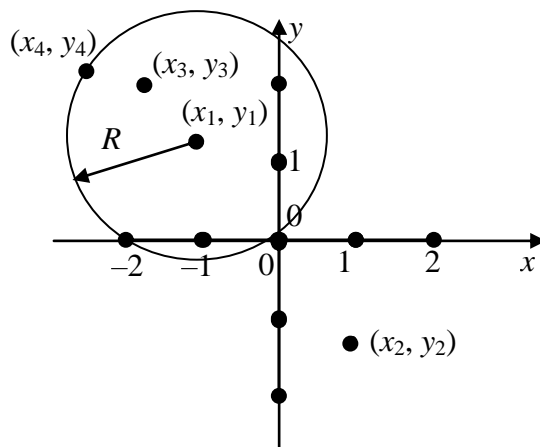


Рисунок 1.5 – Расположение точки относительно окружности

Из этого неравенства следует, что радиус R окружности меньше радиуса окружности с центром в точке (x_1, y_1) , на которой лежит точка с координатами (x_2, y_2) . Соответственно, для точки с координатами (x_3, y_3) выполняется неравенство: $R^2 > (x_3 - x_1)^2 + (y_3 - y_1)^2$.

То есть радиус R окружности, на которой лежит точка с координатами (x_3, y_3) больше радиуса окружности с центром в точке (x_1, y_1) .

Порядок выполнения работы

1. Получить индивидуальный вариант.
2. Определить условия вхождения точки в заданную область.
3. Составить и записать алгоритм решения задачи.
4. Составить программу, реализующую алгоритм:
 - 4.1. описать входные и выходные данные;
 - 4.2. ввести данные с клавиатуры;
 - 4.3. проверить входные данные;
 - 4.4. проверить условие вхождения точки в заданную область;
 - 4.5. вывести результат проверки на экран;
 - 4.6. вывести личные данные.
5. Выполнить компиляцию проекта.
6. Защитить работу.

Примеры проверки вхождения точки с заданными координатами в фигуру

Пример 1. Напишите алгоритм, проверяющий вхождение точки в область, изображенную на рисунке 1.6.

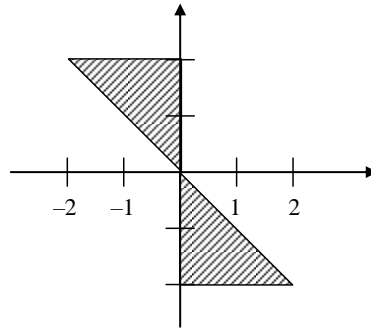


Рисунок 6 – Пример 1

Построим условия вхождения точки в заданную область:

Уравнение прямой, на которой лежат гипотенузы прямоугольных треугольников, образующих фигуру: $y = -x$.

Разобьем фигуру на две части. Точка будет считаться принадлежащей фигуре, если она попадет в первую или вторую часть.

Первую (верхнюю часть) можно ограничить следующими условиями: $(y \geq -x)$ и $(x \leq 0)$ и $(y \leq 2)$.

Первое условие описывает гипотенузу, второе и третье условие описывают катеты. Условия связаны между собой связками **И** (логическое умножение).

Вторую (нижнюю часть) можно ограничить условиями:

$(y \leq -x)$ и $(x \geq 0)$ и $(y \geq -2)$.

Общее условие для двух частей будет выглядеть следующим образом:

если $(y \geq -x)$ **и** $(x \leq 0)$ **и** $(y \leq 2)$ **или**

$(y \leq -x)$ **и** $(x \geq 0)$ **и** $(y \geq -2)$,

то «Точка принадлежит заданной области»,

иначе «Точка не принадлежит заданной области».

Тогда алгоритм проверки вхождения точки с заданными координатами будет выглядеть следующим образом:

алг Пример 1 нач

вещ x, y

ввод x, y

если $(y \geq -x)$ **и** $(x \leq 0)$ **и** $(y \leq 2)$ **или**

$(y \leq -x)$ **и** $(x \geq 0)$ **и** $(y \geq -2)$,

то «Точка принадлежит заданной области»,

иначе «Точка не принадлежит заданной области»

кон

Пример 2. Рассмотрим еще один пример, заданная область изображена на рисунке 1.7. В этом случае: уравнение прямой $y = x$; уравнение окружности $I = (x - 1)^2 + (y - 1)^2$.

Ограниченная область находится:

- правее прямой ($y < x$);
- внутри окружности ($1 > (x - 1)^2 + (y - 1)^2$).

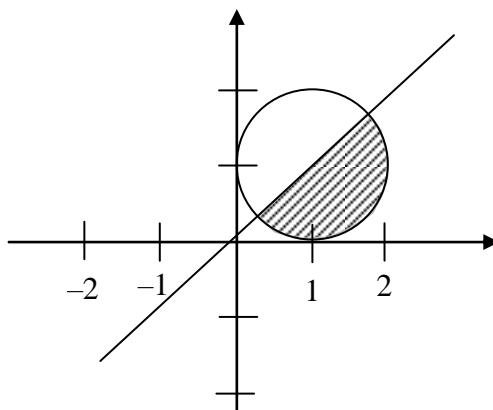


Рисунок 1.7 – Пример 2

Тогда общее условие будет выглядеть следующим образом:

если $y < x$ **и** $1 > (x - 1)^2 + (y - 1)^2$,

то «Точка принадлежит заданной области»,

иначе «Точка не принадлежит заданной области».

В этом случае алгоритм проверки вхождения точки с заданными координатами будет выглядеть следующим образом:

алг Пример 2 нач

вещ x, y

ввод x, y

если $y < x$ **и** $1 > (x - 1)^2 + (y - 1)^2$,

то «Точка принадлежит заданной области»,

иначе «Точка не принадлежит заданной области»

кон

Контрольные вопросы

1. Как должно быть построено условие нахождения точки с заданными координатами относительно прямой?
2. Как определяются условия расположения точки с заданными координатами относительно окружности?
3. Какие логические операторы используются для построения условий?
4. Какие средства языка программирования Си позволяют реализовать разветвляющиеся алгоритмы?

1.5 Лабораторная работа «Вычисление суммы бесконечного ряда»

Цель работы: закрепление навыков программирования итерационных процессов.

Форма проведения: выполнение индивидуального задания.

