ЛАБОРАТОРНАЯ РАБОТА №3

Цель работы:

Приобрести умения и практические навыки по созданию динамических массивов и работе с указателями; приобрести умения и практические навыки для работы с функциями.

Теоретическая часть:

Все определенные в программе данные хранятся в памяти по определенному адресу. И указатели позволяют напрямую обращаться к этим адресам и благодаря этому манипулировать данными. *Указатели* представляют собой объекты, значением которых служат адреса других объектов или функций. Указатели — это неотъемлемый компонент для управления памятью в языке Си.

Значением переменной типа указатель является адрес области памяти. Существует несколько видов указателей: на объект, на функции, на void.

Указатель на объект имеет синтаксис *тип *имя;* и содержит адрес значения определенного типа.

Для вывода значения указателя (адреса объекта) используется специальный спецификатор %p.

Над указателями можно производить различные операции:

– & – получение адреса – применима только к величинам, которые имеют имя и размерность и размещаются в оперативной памяти. Таким образом, нельзя получить адрес скалярного выражения, неименованной константы, регистровой переменной;

– * – разыменование (разадресация) – обращение к ячейке памяти по адресу.

Также возможны различные действия над указателями:

– ++, -- – увеличение или уменьшение адреса на длину элемента данного типа;

– увеличение / уменьшение адреса – аналогично предыдущему действию, но вместо единицы адрес увеличивается на i длин элементов;

– разность указателей – сколько значений такого типа разделяют два адреса: из первого адреса вычитается второй и делится на длину значения данного типа. Суммирование указателей не определено.

– сравнение – адреса равны, если указывают на один объект.

Указатели на разные типы имеют разные типы соответственно. Эти типы несовместимы.

Указатель на void не связан с определенным типом, то есть ему можно присвоить значение любого указателя. Он применяется, когда конкретный тип объекта, адрес которого нужно хранить, неизвестен. Синтаксис следующий: *void * p;*

Данный указатель совместим с указателями любых типов.

Имя массива – адрес его первого элемента, или константный указатель на первой элемент. С массивами можно работать через указатель. Для этого необходимо:

– объявить массив;

– объявить указатель на тип элемента массива;

– присвоить указателю имя массива;

– обратиться к элементам массива через указатель.

```
int A[5] = {1; 2; 3, 4, 5}, B[5];
int *pA, *pB;
pA = &A[0];
pB = B;
for (i = 0; i < 5; i++0) {
     *pB = *pA;
      pB++;
      pA++;
} // либо *pB++ = *pA++;
```

Существуют ситуации, когда память под переменные нельзя выделить статически. В таких случаях память выделяется динамически (в процессе выполнения программы). Для динамического расширения памяти есть функция, которая содержится в библиотеке *malloc.h.*

Память выделяется при помощи функции *malloc (размер требуемого участка памяти)*. Она возвращает указатель на void – адрес начала выделенной области. Таким образом, обращаться к динамическим переменным приходится через указатель.

```
имя переменной = (тип *) malloc (sizeof (тип));
```

Освобождение выделенного участка памяти производится при помощи команды *free (адрес).* Динамические массивы не инициализируются при объявлении, а также не обнуляются. Их преимущество состоит в том, что размерность может определяться в ходе выполнения программы. Доступ к их элементам выполняется так же, как и к элементам статических массивов.

Создание динамического массива:

```
int *m;
int n;
printf("\n Количество элементов: ");
scanf_s("%d", &n);
m = (int *) malloc (n * sizeof (int));
```

Теперь обращаться к элементам данного массива можно также по индексам, например, *m[2];*

Создание двумерных динамических массивов (из n строк и k столбцов):

```
int **m, i, j, k, n;
m = (int **) malloc (n * sizeof (int));
for (i = 0; i < n; i++)
        m[i] = (int *) malloc (k * sizeof (int));
```

Функцией называется последовательность описаний и операторов, которая выполняет какое-либо законченное действие и, возможно, возвращает результат в вызывающую программу. Любая функция должна быть объявлена и определена. Объявлений может быть несколько, определение же только одно.

