****

Инженерная школа информационных технологий и робототехники

15.03.04 Автоматизация технологических процессов и производств

Отделение автоматизации и робототехники

**Отчёт по лабораторной работе №1**

По дисциплине: Основы программирования и алгоритмизации

Вариант №5



Выполнил: студент гр. 8Т32 \_\_\_\_\_\_\_ \_\_\_\_\_ Гаврильев П.П.

Подпись Дата Фамилия И.О.

Проверил: ассистент ОАР \_\_\_\_\_\_\_ \_\_\_\_\_ Павловский А.В.

Подпись Дата Фамилия И.О.

Томск 2024 г.

**Цель работы:**

Изучить и выполнить задания по следующей теории: функции с C++; передача параметров по значению, ссылке и указателю; параметры по умолчанию.

**Ход работы:**

Задание 1.

Суть задачи заключается в том, чтобы найти суммарную площадь всех колец, по введенному количеству колец и внутреннему радиусу. Блок схема задачи представлена на рисунке 1.1. Код программы задания представлена на листинге 1.1.

Для выполнения этой задачи требуется использование функции, блок схема, которой представлена на рисунке 1.2, а код на листинге 1.2. Изначально программа просит у пользователя ввести количество колец n и внутренний радиус r, потом заданные параметры заносятся в цикл, который будет выполняться n раз. В цикле будет вызываться функция, в которой будет рассчитываться площадь одного кольца, потом суммировать с другими площадями колец. И после в конце будет выводиться результат. Итог запуска программы представлен на рисунке 1.3.

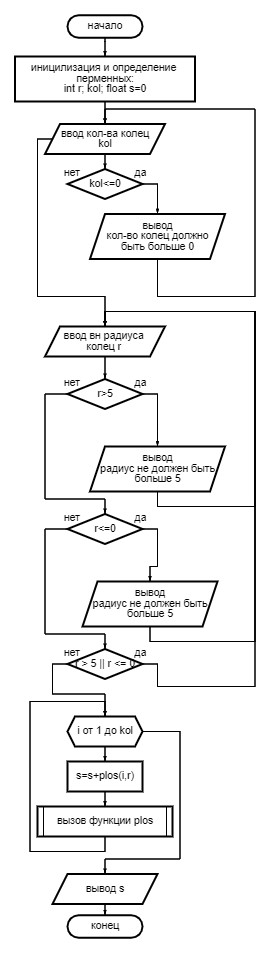


Рисунок 1.1 – Блок схема программы



Рисунок 1.2 – Блок схема функции square(i,r)



Рисунок 1.3 – Итог запуска программы

Листинг 1.1 – Код программы

|  |
| --- |
| #include <iostream>  #include "square.h"  using namespace std;  int main()  {  setlocale(LC\_ALL, "ru");  float s = 0, r=0;  int kol=0;  cout << "Введите количество колец>>";  cin >> kol;  while (kol <= 0) {  cout << "Количество колец должно быть больше 0. Введите еще раз>>";  cin >> kol;  }  cout << "Введите внутренний радиус колец>>";  cin >> r;  do {  while (r > 5) {  cout << "Радиус не должен быть больше или равным 5. Введите еще раз>>";  cin >> r;  }  while (r <= 0) {  cout << "Радиус не должен быть отрицательным. Введите еще раз>>";  cin >> r;  }  } while (r > 5 || r <= 0);  for (int i = 1; i <= kol; i++) {  s = s + square(i, r);  }  cout << "Суммарная площадь колец>>" << s;  } |

Листинг 1.2 - Библиотека plos

|  |
| --- |
| #pragma once  #include <math.h>  float square(int i, float r){  float s = 0, pi = 3.141593;  return (pi \* ((pow((5 \* i), 2)) - (pow(r, 2))));  } |

Задание 2.

В этом задании нужно отсортировать введенный массив с клавиатуры по такому принципу, нужно вывести сначала все отрицательные числа, потом все положительные. Сортировка выполняется по тому порядку, как они были введены.

Нужно создать динамический массив размер, которой будет задаваться пользователем. Элементы массива вводятся с клавиатуры. Нужно пропустить заданный массив через функцию, которая сортирует массив. Блок схема функции показана на рисунке 2.2, а код на листинге 2.2. Блок схема задания представлена на рисунке 2.1, код задания на листинге 2.1, результат программы на рисунке 2.2.