Рекомендации по подготовке к лабораторной работе: для выполнения лабораторной работы необходимо изучить теоретический материал, изложенный в [1]. Основные конструкции структурного программирования рассматриваются в главе 1 пособия (стр. 13 – 15). Описание синтаксиса конструкций циклов в языке Си (стр. 91 – 95) в главе 6 пособия. Математические основы вычисления суммы бесконечного ряда и логика процесса вычисления суммы бесконечного ряда описана на стр. 95 – 97.

Порядок выполнения работы

1. Получить индивидуальный вариант.
2. Если необходимо, преобразовать исходную формулу ряда в рекуррентную формулу.
3. Написать и протестировать программу вычисления суммы бесконечного ряда.
4. Защитить работу.

Контрольные вопросы

1. Почему при программировании алгоритма вычисления бесконечного ряда рекомендуется вывести рекуррентную формулу?
2. К каким ошибкам может привести вычисление степени аргумента и факториала при программировании бесконечного ряда?
3. Что такое точность вычисления бесконечного ряда?
4. Какой прием алгоритмизации был использован для программирования $(-1)^n$?
5. Как зависит количество итераций алгоритма от заданной точности вычислений?
6. Как зависит количество итераций алгоритма от заданного значения алгоритма?

1.6 Лабораторная работа «Обработка статического одномерного массива»

Цель работы: закрепление навыков обработки статического одномерного массива.

Форма проведения: выполнение индивидуального задания.

Рекомендации по подготовке к лабораторной работе: для выполнения лабораторной работы необходимо изучить теоретический материал, изложенный в [1]. Справочная информация об организации работы с массивами описана в главе 8 пособия (стр. 114 – 120). Графические способы представления алгоритмов описываются в главе 1 (стр. 15 – 20).

Порядок выполнения работы

1. Получить индивидуальное задание.
2. Составить алгоритм решения задания и реализовать его графическое представление (блок-диаграмма, диаграмма Насси-Шнайдермана).
3. Составить программу на языке Си.
4. Выполнить отладку и тестирование программы.
5. Защитить работу.

Пример выполнения индивидуального задания

Вводится последовательность из n целых чисел. Сохранить все введенные значения в массиве и найти сумму всех отрицательных чисел (*индивидуальное задание*).

Блок-диаграмма задачи представлена на рисунке 1.8.

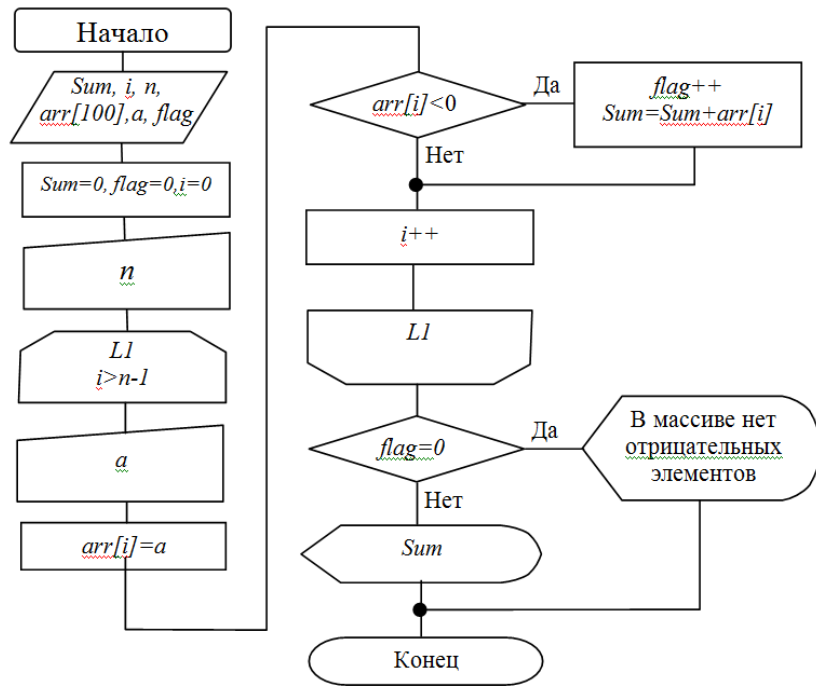


Рисунок 1.8 – Алгоритм поиска суммы отрицательных элементов

Реализация задания на языке Си может быть выполнена следующим образом:

```

int main(int argc, char *argv[])
{
    system("chcp 1251");
    int Sum=0, a, n, arr[100];
    int flag = 0;
    printf("Введите количество членов последовательности не более 100 ");
    scanf("%d",&n);
    for(int i=0; i<n; i++){
        printf("Введите значение члена последовательности");
        scanf("%d", &a);
        arr[i]=a;
        if (arr[i]<0) {Sum+=arr[i]; flag++;}
    }
    if (!flag) printf("В массиве нет отрицательных элементов\n");
    else
    printf("Значение суммы отрицательный членов
    последовательности равна %6d", Sum);}
  
```

Контрольные вопросы

1. Почему нумерация элементов массива в языке Си начинается с 0?
2. Как связаны указатели и массивы?
3. Как располагаются в памяти компьютера элементы массива?
4. Что произойдет если в программе на языке Си обратиться к несуществующему элементу массива?
5. Правильно ли реализован алгоритм поиска минимального элемента массива, представленный ниже?

```

int x[10];
int i,min;
for(i=0;i<n-1;i++)
  
```



```
if(x[i]<x[i+1]) min = x[i]; else min = x[i+1];
```

б. Приведите пример массива, для которых алгоритм будет работать верно.

1.7 Лабораторная работа «Обработка двумерных массивов»

Цель работы: изучение способов работы с динамическими двумерными массивами; реализация простых алгоритмов сортировки и поиска; организация отладки программы с помощью встроенного отладчика.

Форма проведения: выполнение индивидуального задания.

Рекомендации по подготовке к лабораторной работе: для выполнения лабораторной работы необходимо изучить теоретический материал, изложенный в [1]. В главе 8 пособия изложена справочная информация об организации работы с двумерными массивами в языке Си (стр. 114 – 120), способы сортировки наборов данных рассмотрены (стр. 121 – 123). Руководство по использованию встроенного отладчика в IDE описано в главе 2 (стр. 37 – 42).

Порядок выполнения работы

1. Получить индивидуальный вариант.
2. Изучить теоретические аспекты лабораторной работы.
3. Изучить работу отладчика в среде Dev-C++.
4. Разработать и реализовать на языке Си алгоритм решения предложенных задач.
5. Защитить работу, используя средства отладчика на тестовом примере с матрицей [5x5].