Объявление функции должно быть появиться до ее вызова. Синтаксис объявления функции:

```
тип имя (список передаваемых параметров);
```

Синтаксис определения функции:

```
тип имя (список параметров) {тело функции;};
```

Имя функции – правильный идентификатор. *Тип возвращаемого значения* может быть любым, кроме массива и функции, но может быть указателем на массив и функцию. Если функция не возвращает значение, она должна иметь тип void. *Список параметров* определяет значения, которые передаются в функцию при ее вызове. Элементы списка параметров имеют вид: *тип имя*, перечисление разделяется запятой.

В объявлении, определении и при вызове функции типы и порядок следования параметров должны совпадать. Имена могут быть различны, так как в объявлении имена игнорируются, вызываться может с разными параметрами.

Тип возвращаемого значения вместе с типами параметров определяется типом функции. Все величины, описанные внутри функции, и параметры являются локальными по отношению к функции. Их область действия ограничена самой функцией.

Возврат результатов в вызывающую программу может быть реализован следующим образом:

```
return [выражение];
```

Выражение преобразуется к типу возвращаемого значения и возвращается в точку вызова, поэтому вызов функций может выполняться внутри выражения, то есть функция может рассматриваться в качестве аргумента выражения. Если тип функции void, то выражение нужно опустить. Если тип функции void и оператор return по смыслу находится перед закрывающей фигурной скобкой, то его можно опустить. Также операторов return может быть несколько, каждый из которых завершает работу программы.

Формальные параметры – параметры, указываемые в заголовке функции.

Фактические параметры – параметры, указываемые при вызове функции.

Передача параметров в функцию выполняется через стек. Существует два способа передачи параметров:

– по значению – в стек помещается копия значения параметра. По завершению работы стек очищается, измененное значение параметра функции не передается в вызывающую программу. Фактически, параметром, соответствующим формальному параметру, передаваемому по значению, может быть выражение (переменная или константа, в частном случае);

– по адресу – в стек передается адрес аргумента, а функция по этому адресу обращается к параметру. Все изменения, которым подвергается параметр, сохраняются после возврата управления в вызывающую программу. Фактически, параметром, соответствующим формальному параметру, передаваемому по адресу, может быть переменная или левое допустимое выражение.

При передаче параметра по адресу производятся следующие действия:

– в заголовке формальный параметр объявляется как указатель на тип аргумента;

– в теле функции для обращения к аргументу используется операция разыменования (*);

– при вызове функции используется операция взятия адреса.

Задание 1. Числа M и N в данном задании вводить с клавиатуры. Диапазон генерации чисел также принимать с клавиатуры.

1. Даны два массива: A[N] и B[M]. Заполнить массивы случайными числами. Пользуясь указателями создать третий массив, в котором нужно собрать элементы обоих массивов.

2. Даны два массива: A[N] и B[M]. Заполнить массивы случайными числами. Пользуясь указателями создать третий массив, в котором нужно собрать элементы массива A, которые не включаются в B.

3. Даны два массива: A[N] и B[M]. Заполнить массивы случайными числами. Пользуясь указателями создать третий массив, в котором нужно собрать элементы массива B, которые не включаются в A.

4. Даны два массива: A[N] и B[M]. Заполнить массивы случайными числами. Пользуясь указателями создать третий массив, в котором нужно собрать элементы массивов A и B, которые не являются общими для них.

5. Даны два массива: A[N] и B[M]. Заполнить массивы случайными числами. Пользуясь указателями создать третий массив, в котором нужно собрать элементы массивов A и B, которые являются общими для них.

6. Создать одномерный динамический массив из N целых чисел. Заполнить массив случайными числами. Удвоить значения элементов массива, которые стоят на четных местах.

7. Создать одномерный динамический массив из N целых чисел. Заполнить массив случайными числами. Удвоить значения элементов массива, которые стоят на нечетных местах.

