Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное учреждение высшего образования

«Пермский национальный исследовательский политехнический университет»

ПНИПУ

Лабораторная работа

**«Решение нелинейного уравнения разными методами»**

Выполнил:

студент группы РИС-23-3б

Епин Тимофей Евгеньевич

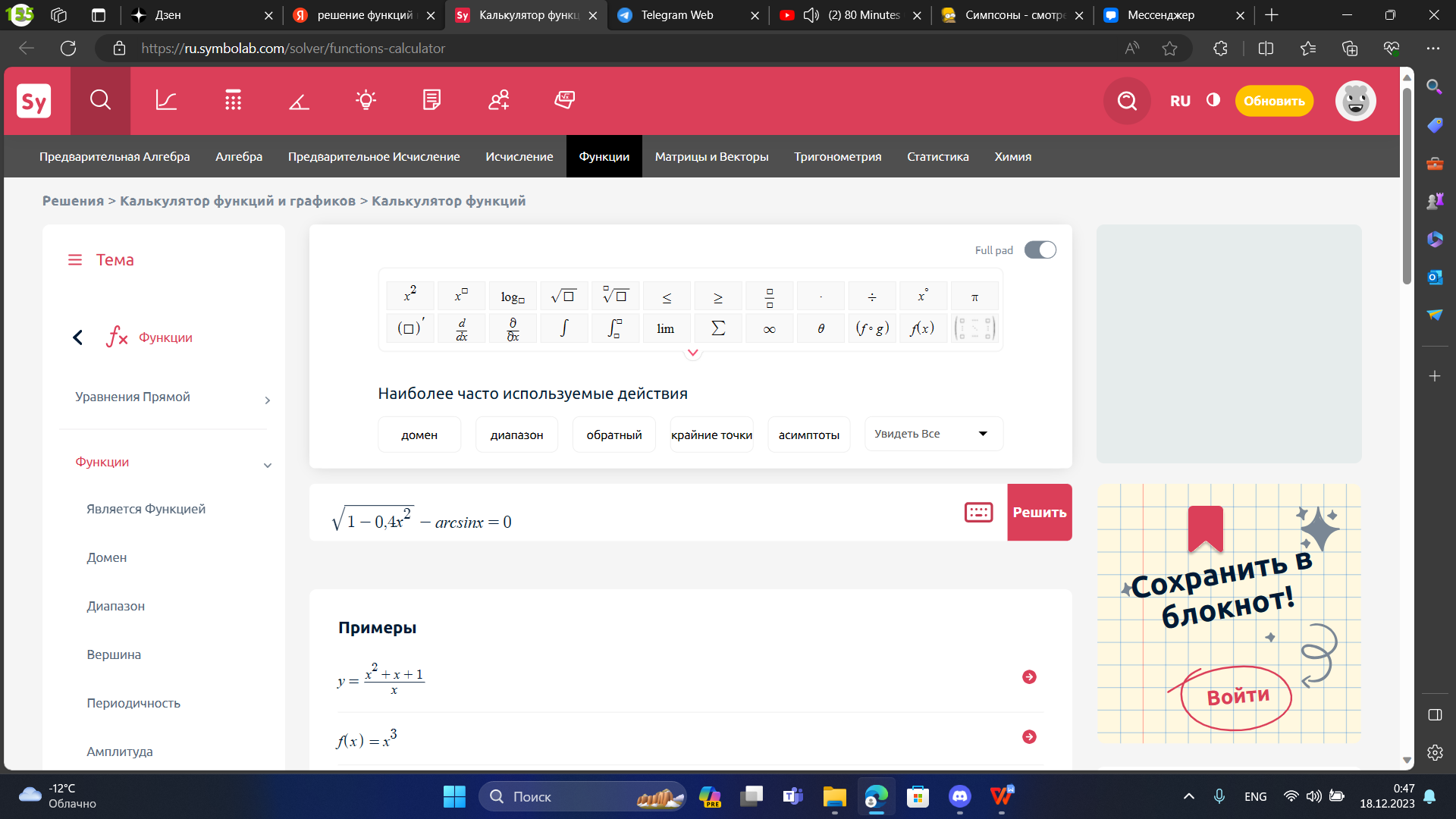
Проверила:

доцент кафедры ИТАС

О.А. Полякова

2023

**Постановка задачи:**

Разработать алгоритм и написать код на языке C++ для решения данного уравнения:

**Метод половинного деления**

**Словесный алгоритм:**

1. В условии задачи уточняется отрезок [a; b], на котором функция имеет корень, сам вид функции , а также требуемая точность E=0,000001;
2. С помощью теоремы1 определяется первый корень, который находится как ;
3. Далее запускается цикл while, который продолжает работу, пока модуль разности между концами отрезка не станет меньше или равен заданной точности ;
   1. Eсли , то значение b заменяется на с, иначе значение a заменяется на с;
4. В результате точное значение корня выводится в виде .