Контрольные вопросы

1. Какая матрица используется в программной реализации – статическая или динамическая?
2. Опишите механизм выделения памяти для двумерного массива?
3. Какими способами можно изменить порядок строк в матрице?
4. Как представить двумерный массив в виде одномерного массива?
5. Что определяет первый индекс двумерного массива?
6. Опишите алгоритм транспонирования матрицы.

1.8 Лабораторная работа «Функции»

Цель работы: закрепление навыков разработки функций.

Форма проведения: выполнение индивидуального задания.

Рекомендации по подготовке к лабораторной работе: для выполнения лабораторной работы необходимо изучить теоретический материал, изложенный в [1]. В главе 7 пособия описаны основные моменты организации функций на языке Си (стр. 103 – 109).

Порядок выполнения работы

1. Получить индивидуальный вариант.
2. Написать программу, решающую поставленную задачу с использованием функций.
3. Отладить и протестировать программу.
4. Подготовить электронный вариант документа, содержащий описание функций.
5. Защитить работу.

Пример описания функции

```
int fprintf( FILE * stream, const char * format, ... );
```

Описание

Функция *fprintf* выполняет форматированный вывод в поток. Записывает в указанный поток последовательность символов в формате, указанном аргументом *format*. После параметра *format*, функция ожидает, по крайней мере, многие дополнительные аргументы, как указано в прототипе.

Параметры:

stream

Указатель на объект типа *FILE*, который связан с потоком.

format

Си-строка, содержащая текст, который будет выведен на поток. Опционально, строка может содержать встроенные метки форматирования, которые заменяются значениями, указанными в последующих дополнительных аргументах и отформатированы требуемым образом.

Дополнительные аргументы

В зависимости от формата строки, функция может принимать дополнительные аргументы, каждый из которых содержит одно значение. Вместо каждого %-тега, указанного в параметре *format*, в поток вывода будет вставлено значение соответствующего аргумента. Количество дополнительных аргументов должно соответствовать количеству %-тегов.

Возвращаемое значение

В случае успеха, возвращается общее число записанных символов.

В случае неудачи, возвращается отрицательное число.

Контрольные вопросы

1. Где в описании функции указывается тип возвращаемого значения?
2. В каком случае необходимо использовать передачу параметров по ссылке?
3. Какой оператор языка Си заканчивает выполнение функции?
4. Обязательно ли функция должна возвращать значение?
5. Как в языке Си можно реализовать передачу функции параметром?

2 МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ПРОВЕДЕНИЮ ПРАКТИЧЕСКИХ ЗАНЯТИЙ

2.1 Практическое занятие «Введение в структурное программирование»

Цель занятия: ознакомление с принципами структурного программирования.

Форма проведения: решение практических заданий.

Порядок проведения занятия

1. Обсуждение основных принципов структурного программирования, введение понятия алгоритма, способы представления алгоритмов.
2. Разбор и решение задач по теме занятия.
3. Выдача индивидуальных вариантов домашнего задания.

Примеры задач

Пример 1

Дано целое число a и натуральное число n . Вычислить $P = a^n$.

Пример 2

Даны натуральные числа a, b . Вычислить произведение $a * b$, используя лишь операции $+, -, =, < >$.

Пример 3

Дано натуральное число a и целое положительное b . Вычислить частное q и остаток r при делении a на b , не используя операции div и mod .

2.2 Практическое занятие «Структурное программирование»

Цель занятия: проверка домашнего задания.

Форма проведения: выборочный опрос студентов.

Порядок проведения занятия

Демонстрация выполнения домашнего задания у доски (защита реализованного в процессе выполнения домашнего задания алгоритма, ответы на дополнительные вопросы).

Примеры индивидуальных заданий

Вариант 1

Поезд прибывает на станцию в a часов b минут и отправляется в c часов d минут. Пассажир пришел на платформу в n часов m минут. Будет ли поезд стоять на платформе? Числа a, b, c, d, n, m – целые, $0 < a \leq 23, 0 < b \leq 59, 0 < c \leq 23, 0 < d \leq 59, 0 < n \leq 23, 0 < m \leq 59$.

Вариант 2

Написать программу умножения двух натуральных чисел (реализовать алгоритм аль-Хорезми).

Вариант 3

Написать программу нахождения всех трехзначных чисел, равных сумме факториалов своих цифр.

2.3 Практическое занятие «Вычислительная сложность алгоритмов»

Цель занятия: обучение основным принципам написания эффективных алгоритмов с точки зрения временных характеристик.

Форма проведения: разбор решений задач.

Порядок проведения занятия

1. Введение понятия вычислительной сложности алгоритма, классификация функций, оценивающих вычислительную сложность алгоритма, принципы построения циклических алгоритмов.

2. Разбор и решение задач по теме занятия.

3. Выдача индивидуальных вариантов домашнего задания.

Примеры задач

Пример 1

Вычисление суммы бесконечного ряда.

Пример 2

Вычисление значения многочлена по формуле Горнера.

Пример 3

Примеры преобразования циклов (чистка циклов, объединение циклов, вынос ветвей из тела циклов, раскрутка циклов).

2.4 Практическое занятие «Разработка циклических алгоритмов»

Цель занятия: проверка домашнего задания.

Форма проведения: выборочный опрос студентов.

Порядок проведения занятия

Демонстрация выполнения домашнего задания у доски (защита реализованного в процессе выполнения домашнего задания алгоритма, ответы на дополнительные вопросы).

Примеры индивидуальных заданий

Вариант 1

Вычислить $\cos(z)$ по итерационной формуле $1 - \frac{z^2}{2!} + \frac{z^4}{4!} - \frac{z^6}{6!} + \dots$ заданной точностью.

Значение точности и переменной z задавать с клавиатуры. Сравнить результаты, полученные при вычислении по формуле и при использовании стандартной функции $\cos()$.

Вариант 2

Дано натуральное n , действительное число x . Вычислить: $\sum_{i=1}^n \frac{x + \cos(ix)}{2^i}$.

Вариант 3

Вычислить \sqrt{x} по итерационной формуле $z_{n+1} = \frac{z_n}{2} + \frac{x}{2z_n}$. Вычисления проводятся

пока $|z_{n+1} - z_n| > eps, z_0 = 1,25$. Значение точности eps и переменной x задавать с клавиатуры. Сравнить результаты, полученные при вычислении по формуле и при использовании стандартной функции $\text{sqrt}()$.

2.5 Практическое занятие «Числовые алгоритмы»

Цель занятия: знакомство с алгоритмами элементарной арифметики.

Форма проведения: разбор решений задач.

