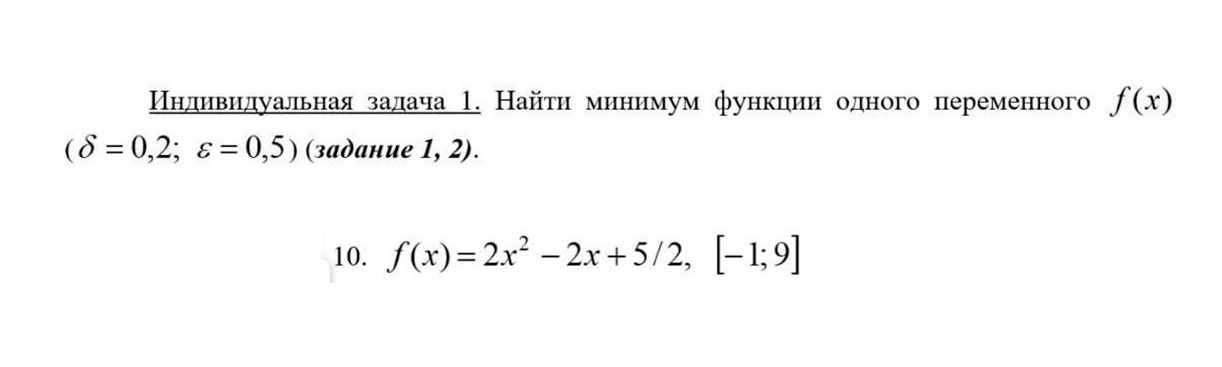
Отчет по 1 лабораторной работе по предмету ”Методы оптимизации”