8. Создать одномерный динамический массив из N целых чисел. Заполнить массив случайными числами. Удвоить значения элементов массива, которые стоят на местах, кратных вводимому с клавиатуры числу.

9. Создать одномерный динамический массив из N целых чисел. Заполнить массив случайными числами. Найти в нем максимальный и минимальный элементы и вывести их на экран.

10. Создать одномерный динамический массив из N целых чисел.

Заполнить массив случайными числами. Заменить отрицательные числа их модулями.

11. Создать одномерный динамический массив из N целых чисел. Заполнить массив случайными числами. Заменить четные числа числами в два раза меньше.

12. Создать одномерный динамический массив из N целых чисел. Заполнить массив случайными числами. Заменить числа, имеющие натуральный корень, этим корнем.

13. Создать одномерный динамический массив из N целых чисел. Заполнить массив случайными числами. Найти среднее арифметическое всех элементов и заменить все элементы, меньшие среднего, на само среднее значение.

14. Создать одномерный динамический массив из N целых чисел. Заполнить массив случайными числами. Перевернуть массив (развернуть его задом наперёд) без использования дополнительного массива.

15. Создать одномерный динамический массив из N целых чисел. Заполнить массив случайными числами. Определить количество уникальных значений в массиве.

16. Создать одномерный динамический массив из N целых чисел. Заполнить массив случайными числами. Определить количество значений в массиве, встречающихся более одного раза.

17. Создать одномерный динамический массив из N целых чисел. Заполнить массив случайными числами. Найти и вывести наиболее часто встречающийся элемент.

18. Создать одномерный динамический массив из N целых чисел. Заполнить массив случайными числами. Поменять местами первый положительный и первый отрицательный элементы.

Задание 2.

Функция main должна вызвать указанную функцию и вывести на экран полученный результат.

1. Написать функцию, которая возвращает в вызывающую программу площадь прямоугольника. Стороны прямоугольника необходимо вводить с клавиатуры. Тип параметров – вещественный.

2. Написать функцию, которая возвращает в вызывающую программу периметр треугольника. Стороны треугольника необходимо вводить с клавиатуры. Тип параметров – вещественный. Также необходимо предусмотреть проверку, возможно ли вообще построить треугольник с указанными сторонами.

3. Написать функцию, которая возвращает в вызывающую программу периметр прямоугольника. Стороны прямоугольника необходимо вводить с клавиатуры. Тип параметров – вещественный.

4. Написать функцию, которая возвращает в вызывающую программу площадь круга радиуса r. Значение радиуса необходимо вводить с клавиатуры. Тип параметров – вещественный.

5. Написать функцию, которая возвращает в вызывающую программу длину окружности радиуса r. Значение радиуса необходимо вводить с клавиатуры. Тип параметров – вещественный.

6. Написать функцию, которая возвращает в вызывающую программу объем параллелепипеда. Значения сторон параллелепипеда необходимо вводить с клавиатуры. Тип параметров – целый.

7. Написать функцию, которая возвращает в вызывающую программу количество секунд, которое содержится в t часах. Количество часов необходимо вводить с клавиатуры. Тип параметров – целый.

8. Написать функцию, которая возвращает в вызывающую программу количество часов, которое содержится в n днях. Количество дней необходимо вводить с клавиатуры. Тип параметров – целый.

9. Написать функцию, которая вычисляет факториал переданного ей целого числа. Параметр – целое число, вводится с клавиатуры. Тип параметров –

целый.

10. Написать функцию, которая определяет, является ли переданное ей целое число простым. Функция возвращает 1, если число простое, и 0 – в противном случае. Тип параметров – целый.

11. Написать функцию, которая возвращает сумму всех цифр переданного ей целого числа. Параметр – целое число, вводится с клавиатуры.

12. Написать функцию, которая возвращает корень квадратный из переданного ей вещественного числа. Использовать встроенную функцию sqrt() из библиотеки math.h.