Порядок проведения занятия

1. Разбор алгоритмов возведения в степень, нахождения наибольшего общего делителя, нахождения простых множителей числа N , проверки на простоту.
2. Разбор и решение задач по теме занятия.
3. Выдача индивидуальных вариантов домашнего задания.

Примеры задач

Пример 1

Эффективный алгоритм возведения в степень.

Пример 2

Алгоритм Евклида.

Пример 3

Проверка числа на простоту.

2.6 Практическое занятие «Одномерные массивы»

Цель занятия: изучение алгоритмов работы с одномерными массивами.

Форма проведения: разбор решений задач.

Порядок проведения занятия

1. Разбор алгоритмов поиска в массивах, добавления элементов в массив, удаления элементов из массива, изменения порядка элементов массива.
2. Разбор и решение задач по теме занятия.
3. Выдача индивидуальных вариантов домашнего задания.

Примеры задач

Пример 1

Алгоритм поиска минимального элемента.

Пример 2

Изменение порядка элементов массива.

Пример 3

Вставка новых элементов массива.

2.7 Практическое занятие «Сортировка одномерных массивов»

Цель занятия: проверка домашнего задания.

Форма проведения: выборочный опрос студентов.

Порядок проведения занятия

Демонстрация выполнения домашнего задания у доски (защита реализованного в процессе выполнения домашнего задания алгоритма, ответы на дополнительные вопросы).

Примеры индивидуальных заданий

Вариант 1

Реализация сортировки одномерного массива методом обмена. Нахождение количества сравнений, выполняемых алгоритмом.

Вариант 2

Реализация сортировки одномерного массива методом вставки. Нахождение количества обменов, выполняемых алгоритмом.

Вариант 3

Реализация сортировки одномерного массива методом выбора. Нахождение количества сравнений, выполняемых алгоритмом.

2.8 Практическое занятие «Двумерные массивы»

Цель занятия: изучение алгоритмов работы с матрицами.

Форма проведения: разбор решений задач.

Порядок проведения занятия

1. Разбор алгоритмов поиска в матрицах, выделения элементов матриц, изменения порядка строк и столбцов матрицы.
2. Разбор и решение задач по теме занятия.
3. Выдача индивидуальных вариантов домашнего задания.

Примеры задач

Пример 1

Алгоритм перехода от двумерного массива к одномерному.

Пример 2

Сортировка строк матрицы.

Пример 3

Выделение главной диагонали квадратной матрицы.

2.9 Практическое занятие «Генерация элементов двумерных массивов»

Цель занятия: проверка домашнего задания.

Форма проведения: выборочный опрос студентов.

Порядок проведения занятия

Демонстрация выполнения домашнего задания у доски (защита реализованного в процессе выполнения домашнего задания алгоритма, ответы на дополнительные вопросы).

Примеры индивидуальных заданий

Вариант 1

Заполнение квадратной матрицы по следующему правилу (на примере матрицы 5×5):

```
1 0 0 0 1
0 1 0 1 0
0 0 1 0 0
0 1 0 1 0
1 0 0 0 1
```

Вариант 2

Заполнение квадратной матрицы по следующему правилу (на примере матрицы 5×5):

1 2 3 4 5
6 7 8 9 10
11 12 13 14 15 .
16 17 18 19 20
21 22 23 24 25

Вариант 3

Заполнение квадратной матрицы по следующему правилу (на примере матрицы 5×5):

1 0 0 0 0
2 1 0 0 0
2 2 1 0 0.
2 2 2 1 0
2 2 2 2 1

3 МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ОРГАНИЗАЦИИ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

3.1 Общие положения

Самостоятельная работа является важной составляющей в изучении дисциплины и состоит из следующих видов деятельности: проработка лекционного материала для подготовки к тестированию и контрольным работам, подготовка к лабораторным работам и практическим заданиям, выполнение домашних заданий, выполнение контрольных работ, самостоятельное изучение тем курса.

Самостоятельная работа над теоретическим материалом направлена на систематизацию и закрепление знаний, полученных на лекционных занятиях и на получение новых знаний по дисциплине, путем самостоятельного изучения тем.

Самостоятельная работа по подготовке к лабораторным работам и практическим занятиям направлена на изучение методического и теоретического материала по теме лабораторной работы или практического занятия.

Выполнение домашних заданий (для студентов очной формы обучения) и контрольных работ (для студентов заочной формы обучения) – полностью самостоятельная работа, направленная на получение навыков самостоятельного составления алгоритмов, реализацию программ, их дальнейшей отладки и тестирования.

3.2 Проработка лекционного материала, подготовка к контрольным работам, лабораторным работам и практическим занятиям

Проработка лекционного курса является одной из важных активных форм самостоятельной работы. Этот вид самостоятельной работы может быть организован следующим образом:

- прочитайте конспект лекции, согласовав Ваши записи с информацией на слайдах лекции;
- попробуйте выполнить самостоятельно примеры программ, разобранных на лекции;
- если в лекции рассматривался какой-либо алгоритм, попытайтесь выполнить этот алгоритм на тестовых данных без использования компьютерной программы; такой способ проработки материалов лекции покажет, правильно ли Вы поняли идею алгоритма;
- изучите дополнительные учебные материалы, рекомендованные преподавателем;
- попытайтесь ответить на контрольные вопросы, которыми, как правило, заканчиваются разделы учебных пособий или учебников;
- если после выполненной работы Вы считаете, что материал освоен не полностью, сформулируйте вопросы и задайте их преподавателю.

Методические указания к ведению конспектов лекций. Лекции по дисциплине проводятся с использованием слайдов. Но это не означает, что лекцию можно просто слушать. Ведение конспектов значительно повышает качество последующей проработки лекционного материала. В силу специфики дисциплины на слайдах лекций очень много алгоритмов, кодов программ, примеров демонстрации работы изучаемых алгоритмов. Но этот материал может быть бесполезен, если Вы не делаете записи в течение лекции, потому что в большинстве случаев, комментарии по представленным на слайдах примерам, лектор выполняет в устной форме.

Можно рекомендовать распечатывать слайды перед лекцией и вести конспект непосредственно на бумажном варианте слайд-презентации.

Одной из форм текущего мониторинга уровня знаний по дисциплине являются контрольные работы. В первом и втором семестре изучения дисциплины проводятся контрольные работы двух типов: тестовые опросы на лекции и контрольные работы, в которых студентам необходимо применить полученные знания на практике. Выполнение выше перечисленных действий поможет подготовиться и к выполнению контрольных работ. В приложении А указаны темы контрольных работ для студентов очной формы обучения и приведены примерные варианты.