1Теорема: «Если непрерывная функция на концах некоторого интервала имеет значения разных знаков, то внутри этого интервала у нее есть корень».

**Значения переменных:**

a – левый конец отрезка;

b – правый конец отрезка;

с – корень уравнения;

E – точность.

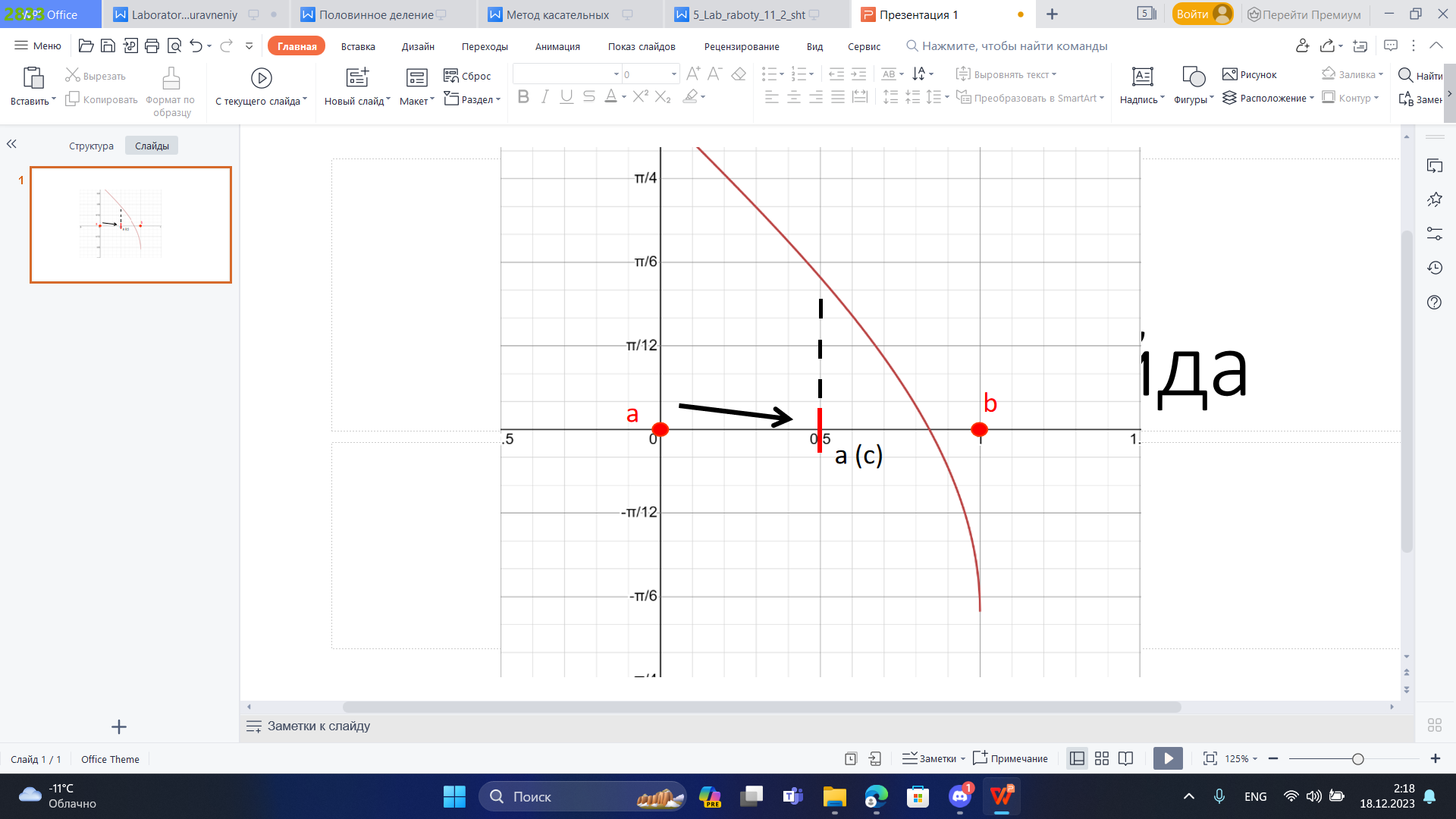
**Решение:**

1. Обозначим начальную и конечную точки отрезка точками **a** и **b** соответственно(**a = 0**, **b = 1**)