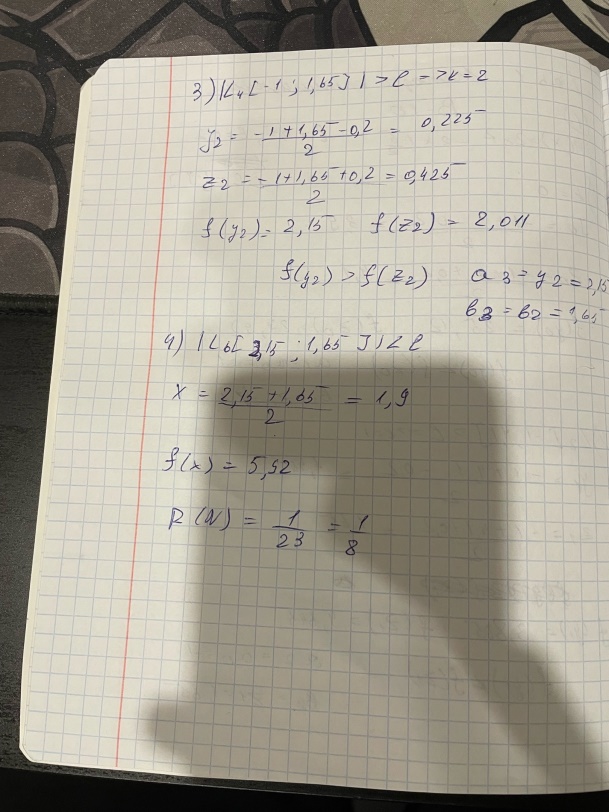
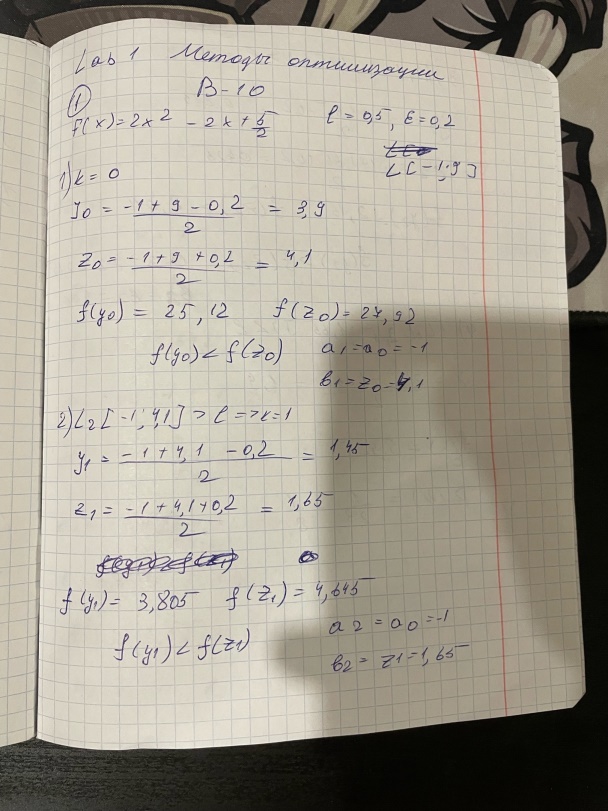
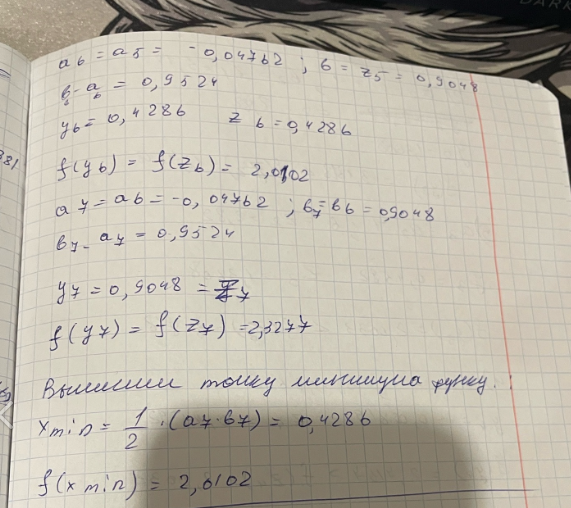
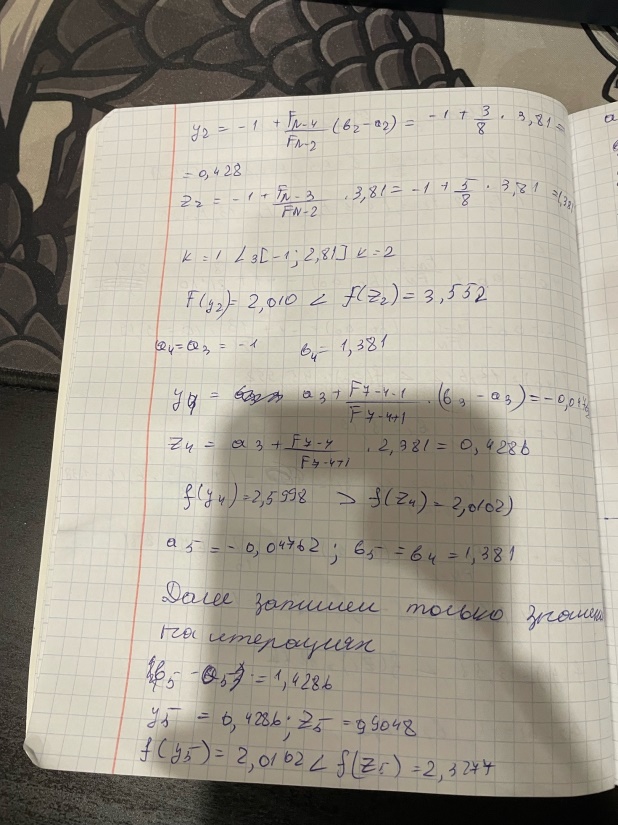