Самостоятельная работа по подготовке к лабораторным работам и практическим занятиям по дисциплине состоит в изучении методических материалов по темам соответствующих видов аудиторных занятий.

Рекомендуется перед выполнением лабораторной работы изучить лекционный и методический материал по теме занятия, ознакомиться с алгоритмами, реализацию которых необходимо выполнить во время проведения занятия. Обратите особое внимание на порядок выполнения работы. Поскольку конечным результатом всех лабораторных работ является компьютерная программа, самостоятельно разработайте структурную схему будущей программы, выполните заготовку проекта, подготовьте самостоятельно тестовые данные. Если при подготовке к занятию остались нерешенные вопросы, обратитесь за консультацией к преподавателю.

3.3 Выполнение домашних заданий и контрольных работ

3.3.1 Общие положения

Выполнение домашних заданий для студентов очной формы обучения и выполнение контрольных работ для студентов заочной формы обучения – это полностью самостоятельный вид деятельности студента. Темами домашних заданий и контрольных работ являются темы дисциплины, не охваченные циклом лабораторных работ и практических занятий.

Выполнение домашнего задания состоит в написании программы по индивидуальному варианту. Обучающиеся самостоятельно разрабатывают алгоритм решения задания, реализуют разработанный алгоритм на языке программирования Си, отлаживают и тестируют написанную программу.

Защита домашнего задания проходит в сроки, установленные преподавателем. Процедура защиты представляет собой индивидуальное собеседование с преподавателем, примерный сценарий которого представлен ниже:

- комментирование студентом кода и логики написанной программы;
- ответы на вопросы преподавателя по коду и логике программы;
- комментирование студентом тестовых данных;
- демонстрация работы программы и пояснение полученных результатов.

Выполнение студентами заочной формы обучения контрольных работ аналогично выполнению домашних заданий студентами очной формы обучения. Контрольные работы выполняются самостоятельно, во время семестра, а защищаются перед преподавателем во время экзаменационных сессий, как это предусмотрено учебным планом дисциплины. Допускается общение с преподавателем во время семестра посредством электронной почты – выполненные контрольные работы могут быть высланы на предварительную проверку.

3.3.2 Домашнее задание «Подготовка к лабораторной работе «Создание консольного приложения в среде DEV-C++. Ввод-вывод информации»

Домашнее задание состоит в самостоятельном изучении среды программирования DEV-C++. Описание порядка действий при создании проекта представлено в [1] на стр. 28 – 41. При выполнении домашнего задания рекомендуется создать проект на языке Си, написать небольшую программу, выполнить компиляцию и запуск проекта. Выполнение этого домашнего задания поможет качественно и своевременно выполнить первую лабораторную работу дисциплины.

3.3.3 Домашнее задание «Целочисленная арифметика»

Домашнее задание формирует навыки работы с основными типами данных языка Си и знакомит обучающихся со способами вывода информации в языке Си. При выполнении рекомендуется изучить теоретический материал, изложенный в [1], стр. 77 – 78. Алгоритмы разбора целых чисел рассматриваются в [5], стр. 9 – 15.

Примерный вариант

а) Вывести на экран число e (основание натурального логарифма) с точностью до десятых.

б) Дано двузначное число. Найти число единиц числа.

3.3.4 Домашнее задание «Условные алгоритмы»

Домашнее задание формирует навыки работы с конструкцией проверки условия в языке Си. При выполнении рекомендуется изучить теоретический материал, изложенный в [1], стр. 87 – 89 и [3], стр. 14 – 17.

Примерный вариант

Даны две точки: $A(x_1, y_1)$ и $B(x_2, y_2)$. Напишите программу, определяющую, какая из точек находится ближе к началу координат.

3.3.5 Домашнее задание «Программирование итерационных алгоритмов»

Домашнее задание формирует навыки работы с конструкциями циклов в языке Си. При выполнении рекомендуется изучить теоретический материал, изложенный в [1], стр. 91 – 98.

Примерный вариант

Дано натуральное n , действительное число x . Вычислить:
$$\sum_{i=1}^n \frac{x + \cos(ix)}{2^i}.$$

3.3.6 Домашнее задание «Сортировка массивов»

Домашнее задание формирует навыки реализации простых сортировок на месте. При выполнении рекомендуется изучить теоретический материал, изложенный в [3], стр. 166 – 170, в [4], глава 2, стр. 69 – 81.

Примерный вариант.

Напишите программу, сортирующую массив методом обмена и вычисляющую количество сравнений, выполняемых алгоритмом.

3.3.7 Домашнее задание «Обработка матриц»

Домашнее задание формирует навыки работы с двумерными массивами языка Си. При выполнении рекомендуется изучить теоретический материал, изложенный в [1], стр. 124 – 132.

Примерный вариант

Напишите программу, заполняющую матрицу $n \times n$ (значение n вводить с клавиатуры) по правилу, которое представлено на примере матрицы 5×5 :

```
1 0 0 0 1
0 1 0 1 0
0 0 1 0 0
0 1 0 1 0
1 0 0 0 1
```

3.3.8 Домашнее задание «Машинное представление графов»

Цель домашнего задания – показать взаимосвязь двух дисциплин, информатики и дискретной математики. Для выполнения домашнего задания будет полезен справочный материал из разделов «Массивы» и «Матрицы» пособия [1], стр. 114 – 132, материал по теории графов пособия [3], стр. 250 – 251 и пособия [6], стр. 6 – 20.

Примерный вариант

Запишите программу получения матрицы смежности орграфа по заданной матрице инцидентности.

3.3.9 Домашнее задание «Текстовые файлы»

Цель домашнего задания – закрепление навыков работы с текстовыми файлами. Для выполнения задания необходимо изучить теоретический материал, изложенный в [1]. В главе 9 пособия описаны стандартные функции языка Си для работы с текстовыми файлами (стр. 144 – 155).

Примерный вариант

Напишите программу, читающую из текстового файла массив целых чисел произвольной размерности. Найдите среднее арифметическое элементов массива и допишите найденное значение в исходный текстовый файл с комментарием «Среднее арифметическое элементов массива – <найденное значение>».