13. Написать функцию, которая вычисляет значение n-го числа Фибоначчи. Параметр – целое число (порядковый номер), вводится с клавиатуры.

14. Написать функцию, которая возвращает значение n-й степени числа x. Параметры – вещественное число x и целое число n, вводятся с клавиатуры.

15. Написать функцию, которая переводит градусы по Цельсию в градусы по Фаренгейту. Параметр – вещественное число (градусы Цельсия), вводится с клавиатуры.

16. Написать функцию, которая возвращает в вызывающую программу сумму и произведение двух целых чисел. Числа необходимо вводить с клавиатуры. Результаты возвращать через параметры, передаваемые по адресу.

17. Написать функцию, которая возвращает в вызывающую программу квадрат и куб аргумента. Значение аргумента необходимо вводить с клавиатуры. Результаты возвращать через параметры, передаваемые по адресу.

18. Написать функцию, которая возвращает в вызывающую программу сумму и среднее арифметическое двух целых чисел. Числа необходимо вводить с клавиатуры. Результаты возвращать через параметры, передаваемые по адресу.

19. Написать функцию, которая возвращает в вызывающую программу наибольшее и наименьшее из трех целых чисел. Числа необходимо вводить с клавиатуры. Результаты возвращать через параметры, передаваемые по адресу.

20. Написать функцию, которая возвращает в вызывающую программу

площадь поверхности и объем куба. Значение стороны куба необходимо вводить с клавиатуры. Результаты возвращать через параметры, передаваемые по адресу.

21. Написать функцию, которая возвращает в вызывающую программу два целых числа, между которыми находится заданное вещественное число. Число необходимо вводить с клавиатуры. Результаты возвращать через параметры, передаваемые по адресу.

22. Написать функцию, которая вычисляет частное и остаток от деления двух целых чисел. Проверить деление на ноль перед выполнением вычислений. Число необходимо вводить с клавиатуры. Результаты возвращать через параметры, передаваемые по адресу.

23. Написать функцию, которая вычисляет синус и косинус переданного угла (в градусах). Угол вводится с клавиатуры. Использовать функции sin() и cos() из math.h. Результаты возвращать через параметры, передаваемые по адресу.

24. Написать функцию, которая вычисляет длину гипотенузы и площадь прямоугольного треугольника. Длины катетов вводятся с клавиатуры. Результаты возвращать через параметры, передаваемые по адресу.

25. Написать функцию, которая преобразует градусы по Цельсию в Фаренгейты и Кельвины. Значение температуры – вещественное число, вводится с клавиатуры. Результаты возвращать через параметры, передаваемые по адресу.

26. Написать функцию, которая находит наибольший общий делитель (НОД) и наименьшее общее кратное (НОК) двух чисел. Числа вводятся с клавиатуры. Результаты возвращать через параметры, передаваемые по адресу.

27. Написать функцию, которая возвращает корни квадратного уравнения (если они существуют). Коэффициенты уравнения вводятся с клавиатуры. Результаты возвращать через параметры, передаваемые по адресу.

28. Написать функцию, которая генерирует n случайных чисел в заданном диапазоне (min-max) и возвращает их сумму и среднее значение через параметры. Результаты возвращать через параметры, передаваемые по адресу.

Числа вывести на экран

29. Написать функцию, которая возвращает в вызывающую программу двоичное и восьмеричное представление заданного числа. Число необходимо вводить с клавиатуры. Результаты возвращать через параметры, передаваемые по адресу.

30. Написать функцию, которая возвращает значение логарифма и экспоненты для заданного числа. Число вводится с клавиатуры. Использовать функции log() и exp() из math.h. Результаты возвращать через параметры, передаваемые по адресу.

