«Отчет по П. 3.2 Простейшие классы, одиночное наследование» Егор Корнаухов \*\*\*. Вариант 10.

Составить описание класса многочлена вида ах2 + bх + с. Предусмотреть методы, реализующие:

• вычисление значения многочлена для заданного аргумента;

• вывод на экран описания многочлена. Написать программу, демонстрирующую все разработанные элементы класса.

Создать дочерний класс, реализующий операции сложения, вычитания и умножения многочленов с получением нового объекта-многочлена.В случае недопустимых значений полей выбрасываются исключения. Написать программу, демонстрирующую все разработанные элементы класса.

1. Постановка задачи:

Функции класса Equation:

public: Equation(); // Конструктор без параметров

public: ~ Equation() // Деструктор

public: void SetABC(); // Метод установки коэффициентов базового уравнения

public: void SetFindX(); // Метод поиска корней уравнения

public: void GetOutput(); // Метод вывода результата работы

Функции класса Second:

public: Second(); // Конструктор без параметров

public: ~Second (); // Деструктор

public: void SetABC(); // Метод установки коэффициентов базового уравнения

public: void SetFindX(); // Метод поиска корней уравнения

public: void GetOutput(); // Метод вывода результата работы

public: void Sum(Equation x1, Equation x2) // Метод суммы многочленов

public: void Dif(Equation x1, Equation x2); // Метод разности многочленов

public: void Mult(Equation x1, Equation x2) // Метод произведения многочленов

public: void OutputB(); // Метод вывода результата работы без корней для второй части задания

Входные данные + ограничения:

А) Переменная A. Тип double. Коэффициент базового уравнения А.

Исключения: строка.

Б) Переменная B. Тип double. Коэффициент базового уравнения B.

Исключения: строка.

В) Переменная C. Тип double. Коэффициент базового уравнения C.

Исключения: строка.

Г) Переменная A1. Тип double. Коэффициент второго базового уравнения А.

Исключения: строка.

Д) Переменная B1. Тип double. Коэффициент второго базового уравнения B.

Исключения: строка.

Е) Переменная C1. Тип double. Коэффициент второго базового уравнения C.

Исключения: строка.

Промежуточные данные + ограничения:

А) Переменная flag. Тип bool. Флаг о наличии корней.

Исключения: буквы, числа, дробные, отрицательные числа, строки.

Б) Переменная count. Тип int. Переменная количества корней.

Исключения: буквы, дробные, отрицательные числа, строки.

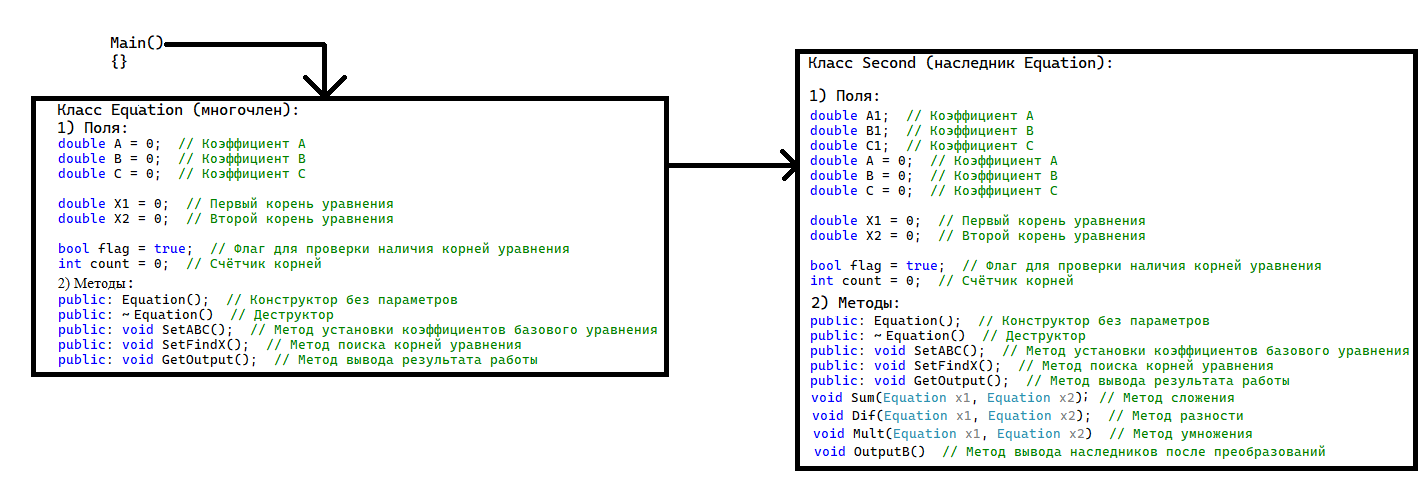
В) Переменная D. Тип double. Переменная дискриминант уравнения.