3.3.10 Контрольная работа «Основы языка Си»

Контрольная работа «Основы языка Си» для студентов заочной формы обучения состоит из четырех заданий, которые необходимо выполнить в форме программы (или отдельных программ) на языке Си. Задания контрольной работы соответствуют темам «Функции», «Структурные переменные», «Текстовые файлы» и «Двоичные файлы». При решении контрольной работы приветствуется создание минимального пользовательского интерфейса – при написании программы используйте строки приглашения, оформляйте вывод полученных значений с соответствующими пояснениями. Перед каждым выполненным заданием из контрольной работы в тексте программы обязательно должны быть написаны комментарии, содержащие следующую информацию – фамилия, имя, отчество студента, формулировка задания, словесное описание алгоритма решения задания. Методические материалы, которые могут быть полезны для решения заданий: [1], стр. 59 – 72, 103 – 113, 144 – 166.

Примерный вариант

Задание 1. Заданы массивы чисел $X [0.. n]$ и $Y[0..m]$. Написать программу, определить значение переменной z . Исходные данные и результат напечатать с пояснительным текстом. Решение задачи оформить с использованием функций. Значения n и m задавать с клавиатуры, значения элементов массива задавать случайным образом.

В формулах расчета u использованы следующие условные обозначения:

$A1(X)$ – сумма элементов массива X ;

$A2(X)$ – сумма положительных элементов массива X ;

$A3(X)$ – сумма отрицательных элементов массива X ;

$A4(X)$ – количество нулевых элементов массива X ;

$A5(X)$ – сумма максимального и минимального элементов массива X ;

$A6(X)$ – среднее арифметическое значение элементов массива X ;

$A7(X)$ – произведение абсолютных значений элементов массива X ;

$A8(X)$ – корень квадратный из суммы положительных элементов массива X ;

$A9(X)$ – натуральный логарифм из суммы абсолютных значений элементов массива X ;

$A10(X)$ – сумма корней квадратных из положительных элементов массива X ;

$M1(X)$ – количество элементов массива X , значения которых меньше $A1$;

$M2(X)$ – количество отрицательных элементов массива X ;

$M3(X)$ – количество элементов массива X , значения которых больше $A6$;

$M4(X)$ – количество элементов массива X , значения которых меньше $A6$;

$M5(X)$ – количество элементов массива X , значения которых больше $A8$.

$$z = \begin{cases} \frac{A4(x) + 2.8 * 10^{-3} * A4(x)}{M1(y) + A4(y)}, & \text{если } M1(x) < 2, \\ 0, & \text{в противном случае.} \end{cases}$$

Задание 2. Исходные данные хранятся в файле

№ рейса	Пункт назначения	Дата	Количество свободных мест
---------	------------------	------	---------------------------

Составить список номеров рейсов и наличия свободных мест по входным данным: пункту назначения и дате. Использовать структурные переменные. Решение задания оформить с использованием функций.

Задание 3. В текстовом файле хранится произвольное количество чисел. Написать программу, которая считывает информацию из файла и находит максимальное число (текстовый файл предварительно создать в блокноте).

Задание 4. Создать файл вещественных чисел, записать в него матрицу вещественных чисел A размерности $m * n$ ($m < n$). Не считывая матрицу в память, реорганизовать файл, сделав матрицу квадратной ($n * n$) путем удаления "лишних" столбцов, начиная с заданного (номер столбца вводить с клавиатуры). Полученному файлу дать новое имя. Напечатать исходную и полученную матрицы с указанием имен файлов.

3.4 Самостоятельное изучение тем теоретической части курса

3.4.1 Функции

Перечень вопросов, подлежащих изучению

1. Синтаксис написания функций в языке Си.
2. Объявление и вызов функций.
3. Локальные переменные.
4. Организация выхода из функции.
5. Передача параметров по ссылке.
6. Рекурсивные функции.

Методические рекомендации по изучению

При изучении темы старайтесь следовать списку вопросов, представленному выше. Процесс изучения будет более эффективным, если Вы параллельно с изучением темы будете выполнять задания практически. Практическую часть самостоятельной работы можно начать с выполнения примеров, описанных в пособии. После реализации примеров попробуйте оформить в виде функций решенные ранее задачи.

Обратите особое внимание на такие понятия, как возвращаемое значение функции, аргументы (параметры функции).

При реализации рекурсивных функций помните – рекурсивная реализация всегда может быть заменена реализацией с использованием цикла.

Рекомендуемые источники

Пермякова, Н. В. Информатика и программирование: Учебное пособие [Электронный ресурс] / Н.В. Пермякова – Томск: ТУСУР, 2016. – 188 с. – Режим доступа: <https://edu.tusur.ru/publications/7678>, стр. 103 – 113.

Подбельский, В.В. Курс программирования на языке Си [Электронный ресурс] : учебник / В.В. Подбельский, С.С. Фомин. – Электрон. дан. – Москва : ДМК Пресс, 2012. – 384 с. – Режим доступа: <https://e.lanbook.com/book/4148> . – Загл. с экрана. Стр. 176 – 238.

3.4.2 Сложные типы данных

Перечень вопросов, подлежащих изучению

1. Структуры.
2. Объединения.
3. Перечисления.

Методические рекомендации по изучению

При изучении темы обратите особое внимание на семантику изучаемых типов данных. Определите для себя типы задач, в которых могут использоваться такие переменные. Параллельно с изучением теоретического материала попробуйте сформулировать задачи, решение которых невозможно без использования этих данных. Обязательно попробуйте набрать и выполнить компиляцию примеров, приведенных в учебном пособии.

Рекомендуемые источники

Пермякова, Н. В. Информатика и программирование: Учебное пособие [Электронный ресурс] / Н.В. Пермякова – Томск: ТУСУР, 2016. – 188 с. – Режим доступа: <https://edu.tusur.ru/publications/7678>, стр. 67 – 69.

Подбельский, В.В. Курс программирования на языке Си [Электронный ресурс] : учебник / В.В. Подбельский, С.С. Фомин. – Электрон. дан. – Москва : ДМК Пресс, 2012. – 384 с. – Режим доступа: <https://e.lanbook.com/book/4148> . – Загл. с экрана. Стр. 239 – 282.

3.4.3 Текстовые файлы

Перечень вопросов, подлежащих изучению

1. Описание файловых переменных.
2. Режимы доступа к файлу.
3. Открытие и закрытие файла.
4. Способы чтения информации из текстового файла.
5. Способы записи информации в текстовый файл.

Методические рекомендации по изучению

При изучении этой темы следуйте списку вопросов, представленному выше. Для более полного владения темой найдите описание структуры FILE. Запомните существующие режимы доступа к файлу. Обратите внимание, язык программирования Си предоставляет самые разнообразные варианты чтения информации из файла. Отметьте для себя, каким образом можно организовать проверку ошибок чтения или записи в файл. Не забывайте про практическую реализацию приведенных в учебных пособиях примеров.