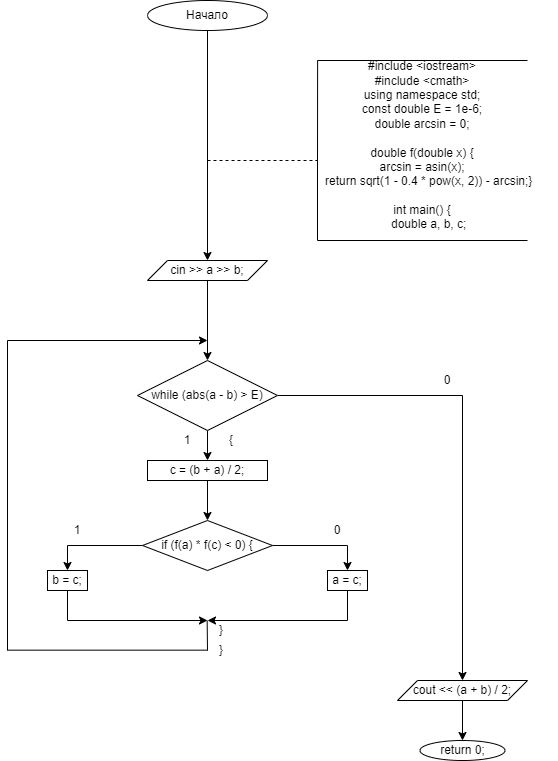
1. В общем виде уравнение имеет вид:
2. Разделим отрезок на две части: с= (b - a)/2 = (1 – 0)/2 = 0,5.
3. Если произведение F(a) \* F(c) < 0, то конец отрезка **b** переносится в **с** (b = c), иначе, начало отрезка **a** переносится в **c** (a = c)

Полученный отрезок делим опять пополам и так далее, пока выполняется условие: |a – b| > ℇ.

