

Министерство образования Республики Беларусь
г. Минск
Государственное учреждение образования
«Белорусский государственный университет информатики и
радиоэлектроники»

Лабораторная работа №1
“Статические массивы”
Учебная группа 313871

Выполнил: Колпаковский Н.С.
Проверил: Селезнев А.И.

2023 год

Цель работы: изучить правила составления текстов программ, научиться работать со статическими массивами. Изучить основные понятия языка С, типы данных, библиотеки, функции, переменные, константы, операции и выражения, вывод и ввод информации, операторы цикла и условий. Научиться работать с одномерными и двумерными массивами. Научиться компилировать и проверять работоспособность программы.

Алфавит языка С состоит из: прописных и строчных букв латинского алфавита, арабских цифр и специальных символов, смысл и правила, использования которых будут рассматриваться далее.

Используемые инструменты, при написании кода:

1. В языке С применяются данные двух категорий: простые (скалярные) и сложные (составные). К основным (базовым) типам данных относятся целый (*int*), вещественный (*float*, *double*) и символьный (*char*) типы. В свою очередь, данные целого типа могут быть короткими (*short*) и длинными (*long*), со знаком (*signed*) и беззнаковыми (*unsigned*). Атрибут *long* может использоваться и с типом *double* – длинное вещественное.

2. Операция присваивания (=). В результате выполнения этой операции переменная, стоящая слева от знака "=", принимает значение выражения, расположенного справа. Отличительной чертой операции присваивания в языке С является то, что она может быть использована в одном выражении более одного раза.

3. Операция инкрементирования (увеличения ++) и декрементирования (уменьшения --) значения операнда. Операнд должен быть целого, плавающего или адресного типа. Операнды целого или плавающего типа преобразуются путем сложения (++) или вычитания (--) единицы. Тип результата соответствует типу операнда. Операнд адресного типа инкрементируется или декрементируется размером объекта, который он адресует. Инкрементированный указатель адресует следующий объект, а декрементированный указатель – предыдущий.

4. Функции ввода/вывода *printf* и *scanf*, они отличаются только названием. Обе имеют управляющую строку и список аргументов. Функция *printf* использует имена переменных, константы и выражения, в то время как функция *scanf* только указатели на переменные. Формат спецификации преобразования аналогичен функции *printf*. В спецификации преобразования допустимы следующие символы:

- d* или *i* - ожидается ввод десятичного числа;
- o* - ожидается ввод восьмеричного целого числа;
- x* - ожидается ввод шестнадцатеричного целого числа;
- u* - ожидается ввод беззнакового числа;
- c* - ожидается ввод одиночного символа;
- s* - ожидается ввод символьной строки;
- e*, *f* или *g* - ожидается ввод вещественного числа;
- p* - ожидается ввод указателя (адреса) в виде шестнадцатеричного числа;
- n* – получается значение, равное числу прочитанных символов из входного потока на момент обнаружения %n;
- // – сканируется множество символов.

5. В языке С имеются три оператора, которые могут нарушить простой линейный характер выполнения программы. К ним относятся: *if* (если) *else* (иначе) (ветвления), *switch* (переключатель) и *goto* (безусловного перехода). Оператор ветвления предназначен для выбора в программе из нескольких возможных вариантов единственного варианта

продолжения вычислительного процесса. Выбор выполняется исходя из результатов анализа значения некоторого выражения.

Оператор **if** имеет следующую общую форму записи:

-**if**(проверка условия) {группа операторов 1 }

-[**else** {группа операторов 2 }]

-При выполнении оператора **if** сначала выполняется проверка условия.

-Если результат - истина (любое отличное от нуля значение), то выполняется группа операторов 1.

-Если результат анализа условия - ложь (равен 0), то выполняется группа операторов 2.

-Если слово **else** отсутствует, то управление передается на следующий после **if** оператор.

6. В языке C существуют три оператора организации циклов: **for** (для), **while** (пока) и **do ... while** (делать пока).