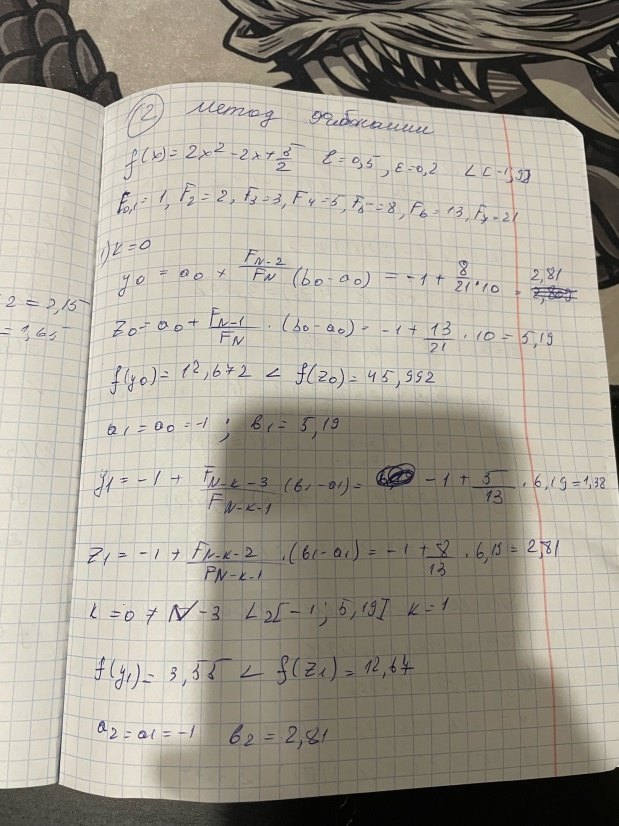
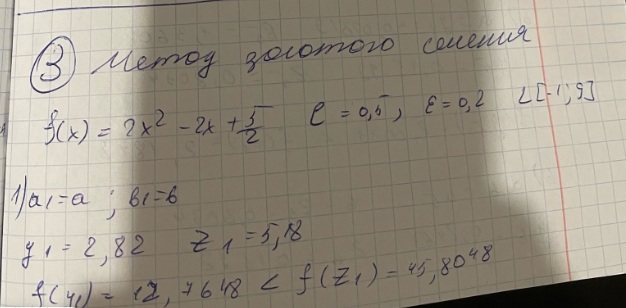
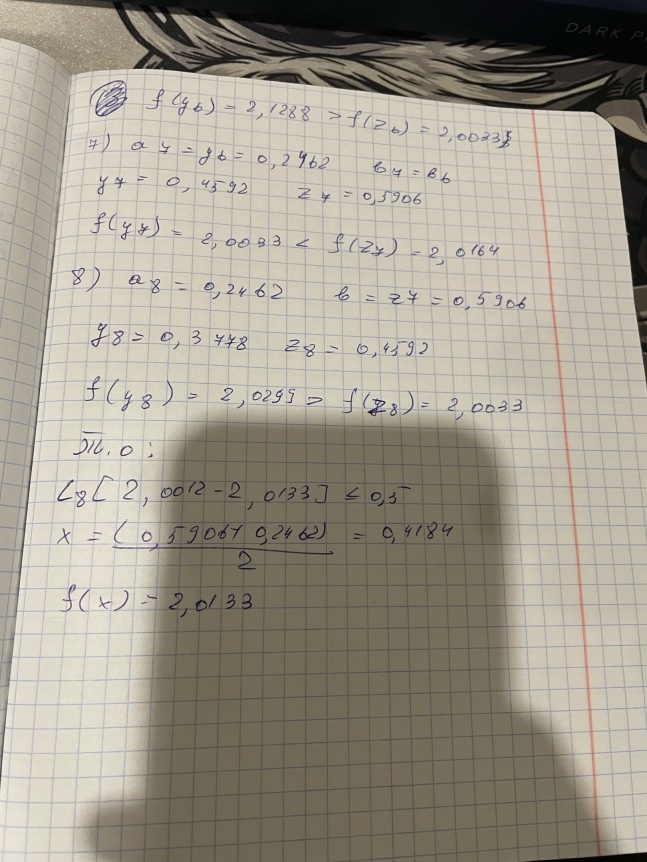
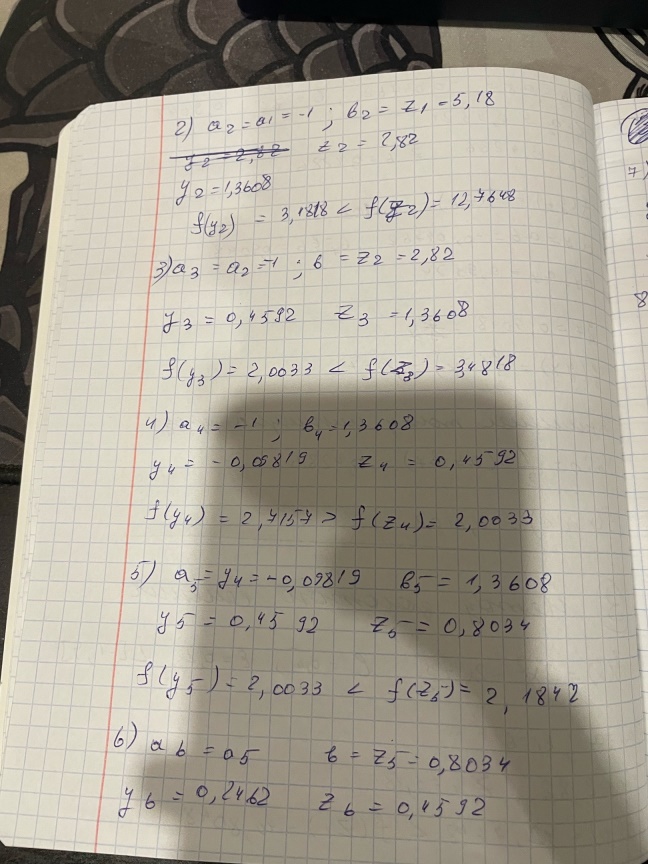
Несветайлов Владислав 34 группа