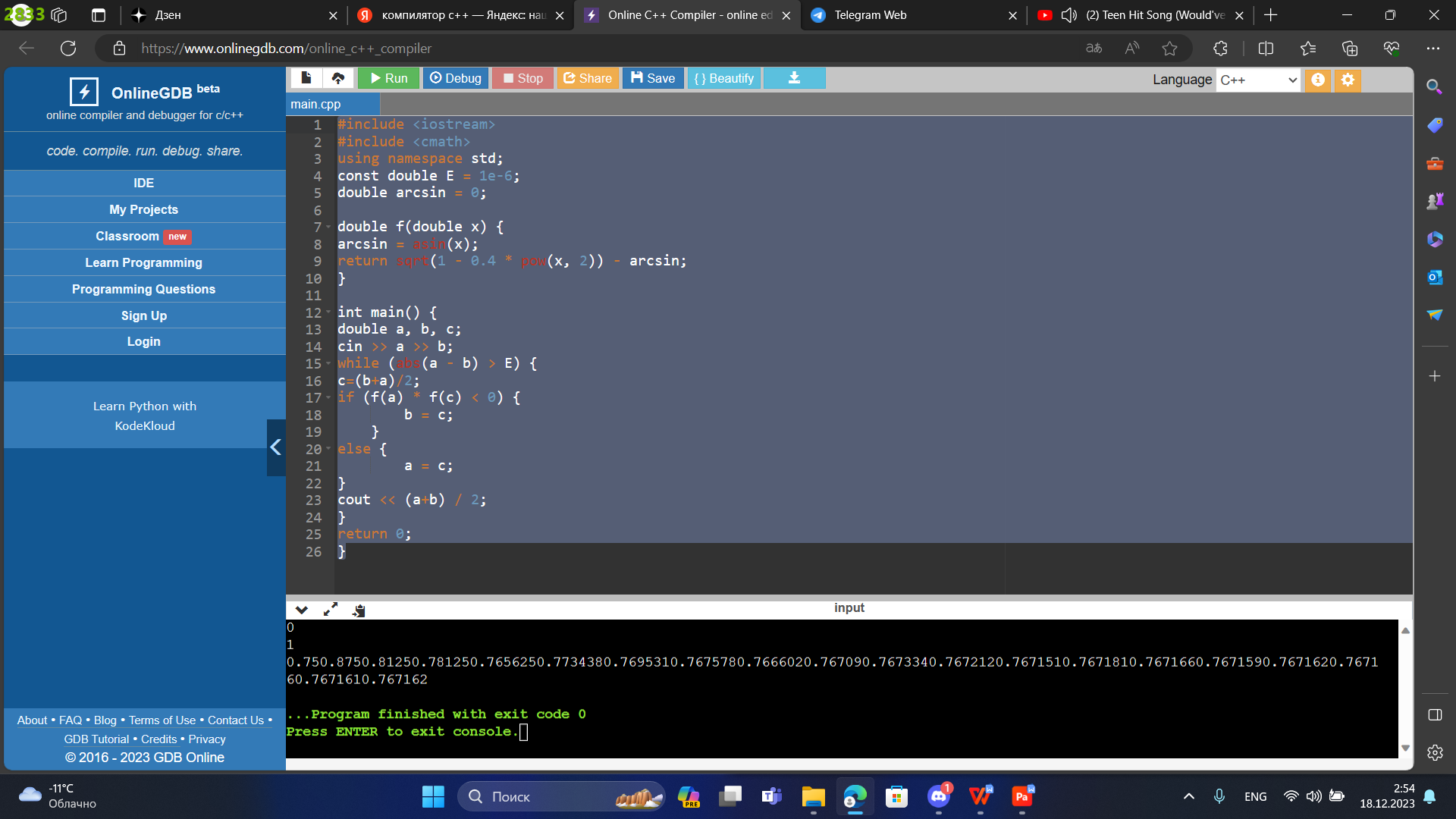


|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **a** | **b** | **c** | **F(a)** | **F(c)** | **F(a) \* F(c)** |
| 0.5 | 1 | 0.5 | 0.425085 | 0.425085 | 0.180697 |
| 0.75 | 1 | 0.75 | 0.322788 | 0.0322788 | 0.00104192 |
| 0.75 | 0.875 | 0.875 | 0.0322788 | -0.232519 | -0.00750543 |
| 0.75 | 0.8125 | 0.8125 | 0.0322788 | -0.0905598 | -0.00292316 |
| 0.765625 | 0.78125 | 0.765625 | 0.00293185 | 0.00293185 | 8.59576e-06 |
| 0.765625 | 0.769531 | 0.769531 | 0.00293185 | -0.00453445 | -1.32943e-05 |
| 0.765625 | 0.767578 | 0.767578 | 0.00293185 | -0.000794598 | -2.32965e-06 |
| 0.76709 | 0.767578 | 0.76709 | 0.000138265 | 0.000138265 | 1.91172e-08 |
| 0.76709 | 0.767334 | 0.767334 | 0.000138265 | -0.000328062 | -4.53595e-08 |
| 0.76709 | 0.767212 | 0.767212 | 0.000138265 | -9.48725e-05 | -1.31175e-08 |
| 0.767151 | 0.767212 | 0.767151 | 2.17028e-05 | 2.17028e-05 | 4.71011e-10 |
| 0.767151 | 0.767181 | 0.767181 | 2.17028e-05 | -3.65832e-05 | -7.93957e-10 |
| 0.767151 | 0.767166 | 0.767166 | 2.17028e-05 | -7.4398e-06 | -1.61464e-10 |
| 0.76716 | 0.767162 | 0.76716 | 3.48877e-06 | 3.48877e-06 | 1.21715e-11 |