Оператор **for** формально записывается в следующем виде:

for(выражение1; выражение2; выражение3) тело цикла;

Тело цикла составляют одна либо некоторое подмножество инструкций, заключенных в фигурные скобки. В выражениях 1,2,3 фигурирует переменная, называемая управляющей. В операторе **for** устанавливается нижняя и верхняя граница изменения переменной цикла и величина (шаг) ее изменения. Каждое из выражений в круглых скобках разделены точкой с запятой. Первое выражение служит для инициализации управляющей переменной. Оно выполняется только один раз в начале выполнения цикла. Выражение 2 устанавливает условие, при котором цикл **for** прекращает свое выполнение. Проверка условия осуществляется перед каждым выполнением цикла. Выражение 3 задает приращение (уменьшение) управляющей переменной

Оператор цикла **do ... while**

В отличие от цикла **while** условие проверяется после каждой итерации (повтора). В общем виде цикл **do while** записывается в следующем виде:

do оператор; **while**(выражение);

или

do { оператор1; оператор2; ... оператор n;

} **while**(выражение);

Тело цикла **do ... while** всегда выполняется, по крайней мере, один раз.

Цикл прекращается, если выражение в скобках принимает ложное значение.

Исходные данные:

1. Написать программу для решения задачи: Заполнить массив натуральными числами, которые кратны 7 и не кратны 3.

2. Написать программу для решения задачи: В двумерном массиве записаны оценки нескольких учеников по учебным предметам. Определить номер ученика с максимальным средним баллом (если таких несколько, вывести первого).

3. Написать программу для решения задачи: Имеется два массива целых чисел: первый заполнен по возрастанию, второй - по убыванию. Объединить массивы в третий массив в порядке возрастания.

//Лабораторная работа №1, задание №1 Колпаковский Н.С. 12.12.2023

```
#include <stdio.h>
```

```
#include <locale.h>
```

```
int main() {
```

```
    start: setlocale(LC_ALL, "Russian");
```

```
    char answer_repeat;
```

```
    int i = 0;
```

```
    int arr[10];
```

```
    int sevenCount = 7;
```

```
    // Цикл while отвечает за заполнение массива числами, которые кратны 7 и не кратны 3
```

```
    while (i < 20) {
```

```
        // Заполняем массив числами, кратными 7 и не кратными 3
```

```
        if (sevenCount % 3 != 0) {
```

```
            arr[i]=sevenCount;
```

```
            i++;
```

```
        }
```

```
        sevenCount+=7;
```

```
    }
```

```
    // Выводим на экран все числа, кратные 7 и не кратные 3
```

```
    for (int k = 0; k < i; k++) {
```

```
        printf("%d ", arr[k]);
```

```
    }
```

```
    printf("Повторить? y/N");
```

```
    // Ввод согласия или не согласия на повторение, игнорируя
```

```
    scanf(" %c", &answer_repeat);
```

```
    if(answer_repeat == 'y' | answer_repeat == 'Y') {
```

```
        // Возврат к началу исполнения кода
```

```
        goto start;
```

```
    }
```

```
return 0;
```

```
}
```

```
7 14 28 35 49 56 70 77 91 98 112 119 133 140 154 161 175 182 196 203 Повторить? y/ny
7 14 28 35 49 56 70 77 91 98 112 119 133 140 154 161 175 182 196 203 Повторить? y/NY
7 14 28 35 49 56 70 77 91 98 112 119 133 140 154 161 175 182 196 203 Повторить? y/NY
7 14 28 35 49 56 70 77 91 98 112 119 133 140 154 161 175 182 196 203 Повторить? y/Nn
*** stack smashing detected ***: terminated
```

```
...Program finished with exit code 0
```

```
Press ENTER to exit console.[]
```