10 вариант

Условие задачи:



Далее решим эту задачу тремя методами: Дихотомия, Фибоначчи и золотого сечения.

1. Решение методом Дихотомии в тетради:  
     
   
2. Решение методом Фибоначчи в тетради:  
     
   
3. Решение методом золотого сечения в тетради:  
     
     
   

Далее разберем решение задачи данными методами в коде, на языке C++:

#include <iostream>

#include <cmath>

#include <vector>

#include <algorithm>

#include <limits>

using namespace std;

double f(double x) {

return 2 \* x \* x - 2 \* x + 5 / 2;

}

// Метод дихотомии

double dichotomy(double left, double right, double epsilon) {

double mid1, mid2;

while (right - left > epsilon) {

mid1 = left + (right - left) / 3;

mid2 = right - (right - left) / 3;

if (f(mid1) < f(mid2)) {

right = mid2;

}

else {

left = mid1;

}

}

return (left + right) / 2;

}

// Метод Фибоначчи

double fibonacci\_method(double left, double right, double epsilon) {

vector<int> fib{ 0, 1, 1 };

int n = 1;

while (fib[n] < (right - left) / epsilon) {

n++;

fib.push\_back(fib[n - 1] + fib[n - 2]);

}

double x1 = left + fib[n - 2] / (double)fib[n] \* (right - left);

double x2 = left + fib[n - 1] / (double)fib[n] \* (right - left);

for (int k = n - 1; k > 1; k--) {

if (f(x1) < f(x2)) {

right = x2;

x2 = x1;

x1 = left + fib[k - 2] / (double)fib[k] \* (right - left);

}

else {

left = x1;

x1 = x2;

x2 = left + fib[k - 1] / (double)fib[k] \* (right - left);

}

}

return (left + right) / 2;

}

// Метод золотого сечения

double golden\_section(double left, double right, double phi) {

double invPhi = (sqrt(5) - 1) / 2;

double x1 = right - invPhi \* (right - left);

double x2 = left + invPhi \* (right - left);

while (fabs(right - left) > 0.001) {

if (f(x1) < f(x2)) {

right = x2;

}

else {

left = x1;

}

x1 = right - invPhi \* (right - left);

x2 = left + invPhi \* (right - left);

}

return (left + right) / 2;

}

int main() {

double left = -1, right = 9;

double epsilon = 0.5;

double min\_dichotomy = dichotomy(left, right, epsilon);

double min\_fibonacci = fibonacci\_method(left, right, epsilon);

double min\_golden\_section = golden\_section(left, right, 0.618);

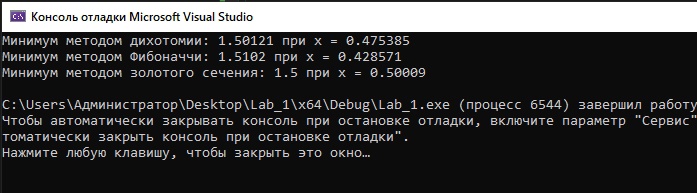
cout << "Минимум методом дихотомии: " << f(min\_dichotomy) << " при x = " << min\_dichotomy << endl;

cout << "Минимум методом Фибоначчи: " << f(min\_fibonacci) << " при x = " << min\_fibonacci << endl;