Рекомендуемые источники

Пермякова, Н. В. Информатика и программирование: Учебное пособие [Электронный ресурс] / Н.В. Пермякова – Томск: ТУСУР, 2016. – 188 с. – Режим доступа: <https://edu.tusur.ru/publications/7678>, стр. 144 – 155.

Подбельский, В.В. Курс программирования на языке Си [Электронный ресурс] : учебник / В.В. Подбельский, С.С. Фомин. – Электрон. дан. – Москва : ДМК Пресс, 2012. – 384 с. – Режим доступа: <https://e.lanbook.com/book/4148> . – Загл. с экрана. Стр. 283 – 321.

3.4.4 Двоичные файлы

Перечень вопросов, подлежащих изучению

1. Способы чтения информации из двоичного файла.
2. Способы записи информации в двоичный файл.
3. Организация прямого доступа в двоичных файлах.

Методические рекомендации по изучению

Изучите функции языка Си, позволяющие считывать информацию из файла и записывать ее в файл. Обратите внимание на многообразие функций языка для организации прямого доступа к файлу. Выделите для себя различия между текстовыми и двоичными файлами. Параллельно с изучением справочного материала попробуйте написать функции записи заданной информации в двоичный файл. Попытайтесь открыть двоичный файл в текстовом редакторе. Найдите объяснение увиденной информации.

Рекомендуемые источники

Пермякова, Н. В. Информатика и программирование: Учебное пособие [Электронный ресурс] / Н.В. Пермякова – Томск: ТУСУР, 2016. – 188 с. – Режим доступа: <https://edu.tusur.ru/publications/7678>, стр. 156 – 166.

Подбельский, В.В. Курс программирования на языке Си [Электронный ресурс] : учебник / В.В. Подбельский, С.С. Фомин. – Электрон. дан. – Москва : ДМК Пресс, 2012. – 384 с. – Режим доступа: <https://e.lanbook.com/book/4148> . – Загл. с экрана. Стр. 322 – 334.

3.4.5 Динамические структуры

Перечень вопросов, подлежащих изучению

1. Линейные однонаправленные списки.
2. Список – стек, список – очередь.
3. Добавление элементов в список.
4. Удаление элемента из списка.
5. Создание сортированных списков.

Методические рекомендации по изучению

Изучение теоретического материала по этой теме желательно сопровождать зарисовками представления списков на бумаге. Понимание физического смысла создаваемой динамической структуры – основная задача самостоятельной работы над этой темой и успешное освоение темы напрямую зависит от решения этой задачи. При изучении рекомендуемых источников помните, что программирование – творческий процесс, поэтому одна и та же задача в разных пособиях может иметь различные варианты реализации в виде программного кода.

Рекомендуемые источники

Потопахин, В. Искусство алгоритмизации [Электронный ресурс] / В. Потопахин. – Электрон. дан. – Москва : ДМК Пресс, 2011. – 320 с. – Режим доступа: <https://e.lanbook.com/book/1269>. – Загл. с экрана. Стр. 224 – 236.

Вирт, Н. Алгоритмы и структуры данных. Новая версия для Оберона [Электронный ресурс] : учебное пособие / Н. Вирт. – Электрон. дан. – Москва : ДМК Пресс, 2010. – 272 с. – Режим доступа: <https://e.lanbook.com/book/1261>. – Загл. с экрана. Стр. 175 – 190.

3.5 Подготовка к экзамену

Студенты дневной формы обучения сдают экзамен по дисциплине в первом и втором семестре изучения. Выполнение всех видов самостоятельных работ, лабораторных работ, посещение практических и лекционных занятий – гарантия успешной сдачи экзамена.

Для подготовки к экзамену рекомендуется повторить темы, вынесенные на экзамен. При подготовке обращайтесь не только к конспектам лекций, но и к рекомендованным преподавателем источникам. Организуйте план повторения материала таким образом, чтобы каждый день прорабатывать примерно одинаковый по объему материал. Изучая учебники и учебные пособия, отвечайте на контрольные вопросы. Прорешайте задачи примерного билета. Если после изучения материала Вы не смогли найти ответы на какие-либо вопросы – посетите консультацию перед экзаменом, кроме ответов на вопросы по теме экзамена на консультации освещаются организационные вопросы проведения экзамена время начала экзамена, время проведения экзамена, план проведения экзамена и т.д..

Пример экзаменационного билета (1 семестр)

Билет 1

Алфавит языка Си. Правила формирования имен идентификаторов. Приведите примеры верно и неверно сформированных имен идентификаторов.

Напишите программу, которая запрашивает с клавиатуры размерность массива, заполняет массив случайными значениями из интервала $[-5, 10]$ и находит сумму максимального и минимального элементов.

Напишите программу, которая запрашивает с клавиатуры размерность квадратной матрицы n , заполняет матрицу по правилу:

1	2	3	4	...	n
0	1	2	3	...	$n-1$
0	0	1	2		
0	0	0	1
0	0	0	0	1	2
0	0	0	0	0	1

и сохраняет ее в текстовом файле с заданным именем. Формирование матрицы и сохранение в текстовом файле оформить в виде функции. Параметры функции – размерность матрицы и имя файла.

4 РЕКОМЕНДУЕМЫЕ ИСТОЧНИКИ

1. Пермякова, Н.В. Информатика и программирование: Учебное пособие [Электронный ресурс] / Н.В. Пермякова – Томск: ТУСУР, 2016. – 188 с. – Режим доступа: <https://edu.tusur.ru/publications/7678>,
2. Мещеряков, Р.В. Методы программирования [Электронный ресурс] : учебно-методическое пособие / Р.В. Мещеряков. – Электрон. дан. – Москва : ТУСУР, 2007. – 237 с. – Режим доступа: <https://e.lanbook.com/book/11631>. – Загл. с экрана.
3. Потопахин, В. Искусство алгоритмизации [Электронный ресурс] / В. Потопахин. – Электрон. дан. – Москва : ДМК Пресс, 2011. – 320 с. – Режим доступа: <https://e.lanbook.com/book/1269>. – Загл. с экрана.
4. Вирт, Н. Алгоритмы и структуры данных. Новая версия для Оберона [Электронный ресурс] : учебное пособие / Н. Вирт. – Электрон. дан. – Москва : ДМК Пресс, 2010. – 272 с. – Режим доступа: <https://e.lanbook.com/book/1261>. – Загл. с экрана.
5. Златопольский, Д.М. Подготовка к ЕГЭ по информатике. Решение задач по программированию [Электронный ресурс] : учебное пособие / Д.М. Златопольский. – Электрон. дан. – Москва : ДМК Пресс, 2017. – 252 с. – Режим доступа: <https://e.lanbook.com/book/100911>. – Загл. с экрана.
6. Асанов, М.О. Дискретная математика: графы, матроиды, алгоритмы [Электронный ресурс] : учебное пособие / М.О. Асанов, В.А. Баранский, В.В. Расин. – Электрон. дан. – Санкт-Петербург : Лань, 2010. – 368 с. – Режим доступа: <https://e.lanbook.com/book/536>. – Загл. с экрана.

