**Отчет по курсовой работе** по курсу \_\_Вычислительные системы\_\_\_\_\_\_\_\_\_\_\_

Студент группы М8О-101Б-21 Постнов Александр Вячеславович, № по списку 17

Контакты www, e-mail:

Работа выполнена: «» 202 1г.

Преподаватель: каф. 806 \_\_\_\_\_Титов В.К.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Входной контроль знаний с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Отчет сдан « » \_\_\_\_\_\_\_\_\_2021\_ г., итоговая оценка \_\_\_\_\_

Подпись преподавателя \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Тема:** процедуры и функции в качестве параметров, решение уравнений\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Цель работы:** составить и отладить программу на СИ на заданную тему. \_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Задание** (*вариант: 17*)**:**  Решить 3 уравнения с помощью 4 методов: дихотомия, метод итераций, метод Ньютона, метод хорд.

1) 0.25 \* x \* x \* x + x – 1.2502 = 0 корень на отрезке [0; 2]

2) x \* x + ln(x) - e(число е ) \* x= 0 корень на отрезке [1.5; 3]

3) cos(x) \* ln(1/x) + e \* x \* x – 1.5 = 0 корень на отрезке [0.5; 0.9]

Уравнения 2, 3 составлены самостоятельно

1. **Оборудование(лабораторное)**:

ЭВМ -, процессор -, имя узла сети - с ОП - ГБ,

НМД - ГБ, терминал- адрес -, принтер -

Другие устройства -

*Оборудование ПЭВМ студента, если использовалось:*

Процессор AMD Ryzen 5 4500U, с ОП 8 ГБ

Другие устройства -

1. **Программное обеспечение:**

Операционная система семейства -, наименование - версия - интерпретатор команд - версия

Система программирования - версия -

Редактор текстов - версия -

Утилиты операционной системы -

Прикладные системы и программы -

Местонахождение и имена файлов программ и данных -

*Программное обеспечение ЭВМ студента, если использовалось:*

Операционная система семейства GNU/Linux, наименование Pop!\_OS версия 21.04

интерпретатор команд GNOME Terminal версия 3.38.2

Система программирования \_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_версия\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Редактор текстов emacs версия 3.27.20

Утилиты операционной системы cat

Прикладные системы и программы -

Местонахождение и имена файлов программ и данных -

1. **Идея, метод, алгоритм**  решения задачи (в формах: словесной, псевдокода, графической [блок-схема, диаграмма, рисунок, таблица] или формальные спецификации с пред- и постусловиями)

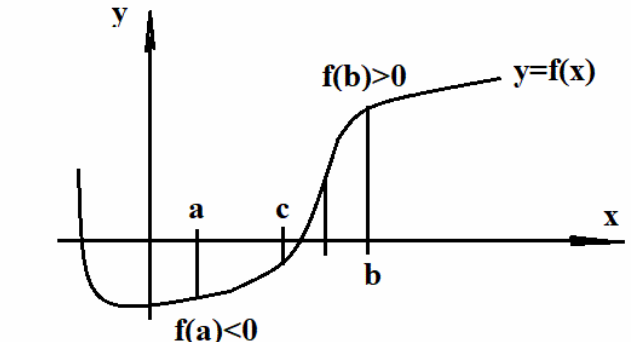
Словесный алгоритм:

1) Дихотомия или метод деления пополам — метод вычисления корней уравнения f (х) = 0, основанный на пошаговом сужении промежутка, в котором находится единственный корень уравнения, пока не добиться заданной точности.

Возьмем две точки х0 и х1, в которых значения функции f (х0) и f (х1) имеют разные знаки. В этом случае между ними имеется хоть один корень функции f.

Разделим промежуток между точками х0 и х1 пополам, обозначим середину отрезка точкой х2, которая равняется: х2 = (х0 + х1) / 2. Тогда f (х2) f (х0) <= 0 или f (х1) f (х0)<= 0.

Выбираем ту часть отрезка, на концах которого функция имеет разные знаки, и вновь делим полученный отрезок пополам. С каждым делением точность увеличивается вдвое. Деление «разнознакового» отрезка продолжаем, пока не сузится область нахождения корня функции, что поможет наконец найти его с большой степенью точности



2) Метод итераций:

Пусть дана функция f(x) Заменим исходное уравнение f(x) на эквивалентное F(x) = x. Выберем начальное приближение корня x0. Тогда получим некоторое число x1 = F(x0) Теперь подставляя вместо x0 число x1 получим x2 = F(x1) . Повторяя этот процесс, будем иметь последовательность чисел. Если последовательность сходящаяся, это будет происходить до тех пор, пока (fabs(x(n) – x(n – 1))) > eps(машинное эпсилон). Функция будет возвращать х(n)