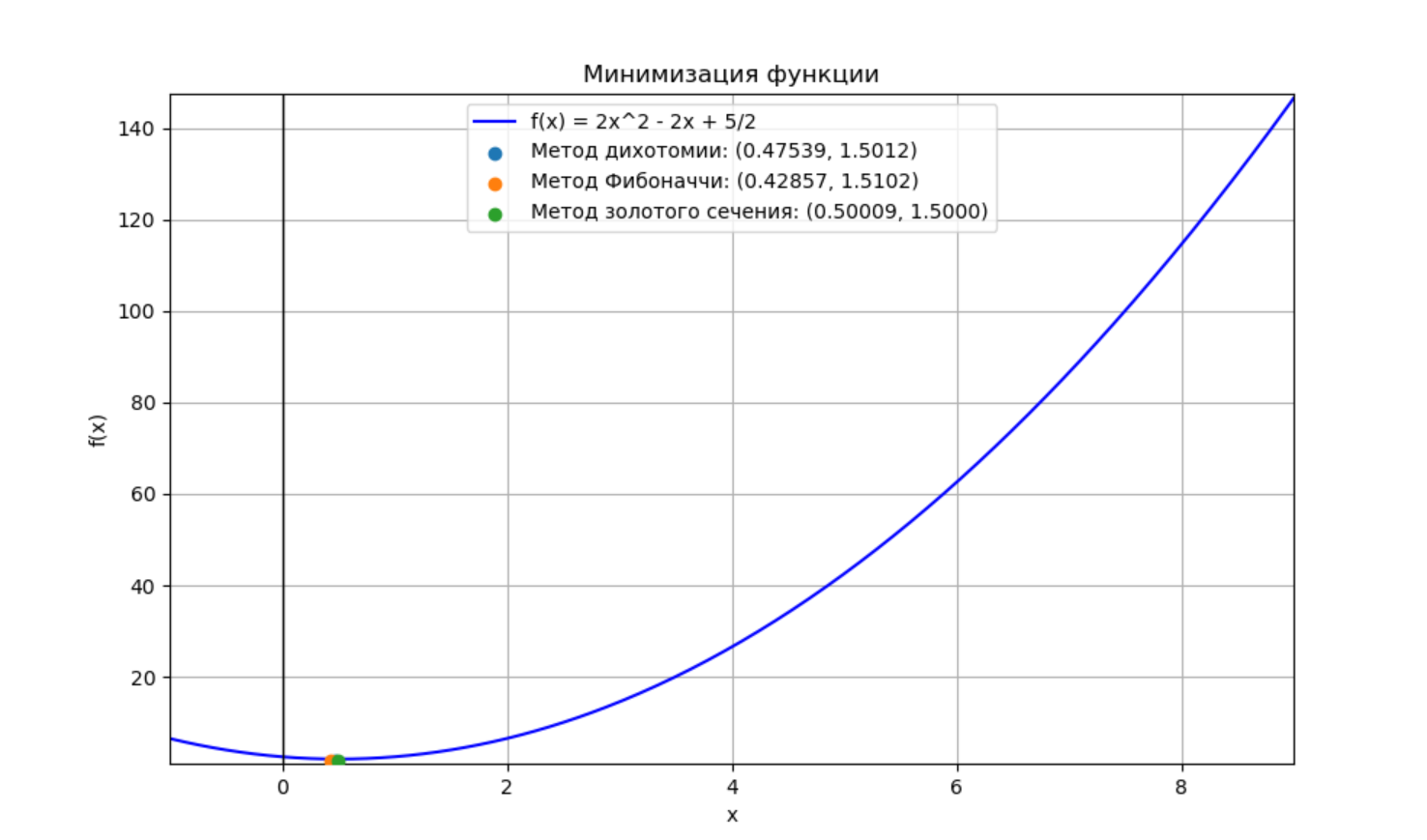
cout << "Минимум методом золотого сечения: " << f(min\_golden\_section) << " при x = " << min\_golden\_section << endl;

return 0;

}  
  
По результатам работы данного кода получаем вывод:



Можем составить график минимумов, согласно этому выводу:



Вывод: Во время выполнения работы были проанализированы 3 метода оптимизации.