ПРИЛОЖЕНИЕ А

Темы и примерные варианты контрольных работ

1. Синтаксис и алфавит языка Си

Вариант 1		
<p>1. Выберите тип передачи управления, использующийся в структурном программировании:</p> <ul style="list-style-type: none"> • безусловная передача • условная передача • функционально-зависимая передача 	<p>Посчитайте количество лексем в представленном фрагменте программы:</p> <pre>float x,y,z; printf(" -->");</pre>	<p>Выберите ключевые слова Си:</p> <ul style="list-style-type: none"> • if • while • main • factorial • integer

2. Основные типы данных. Условный оператор

Вариант 1		
<p>Опишите переменную x как указатель на тип float.</p>	<p>Что будет храниться по адресу y, если выполнится фрагмент программы:</p> <pre>int *y; int k = 12; y = &k;</pre>	<p>Что будет выведено на экран при выполнении следующего фрагмента программы:</p> <pre>int x = 7; int y = 9; int z = 0; if (x>y) { z = y*2; y = x*4; } else { z = x*2; x = y+x;} printf (" %d %d %d", x,y,z);</pre>

3. Циклы в языке Си

Вариант 1 Фамилия _____ гр_____		
<p>1. Используя цикл while, запишите фрагмент программы, который выводит на экран числа 2 5 8 11 14 17 20. Описание использованных переменных обязательно.</p>	<p>2. Что будет выведено на экран при выполнении следующего фрагмента программы:</p> <pre>int i = 25; do{ printf("%3d",i); i-=2; } while(i>=13);</pre>	<p>3. Запишите фрагмент программы, решающей следующую задачу (используйте цикл for):</p> <p>Вывести на экран числа от 0 до 12 с шагом 0.25. Фрагмент обязательно должен содержать описания использованных переменных.</p>

4. Программирование циклических процессов

Вариант 1

1. Напечатать таблицу кубов целых чисел от a до $a+10$ в виде:

```
*****
* a *   куб                *
*****
*   *                       *
*   *                       *
*   *                       *
*****
```

Значение a вводить с клавиатуры.

2. Последовательность чисел a_0, a_1, a_2, \dots образуется по закону: $a_0 = 1, a_k = ka_{k-1} + 1/k$ ($k = 1, 2, \dots$). Дано натуральное число n . Получить a_1, a_2, \dots, a_n .

5. Массивы в языке Си

Вариант 1 Фамилия _____ гр _____		
<p>1. Что будет выведено на экран при выполнении следующей программы:</p> <pre>int main(int argc, char *argv[]) { system("chcp 1251"); int x[10] = {2,7,6,1,9,5,8,3,4,0}; int k = 0; for (i=0;i<10;i++) if (x[i]%2==0) printf("%3d",i); printf("\n"); system("PAUSE"); return EXIT_SUCCESS; }</pre>	<p>2. Что будет выведено на экран при выполнении следующей программы:</p> <pre>int main(int argc, char *argv[]) { system("chcp 1251"); int x[10] = {2,7,6,1,9,5,8,3,4,0}; int k = x[0]; for (i=1;i<10;i++) if (x[i]>k) k = x[i]; printf("%3d",k); printf("\n"); system("PAUSE"); return EXIT_SUCCESS; }</pre>	<p>3. Что будет выведено на экран при выполнении следующей программы:</p> <pre>int main(int argc, char *argv[]) { system("chcp 1251"); int x[10] = {2,7,6,1,9,5,8,3,4,0}; int i,j,k; int m = 3; for (j=0;j<m;j++){ k = x[9]; for (i=9;i>0;i--){ x[i] = x[i-1]; x[0] = k; } for (i=1;i<10;i++) printf("%3d",x[i]); printf("\n"); system("PAUSE"); return EXIT_SUCCESS; }</pre>

6. Матрицы в языке Си

Вариант 1		
<p>Что будет выведено на экран при выполнении следующей программы:</p> <pre>int main(int argc, char *argv[]) {int A[5][5]={1,2,3,4,5, 6,7,8,9,10, 11,12,13,14,15, 16,17,18,19,20, 21,22,23,24,25}; int i,j,k,n=5; k = A[0][0]; for(i=0;i<n;i++) for(j=0;j<=i;j++) if (A[i][j]>k) k=A[i][j]; printf("%d\n",k); system("PAUSE"); return 0; }</pre>	<p>Что будет выведено на экран при выполнении следующей программы:</p> <pre>int main(int argc, char *argv[]) {int A[5][5]={1,2,3,4,5, 6,7,8,9,10, 11,12,13,14,15, 16,17,18,19,20, 21,22,23,24,25}; int i,j,n=5,p; int S[5] = {0,0,0,0,0}; p = 0; for(i=0;i<n;i++){ for(j=0;j<n;j++){ S[i]+=A[i][j]; if (S[p]<=S[i])p = i; } for(i=0;i<n;i++) printf("%d ", A[p][i]); printf("\n"); system("PAUSE"); return 0; }</pre>	<p>Сколько байт памяти занимает переменная <code>struct FIO</code></p> <pre>struct FIO{ char Name[n]; float ball[m]; int k;} при n = 5, m =4</pre>

7. Программирование обработки матриц и массивов

Вариант 1

Реализация второй и третьей задачи без использования дополнительного массива – 2 балла, с использованием дополнительного массива – 1 балл. В третьей задаче не допускается изменять матрицу на этапе определения элементов.

a. С клавиатуры вводится размерность массива. Сам массив задается случайным образом. Найти самую длинную последовательность одинаковых элементов.

b. С клавиатуры вводится размерность массива. Сам массив задается случайным образом. Изменить массив, вставив после каждого минимального элемента новый элемент, значение которого равно значению последнего элемента

c. С клавиатуры вводится количество строк и столбцов матрицы. Элементы матрицы задаются случайным образом. Изменить матрицу, удалив строки с первым положительным элементом.