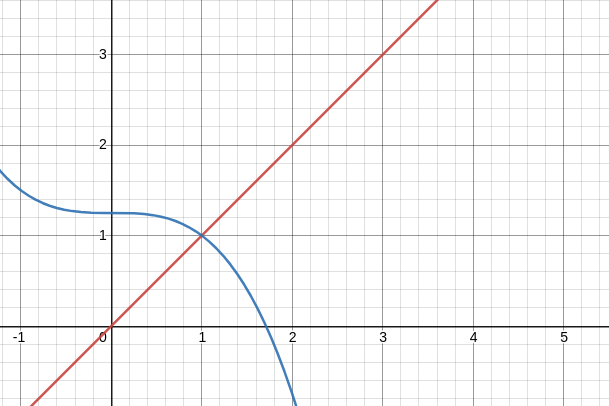
Условие сходимости: |F’(x)| < 1

Найдем F1(x), F2(x), F3(x):

1. 0.25 \* x \* x \* x + x – 1.2502 = 0

x = 1.2502 – 0.25 \* x \* x \* x, значит

F1(x) = 1.2502 – 0.25 \* x \* x \* x



Красная линия: y = x;

Синяя линия: y = 1.2502 – 0.25 \* x \* x \* x;

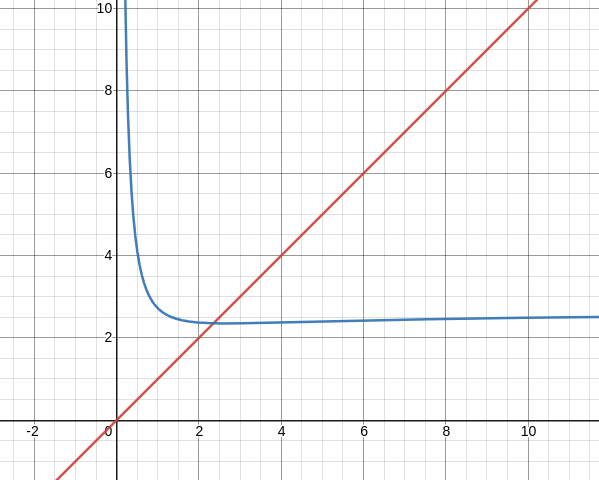
Видно по графикам, что условие сходимости выполняется

2. x \* x + ln(x) – e \* x = 0

x \* x = e \* x – ln(x)

x = (e \* x – ln(x)) / x

F2(x) = (e \* x – ln(x)) / x



Красная линия: y = x;

Синяя линия: y = (e \* x – ln(x)) / x;

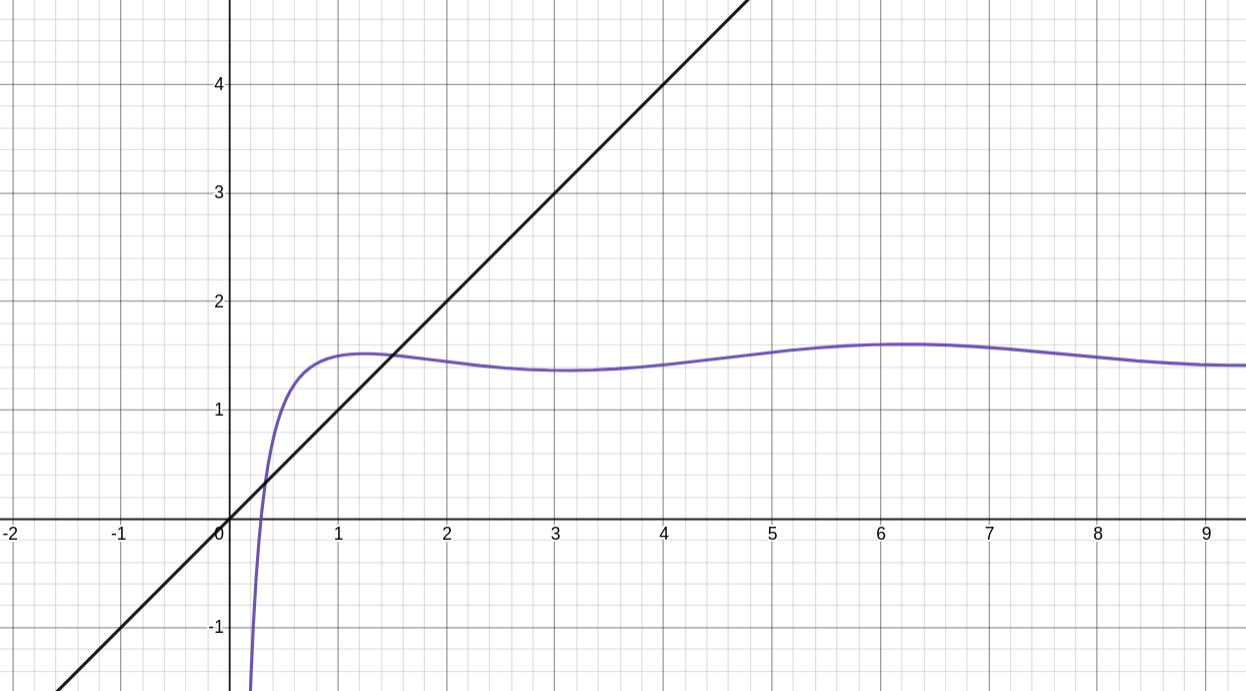
Видно по графикам, что условие сходимости выполняется

