Федеральное государственное автономное

образовательное учреждение

высшего образования

«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий

Кафедра вычислительной техники

**ОТЧЕТ О ПРАКТИЧЕСКОЙ** **РАБОТЕ № 24**

Приближенные методы решения уравнений и вычисления определенного интеграла  
 Вариант № 2

Преподаватель \_\_\_\_\_\_\_\_ Пушкарев К. В.

подпись, дата

Студент КИ18-09б, 031830645 \_\_\_\_\_\_\_\_ Котов С.А.

подпись, дата

Студент КИ18-09б, 031831293 \_\_\_\_\_\_\_\_ Овсянников В.А.

подпись, дата

Студент КИ18-09б, 031831050 \_\_\_\_\_\_\_\_ Машковская Н.В.

подпись, дата

Красноярск 2019

**1 Дополнительное упражнение № 1 (вариант 2)**

Напечатать в порядке возрастания корни уравнений, вычисленные с заданной точностью, двумя способами: методом хорд и методом половинного деления:

1 / (1 + × ) =

4 – – 5 = 0

Точность и интервал вводятся с клавиатуры. Полученные результаты сравнить.

**2 Цель работы**

Получить практические навыки использования указателей на функции

как параметров функции, получить опыт командной работы.

**3 Графическая схема алгоритма**

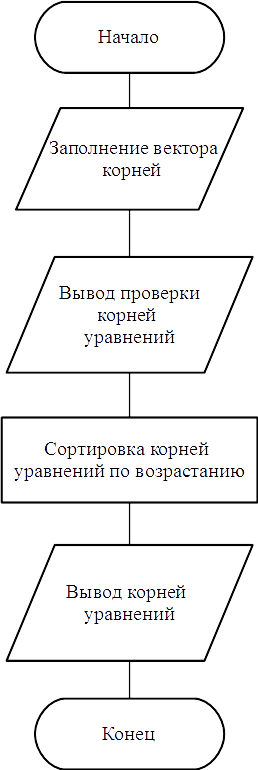


Рисунок 1 – Графическая схема алгоритма

**4 Код программы**

|  |  |
| --- | --- |
| 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  26  27 28 29 30 31 32 33 34 35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78 | #include "stdio.h"  #include <iostream>  #include <clocale>  #include <cmath>  #include <algorithm>  #include <vector>  using namespace std;  typedef double(\*left\_part)(const double);  double func1(const double x) {  return (1 / (1 + x \* x)) - x;  }  double func2(const double x) {  return 4 \* x - log(x) - 5;  }  double func3(const double x) {  return x - sin(x) - 0.25;  }  // Метод половинного деления(бисекции)  double solution\_div2(left\_part f, double a, double b, const double eps) {  double x;  do {  x = (a + b) / 2;  if (f(a) \* f(x) > 0) {  a = x;  }  else {  b = x;  }  } while (fabs(b - a) > eps);  return x;  }  // Метод хорд  double solution\_chord(left\_part f, double a, double b, const double eps) {  while (fabs(b - a) > eps) {  a = b - (b - a) \* f(b) / (f(b) - f(a));  b = a - (a - b) \* f(a) / (f(a) - f(b));  }  return b;  }  int main() {  setlocale(LC\_ALL, "");  double a, b, eps;  vector<left\_part> f = { func1, func2, func3 };// Вектор указателей на функции  vector<double> solutions;// Вектор результатов  for (int i = 0; i < f.size(); i++) {  cout << "Введите a, b и eps для " << i + 1 << " уравнения: ";  cin >> a >> b >> eps;  if (f[i](a) \* f[i](b) > 0) {  cout << "Нет корней" << endl;  }  else {  double temp = solution\_div2(f[i], a, b, eps);  solutions.push\_back(temp);// Запись корней функции(метод бисекции)  cout << "Проверка корней уравнений: " << f[i](temp) << " ";  temp = solution\_chord(f[i], a, b, eps);  solutions.push\_back(temp);// Запись корней функции(метод хорд)  cout << f[i](temp) << endl;  }  }  // Сортировка корней уравнений  sort(solutions.rbegin(), solutions.rend());  cout << "Корни уравнений: " << endl;  for (int i = 0; i <= f.size(); i++) {  cout << solutions[i] << " ";  }  return 0;  } |

**5 Результаты выполнения экспериментальной части работы**

Результаты запуска программы с различными входными значениями приведены ниже.