Вопросы к теоретическому материалу

1. Что называется указателем?
2. Что является значением переменной типа указатель?
3. Перечислите виды указателей в Си.
4. Укажите синтаксис указателя на объект.
5. Перечислите операции, которые можно производить над указателями.
6. Охарактеризуйте операцию получения адреса, производимую над указателями.
7. Охарактеризуйте операцию разыменования, производимую над указателями.
8. Перечислите действия, которые можно производить над указателями.
9. Охарактеризуйте нахождение разности указателей. Укажите, для чего это можно применять. Определено ли суммирование указателей?
10. Охарактеризуйте указатель на void.
11. Опишите последовательность действий при работе с массивами через указатель.
12. Какая функция применяется для динамического выделения памяти? В какой библиотеке она содержится?
13. Укажите, каким образом выделяется память под одну переменную при помощи функции malloc.
14. Укажите, при помощи какой команды происходит освобождение выделенного участка памяти.
15. Кратко опишите процесс создания динамического массива.
16. Что называется функцией?
17. Укажите синтаксис объявления функции.
18. Укажите синтаксис описания функции.
19. При помощи какой команды можно организовать возврат результатов в вызывающую программу?

20. Что называется формальными параметрами функции?

21. Что называется фактическими параметрами функции?

22. Какие два способа передачи параметров в функции существует?

23. Охарактеризуйте передачу параметров в функцию по значению.

24. Охарактеризуйте передачу параметров в функцию по адресу.

25. Перечислите действия, которые необходимо совершить при передаче параметра по адресу

ПРОЦЕСС СДАЧИ ЛАБОРАТОРНОЙ РАБОТЫ

По итогам выполнения каждой лабораторной работы студент:

1. Демонстрирует преподавателю правильно работающие программы;
2. Демонстрирует приобретенные теоретические знания, отвечая на пять вопросов по лабораторной работе;
3. Демонстрирует отчет по выполненной лабораторной работе, соответствующий всем требованиям.

Отчет по лабораторной работе оформляется по шаблону, представленному в приложении 1. Требования к отчету представлены в приложении 2.

ПРИЛОЖЕНИЕ 1. ШАБЛОН ОТЧЕТА

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Казанский национальный исследовательский технический университет им. А.Н. Туполева – КАИ»

Институт компьютерных технологий и защиты информации

Отделение СПО ИКТЗИ (Колледж информационных технологий)

ЛАБОРАТОРНАЯ РАБОТА №

по дисциплине

СИСТЕМНОЕ ПРОГРАММИРОВАНИЕ

Работу выполнил

Студент гр.43__

Фамилия И.О.

Принял

Преподаватель Григорьева В.В.

Казань, 2025 г.

1. **Цель работы.**

2. **Задание на лабораторную работу** – вставляется задание на лабораторную работу, соответствующее индивидуальному, выданному преподавателем, варианту студента.

3. **Результат выполнения работы** – формируется описание хода выполнения работы и вставляются скриншоты с результатами работы разработанных программ (скриншоты должны быть подписаны, например, *Рисунок 1. Начальное состояние программы* и т.п.).

4. **Листинг программы** – вставляется код разработанной программы

ПРИЛОЖЕНИЕ 2. ТРЕБОВАНИЯ К ОФОРМЛЕНИЮ ОТЧЕТА

Лист документа должен иметь книжную ориентацию, поля документа должны составлять: левое – 3 см, правое – 1,5 см, верхнее – 2 см, нижнее 2 см.

Нумерация страниц – внизу страницы по центру, первая страница не нумеруется

Междустрочный интервал – 1,5 (полуторный), отступ первой строки – 1,25.

Текст документа должен быть выполнен с использованием шрифта Times New Roman, размер – 14, выравнивание – по ширине. Заголовки выполняются тем же шрифтом, но размера 16, полужирное начертание, размещение – по центру.

Рисунки должны размещаться по центру, они нумеруются по порядку. Перед рисунком в тексте на него должна быть ссылка. Подпись рисунка должна располагаться по центру и быть выполнена шрифтом Times New Roman, размер – 12. Сначала происходит нумерация рисунка, а затем пишется его название.