3. cos(x) \* ln(1 / x) + e \* x \* x – 1.5 = 0

e \* x \* x = 1.5 – cos(x) \* ln(1 / x)

x = (1.5 – cos(x) \* ln(1 / x)) / x / e

F3(x) = (1.5 – cos(x) \* ln(1 / x)) / x / e;



Черная линия: y = x

Синяя линия: F3(x)

Видно по графикам, что условие сходимости выполняется(на заданном промежутке)

3) Метод Ньютона

Метод Ньютона – Суть метода состоит в разбиении отрезка на два отрезка с помощью касательной и выборе нового отрезка от точки пересечения касательной с осью абсцисс до неподвижной точки, на которой функция меняет знак и содержит решение.

Построение касательных продолжается до достижения необходимой точности. Член последовательности вычисляется таким образом:

x = x – f(x) / fp(x), где fp(x) – это производная функции.

Как в методе итераций образуется последовательность x\_n, и изменения до тех пора (fabs(x(n) – x(n-1)) > eps)

Вычислим производные функций:

Fp1 = (f1)’ = (0.25 \* x \* x \* x + x – 1.2502)’ = 0.75 \* x \* x + 1;

Fp2 = (f2)’ = (x \* x + ln(x) – e \* x)’ = 2 \* x + 1 / x – e;

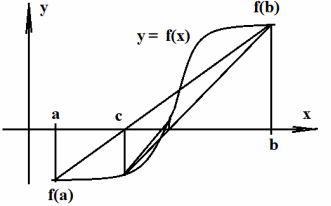
Fp3 = (f3)’ = (cos(x) \* ln(1 / x) + e \* x \* x – 1.5)’ = -sin(x) \* ln(1 / x) – cos(x) \* x / (x \* x) + 2 \* e \* x;

4) Метод хорд

В отличие от метода половинного деления, метод хорд предлагает, что деление рассматриваемого интервала будет выполняться не в его середине, а в точке пересечения хорды с осью абсцисс (ось - Х). Следует отметить, что под хордой понимается отрезок, который проведен через точки рассматриваемой функции по концам рассматриваемого интервала. Рассматриваемый метод обеспечивает более быстрое нахождение корня, чем метод половинного деления, при условии задания одинакового рассматриваемого интервала.

Тогда рассматриваемое с(благодаря которому будет сдвигаться граница) можно будет выразить из:

f(b)/(b – c) = f(a) / (c – a) (a, b – границы, с – точка пересечения хорд)

c = (f(a) \* b – f(b) \* a) / (f(a) – f(b))

Если f(a) \* f(b) > 0, тогда a = c, иначе b = c;

Это будет происходить до тех пор, пока fabs(c) > eps

**7. Сценарий выполнения работы** [план работы, первоначальный текст программы в черновике (можно на отдельном листе) и тесты либо соображения по тестированию].

#include <stdio.h>

#include <cmath>

#include <locale.h>

const double eps = 0.000001;

double dichotomy(double f(double), double, double);

double iteration(double f(double), double, double);

double tangent(double f(double), double F(double), double, double);

double chord(double f(double), double, double);

double f1(double);

double F1(double);

double Fp1(double);

double f2(double);

double F2(double);

double Fp2(double);

double f3(double);

double F3(double);

double Fp3(double);

int main(void){

setlocale(LC\_ALL, "Rus");

printf("Корень функции f1 методом деления пополам = %.5f\n", dichotomy(f1,0.,2.));

printf("Корень функции f1 методом итераций = %.5f\n", iteration(F1, 0., 2.));

printf("Корень функции f1 методом касательных = %.5f\n", tangent(f1,Fp1,0.,2.));