//Лабораторная работа №1, задание №2 Колпаковский Н.С. 12.12.2023

```
#include <stdio.h>
```

```
#include <locale.h>
```

```
#define MAX_STUDENTS 5
```

```
#define MAX_SUBJECTS 5
```

```
int main() {
```

```
    start: setlocale(LC_ALL, "Russian");
```

```
    char answer_repeat;
```

```
    int students_count, subjects_count, i, j ;
```

```
    // Ввод количества учеников и предметов
```

```
    printf("Введите количество учеников: ");
```

```
    scanf("%d", &students_count);
```

```
    printf("Введите количество предметов: ");
```

```
    scanf("%d", &subjects_count);
```

```
    // Проверка наличия учеников и предметов
```

```
    if (n <= 0 || subjects_count <= 0) {
```

```
        printf("Ошибка! Недопустимые значения!");
```

```
        return 1;
```

```
    }
```

```
    // Инициализация массива оценок
```

```
    int marks[MAX_STUDENTS][MAX_SUBJECTS];
```

```
    // Ввод оценок
```

```
    printf("Введите оценки:\n");
```

```
    // Цикл учеников
```

```
    for (i = 0; i < students_count; i++) {
```

```
        // Цикл предметов i-того ученика
```

```
        for (j = 0; j < subjects_count; j++) {
```

```

        printf("Введите оценку ученика номер %d: ", i+1);

        scanf(" %d", &marks[i][j]);

    }
}

// Нахождение номера ученика с максимальным средним баллом

// Инициализация переменных
double max_avg = 0.0;
int max_avg_student = -1;

// Цикл учеников
for (i = 0; i < students_count; i++) {
    // Переменная для расчета общего количества баллов
    double sum = 0.0;

    // Цикл предметов i-того ученика
    for (j = 0; j < subjects_count; j++) {
        // Сложение оценок
        sum += marks[i][j];
    }

    // Нахождение среднего значения
    double avg = sum / subjects_count;

    // Если найденное среднее значение больше максимального найденного(при первой итерации всегда
    правда),
    // то заменяет его
    if (avg > max_avg) {
        max_avg = avg;
        max_avg_student = i + 1;
        printf("%d", max_avg_student);
    }
}

// Вывод номера ученика с максимальным средним баллом
if (max_avg_student != -1) {
    printf("Номер ученика с максимальным средним баллом: %d\n", max_avg_student);
}
else {
    printf("Ошибка! Не удалось найти ученика с максимальным средним баллом!\n");
}

```

```

    }

    printf("Повторить? y/N");

    // Ввод согласия или не согласия на повторение, игнорируя
    scanf(" %c", &answer_repeat);

    if(answer_repeat == 'y' | answer_repeat == 'Y') {

        // Возврат к началу исполнения кода

        goto start;

    }

    return 0;
}

```

```

Введите количество учеников: 2
Введите количество предметов: 2
Введите оценки:
Введите оценку ученика номер 1: 3
Введите оценку ученика номер 1: 4
Введите оценку ученика номер 2: 6
Введите оценку ученика номер 2: 7
12Номер ученика с максимальным средним баллом: 2
Повторить? y/NY
Введите количество учеников: 1
Введите количество предметов: 1
Введите оценки:
Введите оценку ученика номер 1: 2
1Номер ученика с максимальным средним баллом: 1
Повторить? y/N