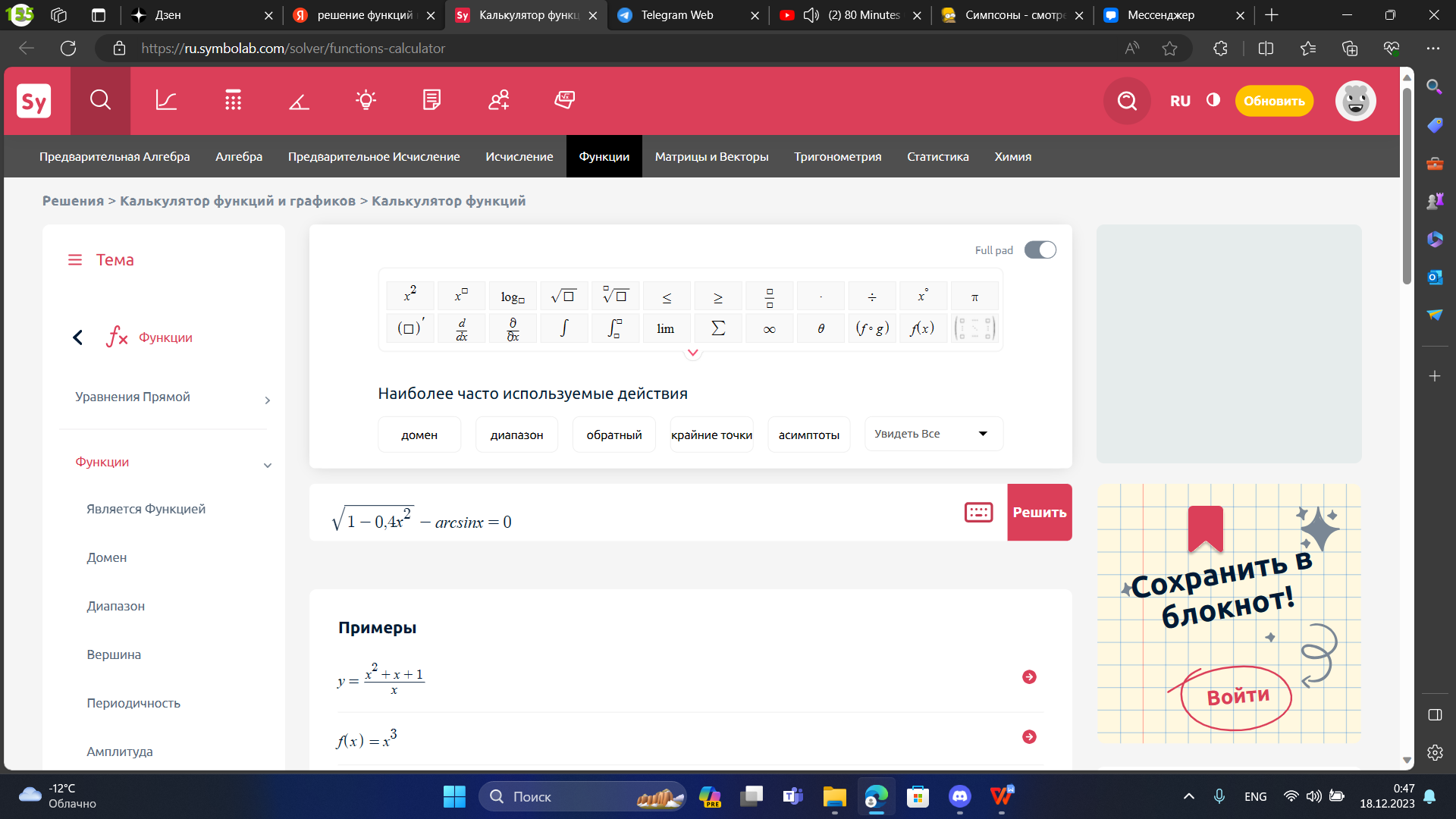
**Блок-схема**

**Код**

|  |
| --- |
| #include <iostream>  #include <cmath>  using namespace std;  const double E = 1e-6;  double arcsin = 0;  double f(double x) {  arcsin = asin(x);  return sqrt(1 - 0.4 \* pow(x, 2)) - arcsin;  }  int main() {  double a, b, c;  cin >> a >> b;  while (abs(a - b) > E) {  c=(b+a)/2;  if (f(a) \* f(c) < 0) {  b = c;  }  else {  a = c;  }  cout << (a+b) / 2;  }  return 0;  } |

**Работа программы:**

**Метод касательных**

**Анализ задачи:**

Необходимо решить уравнение с точностью до

E = 0,000001 на отрезке [0;1].

1. Обозначим начальную и конечную точки отрезка точками **a** и **b** соответственно.