printf("Корень функции f1 методом хорд = %.5f\n\n", chord(f1,0.,2.));

printf("Корень функции f2 методом деления пополам = %.5f\n", dichotomy(f2,1.5,3.));

printf("Корень функции f2 методом итераций = %.5f\n", iteration(F2, 1.5, 3.));

printf("Корень функции f2 методом касательных = %.5f\n", tangent(f2,Fp2,1.5, 3.));

printf("Корень функции f2 методом хорд = %.5f\n\n", chord(f2,1.5, 3.));

printf("Корень функции f3 методом деления пополам = %.5f\n", dichotomy(f3,0.5,0.9));

printf("Корень функции f3 методом итераций = %.5f\n", iteration(F3, 0.5, 0.9));

printf("Корень функции f3 методом касательных = %.5f\n", tangent(f3,Fp3,0.5, 0.9));

printf("Корень функции f3 методом хорд = %.5f\n\n", chord(f3,0.5, 0.9));

return 0;

}

double dichotomy(double f(double), double a, double b){

double prevX = b, x = (a + b) / 2.;

while (fabs(prevX - x) > eps){

if (f(x) \* f(a) > 0)

a = x;

else

b = x;

prevX = x;

x = (a + b) / 2.;

}

return x;

}

double iteration(double f(double), double a, double b){

double prevX = a, x = f(prevX);

while (fabs(x - prevX) > eps){

prevX = x;

x = f(x);

}

return x;

}

double tangent(double f(double), double F(double), double a, double b){

double prevX = (a + b / 2.), x = prevX - f(prevX) / F(prevX);

while (fabs(prevX - x) > eps){

prevX = x;

x = prevX - f(prevX) / F(prevX);

}

return x;

}

double chord(double f(double), double a, double b){

double prevX = b, ya = f(a), yb = f(b);

double x = (ya \* b - yb \* a) / (ya - yb);

while (fabs(prevX - x) > eps){

if (ya \* f(x) > 0)

a = x;

else

b = x;

ya = f(a), yb = f(b);

prevX = x;

x = (ya \* b - yb \* a) / (ya - yb);

}

return x;

}

double f1(double x){

return 0.25 \* x \* x \* x + x - 1.2502;

}

double F1(double x){

return 1.2502 - 0.25 \* x \* x \* x;

}

double Fp1(double x){

return 0.75 \* x \* x + 1;

}

double f2(double x){

return x \* x + log(x) - exp(1.) \* x;

}

double F2(double x){

return ((exp(1.) \* x - log(x)) / x);

}

double Fp2(double x){

return (2 \* x + 1 / x - exp(1.));

}

double f3(double x){

return cos(x) \* log(1./x) + exp(1.) \* x \* x - 1.5;

}

double F3(double x){

return (1.5 - cos(x) \* log(1./x)) / x / exp(1.);

}

double Fp3(double x){

return -sin(x) \* log(1./x) - cos(x) \* x / (x \* x) + 2 \* exp(1.) \* x;

}

Соображения по тестированию:

на 1 вариант уже есть ответ — можно проверить по нему

2 и 3 варианты — можно подставить в уравнение и вычислить в калькуляторе

Так же, все методы должны дать одинаковый ответ.

*Допущен к выполнению работы.*  **Подпись преподавателя \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**8. Распечатка протокола**  (подклеить листинг окончательного варианта программы с тестовыми примерами, подписанный преподавателем).

[alex@alex kp4(?)]$ cat head.txt

-----------------------------------------------------

| Курсовая работа |

| Численные методы решения уравнений |

| Выполнил: студент группы М8О-101Б-21 |

| Постнов Александр Вячеславович |

-----------------------------------------------------

[alex@alex kp4(?)]$ cat main.cpp

#include <stdio.h>

#include <cmath>

#include <locale.h>

const double eps = 0.000001;

double dichotomy(double f(double), double, double);

double iteration(double f(double), double, double);

double tangent(double f(double), double F(double), double, double);

double chord(double f(double), double, double);

double f1(double);

double F1(double);

double Fp1(double);

double f2(double);

double F2(double);

double Fp2(double);

double f3(double);

double F3(double);

double Fp3(double);

int main(void){

setlocale(LC\_ALL, "Rus");

printf("Корень функции f1 методом деления пополам = %.5f\n", dichotomy(f1,0.,2.));

printf("Корень функции f1 методом итераций = %.5f\n", iteration(F1, 0., 2.));

printf("Корень функции f1 методом касательных = %.5f\n", tangent(f1,Fp1,0.,2.));

printf("Корень функции f1 методом хорд = %.5f\n\n", chord(f1,0.,2.));