```



```
#include <stdio.h>
```

```
// Функция слияния массивов
```

```
void mergeArrays(int* firstArr, int* secondArr, int* thirdArr, int firstLength, int secondLength) {
```

```
    // Инициализация переменных
```

```
    int i = 0, j = 0, k = 0;
```

```
    while (i < firstLength && j < secondLength) {
```

```
        // Берем меньшее значение и вставляем его в конечный массив
```

```
        if (firstArr[i] <= secondArr[j]) {
```

```
            thirdArr[k++] = firstArr[i++];
```

```
        }
```

```
        else {
```

```
            thirdArr[k++] = secondArr[j++];
```

```
        }
```

```
    }
```

```
    while (i < firstLength) {
```

```
        thirdArr[k++] = firstArr[i++];
```

```
    }
```

```
    while (j < secondLength) {
```

```
        thirdArr[k++] = secondArr[j++];
```

```
    }
```

```
}
```

```
// Функция сортировки "пузырьком"
```

```
void bubbleSort(int* array, int n) {
```

```
    // Инициализация переменных
```

```
    int i, j, temp;
```

```
    for (i = 0; i < n - 1; i++) {
```

```

    for (j = 0; j < n - i - 1; j++) {
        // Если значение больше следующего поменять местами
        if (array[j] > array[j + 1]) {
            temp = array[j];
            array[j] = array[j + 1];
            array[j + 1] = temp;
        }
    }
}
}

```

```

int main() {
    start:
    char answer_repeat;

    // Первый исходный массив
    int frArr[] = { 1, 2, 3, 4, 5 };
    // Второй исходный массив
    int scArr[] = { 10, 9, 8, 7, 6 };
    // Пустой массив для объединенного содержимого
    int thArr[10];

    // Длина первого массива
    int firstArrayLength = sizeof(frArr) / sizeof(frArr[0]);
    // Длина второго массива
    int secondArrayLength = sizeof(scArr) / sizeof(scArr[0]);

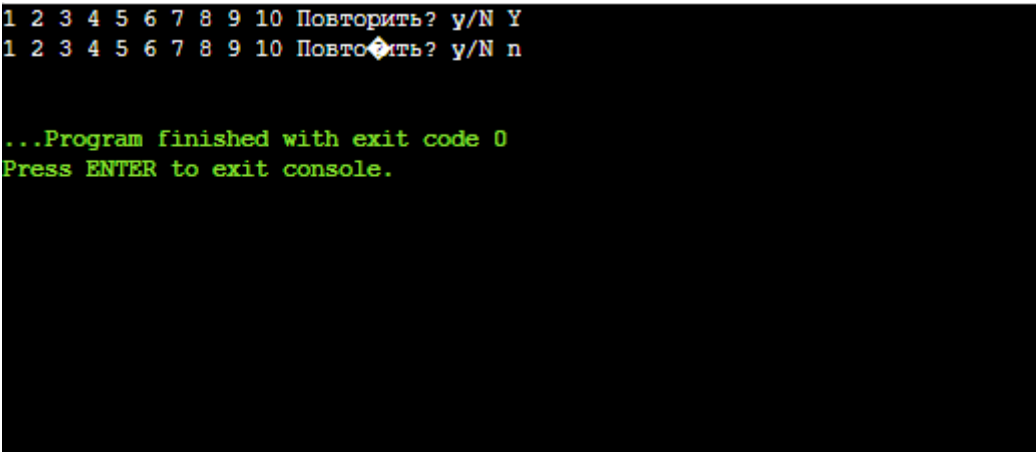
    // Объединение массивов
    mergeArrays(frArr, scArr, thArr, firstArrayLength, secondArrayLength);

    // Сортировка "пузырьком"
    bubbleSort(thArr, firstArrayLength + secondArrayLength); // Сортировка объединенного массива

    for (int i = 0; i < firstArrayLength + secondArrayLength; i++) {
        printf("%d ", thArr[i]); // Вывод элементов объединенного и отсортированного массива
    }
}

```

```
printf("Повторить? y/N");  
  
// Ввод согласия или не согласия на повторение, игнорируя  
scanf(" %c", &answer_repeat);  
  
if(answer_repeat == 'y' | answer_repeat == 'Y') {  
    // Возврат к началу исполнения кода  
    goto start;  
}  
  
  
return 0;  
}
```



```
1 2 3 4 5 6 7 8 9 10 Повторить? y/N Y  
1 2 3 4 5 6 7 8 9 10 Повторить? y/N n  
  
...Program finished with exit code 0  
Press ENTER to exit console.
```