1. В общем виде уравнение имеет вид:
2. Определим начальное приближение:



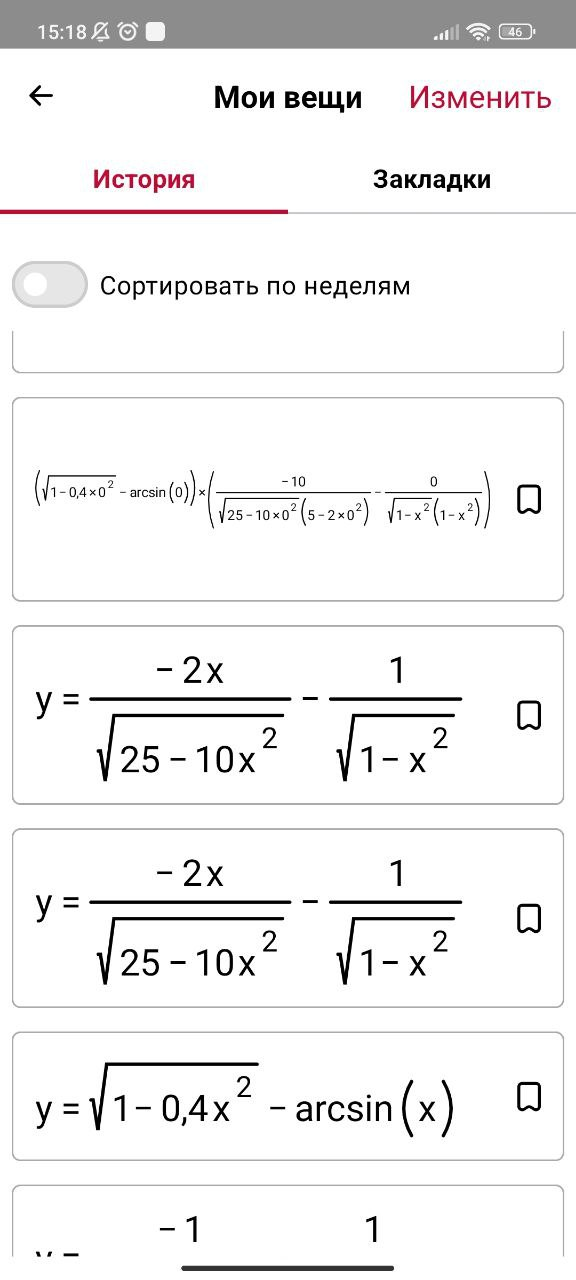
* левый конец: x0 = a, если
* правый конец: x0 = b, если