printf("Корень функции f2 методом деления пополам = %.5f\n", dichotomy(f2,1.5,3.));

printf("Корень функции f2 методом итераций = %.5f\n", iteration(F2, 1.5, 3.));

printf("Корень функции f2 методом касательных = %.5f\n", tangent(f2,Fp2,1.5, 3.));

printf("Корень функции f2 методом хорд = %.5f\n\n", chord(f2,1.5, 3.));

printf("Корень функции f3 методом деления пополам = %.5f\n", dichotomy(f3,0.5,0.9));

printf("Корень функции f3 методом итераций = %.5f\n", iteration(F3, 0.5, 0.9));

printf("Корень функции f3 методом касательных = %.5f\n", tangent(f3,Fp3,0.5, 0.9));

printf("Корень функции f3 методом хорд = %.5f\n\n", chord(f3,0.5, 0.9));

return 0;

}

double dichotomy(double f(double), double a, double b){

double prevX = b, x = (a + b) / 2.;

while (fabs(prevX - x) > eps){

if (f(x) \* f(a) > 0)

a = x;

else

b = x;

prevX = x;

x = (a + b) / 2.;

}

return x;

}

double iteration(double f(double), double a, double b){

double prevX = a, x = f(prevX);

while (fabs(x - prevX) > eps){

prevX = x;

x = f(x);

}

return x;

}

double tangent(double f(double), double F(double), double a, double b){

double prevX = (a + b / 2.), x = prevX - f(prevX) / F(prevX);

while (fabs(prevX - x) > eps){

prevX = x;

x = prevX - f(prevX) / F(prevX);

}

return x;

}

double chord(double f(double), double a, double b){

double prevX = b, ya = f(a), yb = f(b);

double x = (ya \* b - yb \* a) / (ya - yb);

while (fabs(prevX - x) > eps){

if (ya \* f(x) > 0)

a = x;

else

b = x;

ya = f(a), yb = f(b);

prevX = x;

x = (ya \* b - yb \* a) / (ya - yb);

}

return x;

}

double f1(double x){

return 0.25 \* x \* x \* x + x - 1.2502;

}

double F1(double x){

return 1.2502 - 0.25 \* x \* x \* x;

}

double Fp1(double x){

return 0.75 \* x \* x + 1;

}

double f2(double x){

return x \* x + log(x) - exp(1.) \* x;

}

double F2(double x){

return ((exp(1.) \* x - log(x)) / x);

}

double Fp2(double x){

return (2 \* x + 1 / x - exp(1.));

}

double f3(double x){

return cos(x) \* log(1./x) + exp(1.) \* x \* x - 1.5;

}

double F3(double x){

return (1.5 - cos(x) \* log(1./x)) / x / exp(1.);

}

double Fp3(double x){

return -sin(x) \* log(1./x) - cos(x) \* x / (x \* x) + 2 \* exp(1.) \* x;

}

[alex@alex kp4(?)]$ g++ main.cpp

[alex@alex kp4(?)]$ ./a.out

Корень функции f1 методом деления пополам = 1.00011

Корень функции f1 методом итераций = 1.00011

Корень функции f1 методом касательных = 1.00011

Корень функции f1 методом хорд = 1.00011

Корень функции f2 методом деления пополам = 2.35458

Корень функции f2 методом итераций = 2.35458

Корень функции f2 методом касательных = 2.35458

Корень функции f2 методом хорд = 2.35458

Корень функции f3 методом деления пополам = 0.65415

Корень функции f3 методом итераций = 0.65415

Корень функции f3 методом касательных = 0.65415

Корень функции f3 методом хорд = 0.65415

**9. Дневник отладки** должен содержать дату и время сеансов отладки и основные события (ошибки в сценарии и программе, нестандартные ситуации) и краткие комментарии к ним. В дневнике отладки приводятся сведения об использовании других ЭВМ, существенном участии преподавателя и других лиц в написании и отладке программы.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| № | Лаб. или дом. | Дата | Время | Событие | Действие по исправлению | Примечание |
| 1 | дом | 15.11.2021 | 09:45 | Не скомпилировалась программа, из-за того что не поставил ; в конец 50 строки | Поставил ; |  |
|  |  |  |  |  |  |  |

**10. Замечания автора :**

11.**Выводы**

\_\_В ходе курсовой работы я составил и отладил программу на СИ, изучил численные методы решения уравнений, а также узнал, что функции могут выступать в роле параметров функции. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Недочёты при выполнении задания могут быть устранены следующим образом: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись студента \_Постнов\_\_\_\_\_\_\_\_\_\_\_\_\_\_