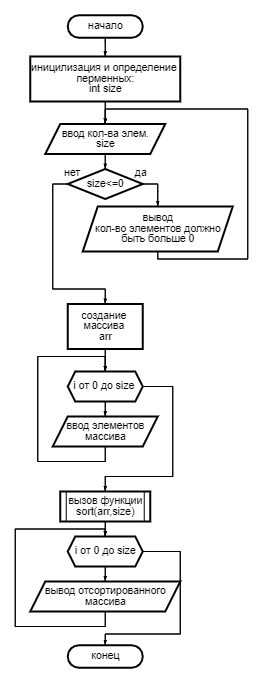


Рисунок 2.1 – Блок схема программы

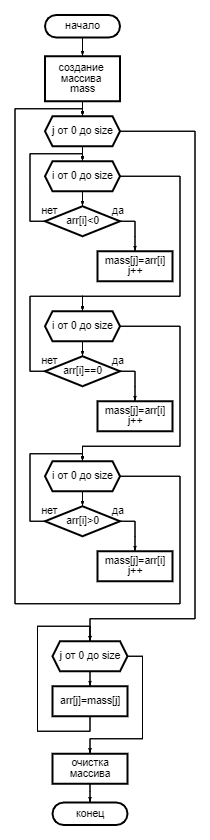


Рисунок 2.2-Блок схема функции sort

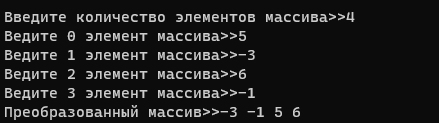


Рисунок 2.3 – Итог запуска программы

Листинг 2.1 – Код программы

|  |
| --- |
| #include <iostream>  #include "sort.h"  using namespace std;  int main()  {  setlocale(LC\_ALL, "ru");  int size;  cout << "Введите количество элементов массива>>";  cin >> size;  while (size <= 0) {  cout << "Количество элементов должно быть больше 0. Введите еще раз>>";  cin >> size;  }  float\* arr = new float[size];  for (int i = 0; i < size; i++)  {  cout << "Ведите " << i+1 << " элемент массива>>";  cin >> arr[i];  }  cout << "Преобразованный массив>>";  sort(arr, size);  for (int j = 0; j < size; j++) {  cout << arr[j] << " ";  }  delete[] arr;  } |

Листинг 2.2-библиотека sort

|  |
| --- |
| #pragma once  #include "sort.h"  void sort(float\* arr, int size) {  float\* mass = new float[size];  for (int j = 0; j < size; j++) {  for (int i = 0; i < size; i++)  {  if (arr[i] < 0) {  mass[j] = arr[i];  j++;  }  }  for (int i = 0; i < size; i++)  {  if (arr[i] == 0) {  mass[j] = arr[i];  j++;  }  }  for (int i = 0; i < size; i++)  {  if (arr[i] > 0) {  mass[j] = arr[i];  j++;  }  }  }  for (int j = 0; j < size; j++) {  arr[j] = mass[j];  }  delete[] mass;  } |

Задание 3.

Нужно реализовать функцию вычисляющую значение с точностью .

Чтобы выполнить задание нужно написать код, который будет принимать значения x, eps. Заданные значения будут передаваться функции, которая будет вычислять уравнение. Блок схема показана на рисунке 3.1, код программы на листинге 3.1. Блок схема функции представлена на рисунке 3.2, код на листинге 3.2. Вывод программы показана на рисунке 3.3.

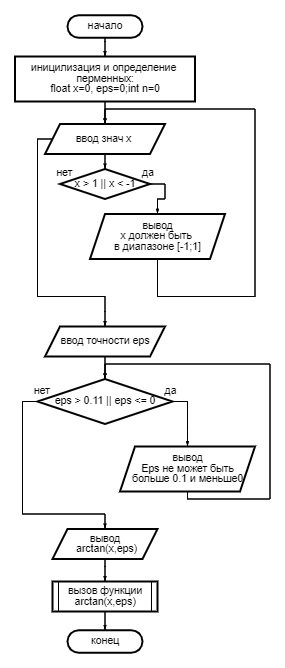


Рисунок 3.1 – Блок схема программы

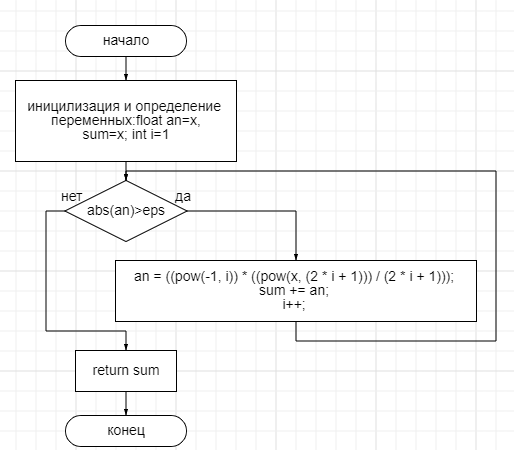


Рисунок 3.2 – Блок схема функции arctan(x,eps)

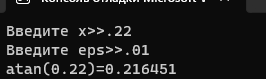


Рисунок 3.3 – Итог запуска программы

Листинг 4 – Код программы 3

|  |
| --- |
| #include <iostream>  #include "arctanx.h"  #include <math.h>  int main()  {  setlocale(LC\_ALL, "ru");  float x = 0, eps = 0;  std::cout << "Введите x>>"; std::cin >> x;  while (x > 1 || x < -1) {  std::cout << "x должен быть в диапазоне [-1;1]. Введите еще раз>>";  std::cin >> x;  }  std::cout << "Введите eps>>"; std::cin >> eps;  while (eps > 0.11 || eps <= 0) {  std::cout << "Eps не может быть больше 0.1 и меньше 0. Введите еще раз>>"; std::cin >> eps;  }    std::cout << "atan(" << x << ")=" << arctanx(x, eps);  } |

Листинг 5 – библиотека arctanx

|  |
| --- |
| #pragma once  #include <math.h>  float arctanx(float x, float eps)  {  float an = x, sum = x;  int i = 1;  while (fabsf(an) > eps) {  an = ((pow(-1, i)) \* ((pow(x, (2 \* i + 1))) / (2 \* i + 1)));  sum += an;  i++;  }  return sum;  } |

**Вывод**

В ходе лабораторной работы была изучена следующая информация: функции; передача параметров по значению, ссылки и указателю, динамический массив. На основе этой информации была проделана работа по написанию кода, представленных на листингах (1.1, 1.1, 2.1, 2.2, 3.1, 3.2), для заданий. Также были сделаны блок схемы, которые показаны на рисунках (1.1, 1.2, 2.1, 2.2, 3.1, 3.2). В итоге благодаря этой информации можно выполнить различные задачи, таких как вычисление математических задач, физические задачи, сортировка данных и т.д.