1. Пока не найдется ,будем выполнять итерации поиска **x:**

****

**Решение:**

1. **Определение начального приближения:**



**F(a) \* F’’(a) = F(0) \* F’’(0) =**

**= -2/5;**

**F(b) \* F’’(b) = F(1) \* F’’(1) –** Не определён, т.к. в знаменателе получается ноль.

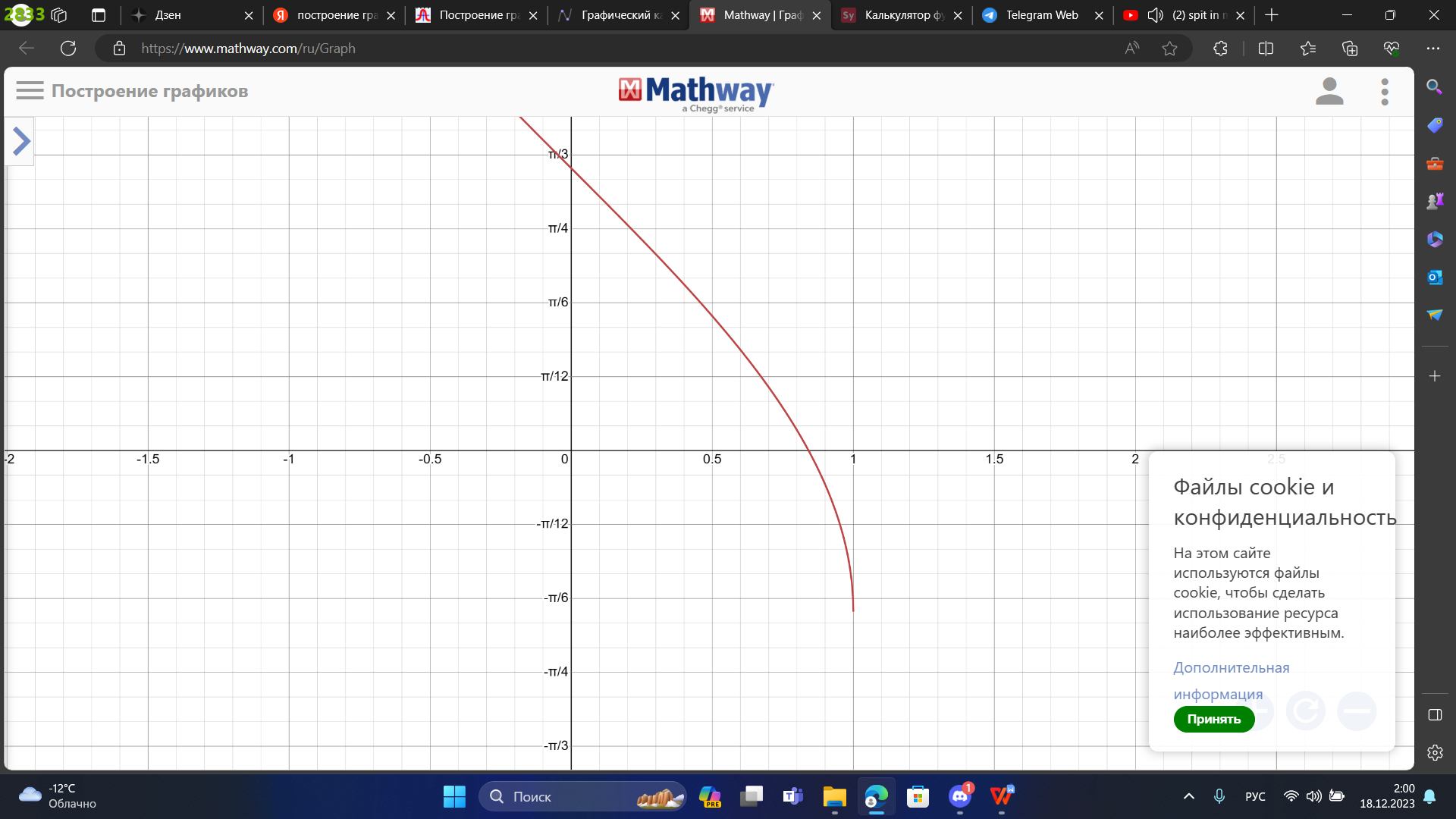
Сместим правую границу: **b** = 0,9.



**F(b) \* F’’(b) = F(0,9) \* F’’(0,9) =**

**= 3,44**

Значит, приближение начнём с правой границы.

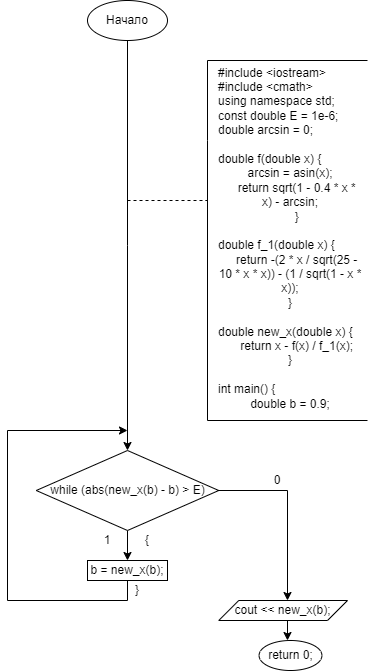


b (x0)

a

a (c)

**Блок схема:**

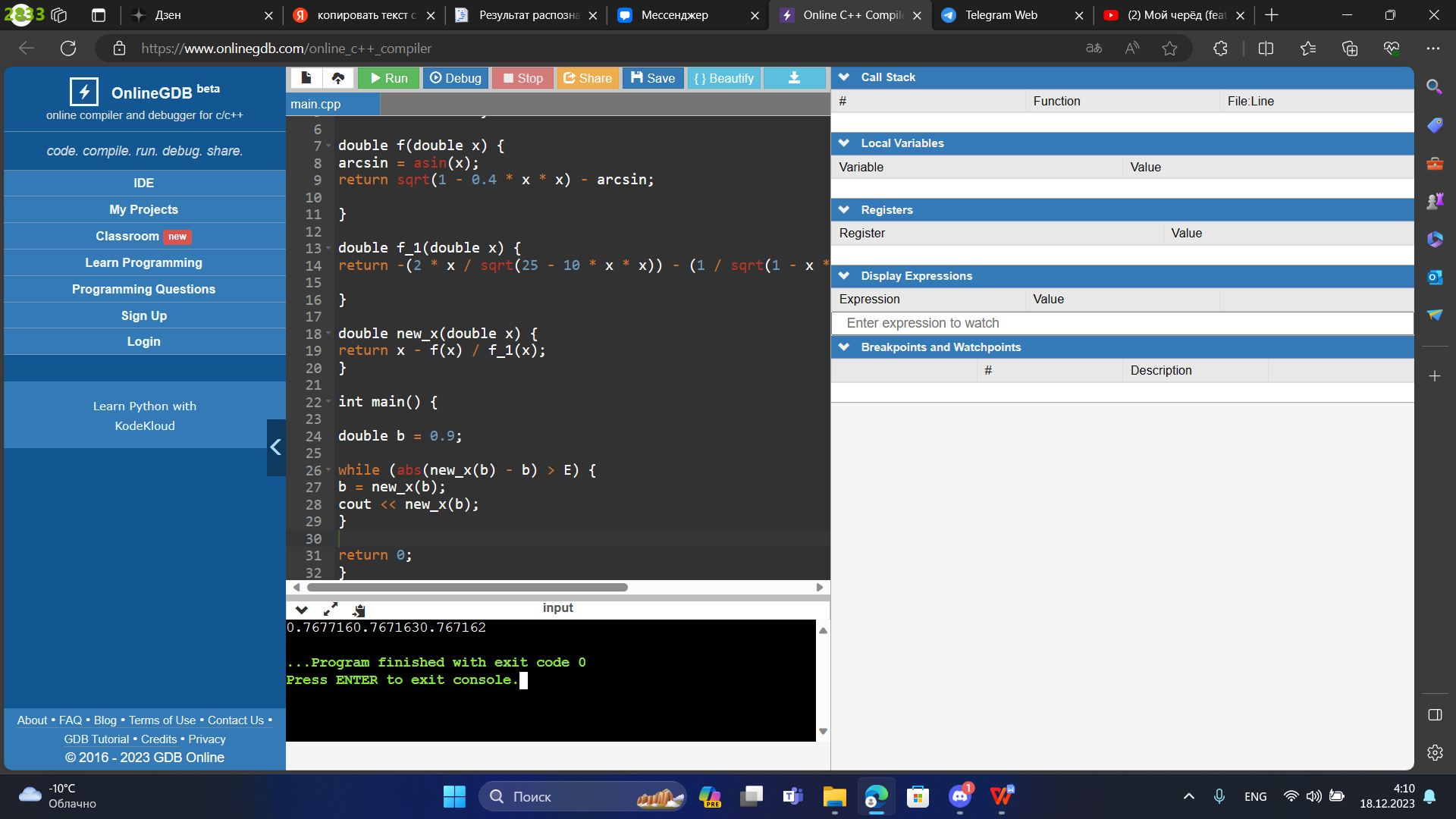
****

|  |  |  |
| --- | --- | --- |
| **Xn** | **Xn-1** | **|Xn – Xn-1|** |
| 0.9 | 0.791078 | 0.108922 |
| 0.791078 | 0.767716 | 0.023362 |
| 0.767716 | 0.767163 | 0.000553079 |

**Код:**

|  |
| --- |
| #include <iostream>  #include <cmath>  using namespace std;  const double E = 1e-6;  double arcsin = 0;  double f(double x) {  arcsin = asin(x);  return sqrt(1 - 0.4 \* x \* x) - arcsin;  }  double f\_1(double x) {  return -(2 \* x / sqrt(25 - 10 \* x \* x)) - (1 / sqrt(1 - x \* x));  }  double new\_x(double x) {  return x - f(x) / f\_1(x);  }  int main() {  double b = 0.9;  while (abs(new\_x(b) - b) > E) {  b = new\_x(b);  cout << new\_x(b);  }  return 0;  } |

**Работа программы:**



**Вывод:**

Метод Ньютона или же метод касательных находит приближенное число быстрее и точнее, чем метод половинного деления.