Исключения: буквы, строки.

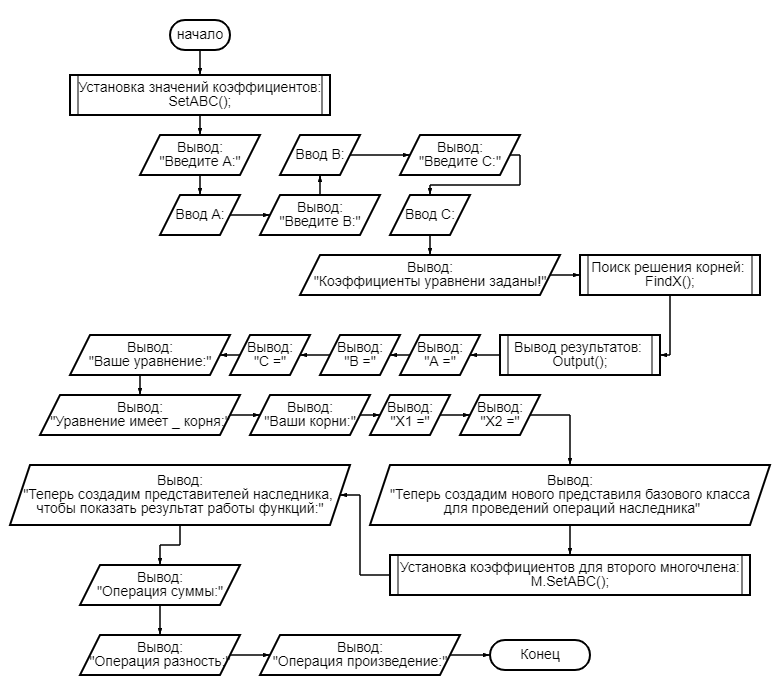
Среда разработки: Visual Studio 2022.

2. Разработка программы:

2.1 Описание классов, полей и методов классов.



2.2Разработка схемы алгоритма.



2.3 Описание пользовательского интерфейса.

«Взаимодействие пользователя с программой осуществляется с помощью диалога».

Примеры диалогов:

1. Ввод коэффициента базового уравнения А:

Вводимое значение коэффициента А – “Cat”



Вводимое значение коэффициента А – 5



1. Ввод коэффициента базового уравнения B:

Вводимое значение коэффициента B – “Dog”



Вводимое значение коэффициента B – 6

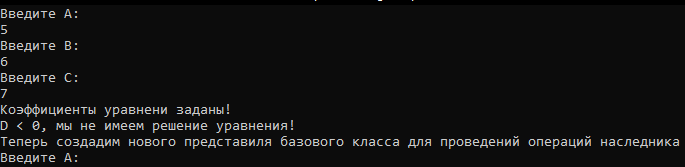


1. Ввод коэффициента базового уравнения C:

Вводимое значение коэффициента C – “Mouse”



Вводимое значение коэффициента B – 7



1. Ввод коэффициента второго базового уравнения А:

Вводимое значение коэффициента А – “Cat”



Вводимое значение коэффициента А – 5



1. Ввод коэффициента второго базового уравнения B:

Вводимое значение коэффициента B – “Dog”



Вводимое значение коэффициента B – 6

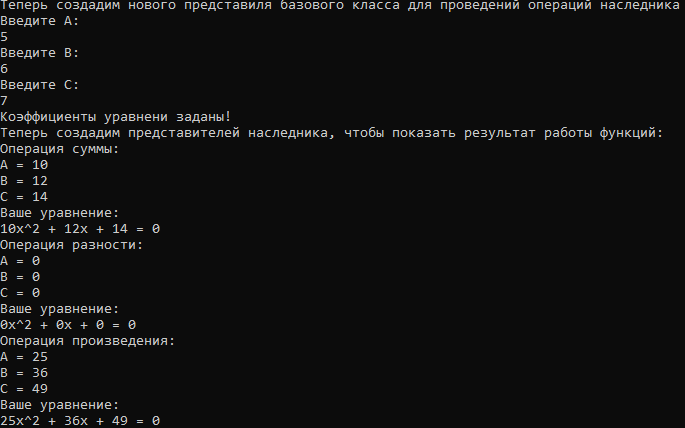


1. Ввод коэффициента базового уравнения C:

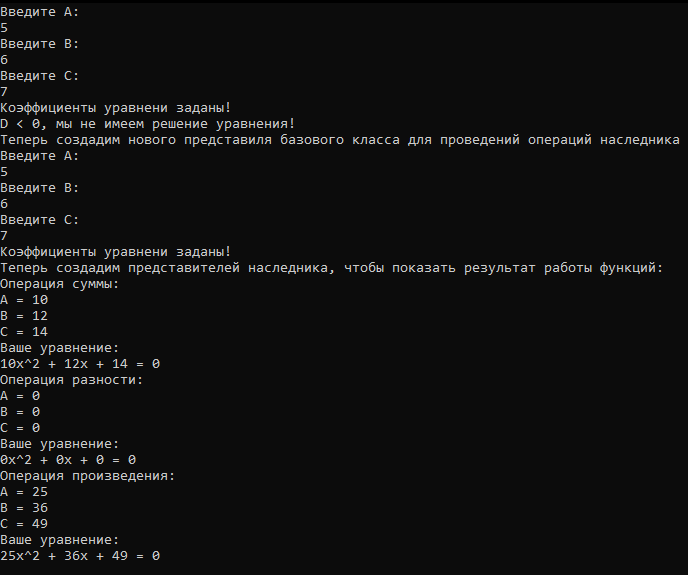
Вводимое значение коэффициента C – “Mouse”



Вводимое значение коэффициента B – 7



Итог работы всей программы:



3. Реализация и тестирование программы

**3.1. Описание разработанной программы**

Для отцовского класса у нас определены методы для установления и получения основных необходимых полей класса (Класс животного, количества конечностей, количества предков животного).

Для дочерного класса у нас определены всё те же методы, но добавлены поля с методы связанные с кличкой животного.

В функции main мы создаём представителя дочерного класса. Затем поочерёдно вызываем функции вывода основных полей представителя класса. Получаем ответы, что все поля пустые (что и должно быть в самом начале программы). Потом поочерёдно вызываем функции для записи информации в основные поля класса. После чего выводим значения всех полей в консоль.

Дальше мы работаем с методами дочерного класса. Вызываем методы для установки значений полей дочерного класса и выводим с помощью иных методов установленные нами значения.

**3.2. Тестирование программы**

|  |  |  |
| --- | --- | --- |
| Входные данные | Результат программы | Смысл |
| N.SetABC()  “Cat” 1 5 | Некорректный ввод! | Проверка на строку |
| N.SetABC()  5 5 6 | Коэффициенты уравнени заданы!  D < 0, мы не имеем решение уравнения!  Теперь создадим нового представиля базового класса для проведений операций наследника | Проверка на корректность работы программы |
| N.SetABC()  -5 5.5 6 | Коэффициенты уравнени заданы!  A = -5  B = 5.6  C = 6  Ваше уравнение:  -5x^2 + 5.6x + 6 = 0  Уравнение имеет 2 корня:  Ваши корни:  X1 = -0.670285  X2 = 1.79028  Теперь создадим нового представиля базового класса для проведений операций наследника | Проверка на иной случай ветвления решения |
| Cat.SetCountLimbs();  (5.5) | Incorrect input! | Проверка на дробное число |
| M.SetABC();  1 2 3 | Коэффициенты уравнени заданы!  Теперь создадим представителей наследника, чтобы показать результат работы функций:  Операция суммы:  A = -4  B = 7.6  C = 9  Ваше уравнение:  -4x^2 + 7.6x + 9 = 0  Операция разности:  A = -6  B = 3.6  C = 3  Ваше уравнение:  -6x^2 + 3.6x + 3 = 0  Операция произведения:  A = -5  B = 11.2  C = 18  Ваше уравнение:  -5x^2 + 11.2x + 18 = 0 | Проверка на корректность работы программы |
| M.SetABC();  5.5 “sk” | Incorrect input! | Проверка на строку |

Возможный диалог:

Ввод коэффициентов базового уравнения A – ввод коэффициентов базового уравнения B - ввод коэффициентов базового уравнения C - ввод коэффициентов второго базового уравнения A – ввод коэффициентов второго базового уравнения B - ввод коэффициентов второго базового уравнения C.

Код:

#include <iostream> // Стандартная библиотека ввода-вывода

using namespace std; // Использования стандартного пространства имён

void InputDouble(double\*); // Прототип функции проверки на корректность ввода

class Equation // Базовый класс для работы с уравнением

{

public:

double A = 0; // Коэффициент А

double B = 0; // Коэффициент В

double C = 0; // Коэффициент С

double X1 = 0; // Первый корень уравнения

double X2 = 0; // Второй корень уравнения

bool flag = true; // Флаг для проверки наличия корней уравнения

int count = 0; // Счётчик корней

public:

void SetABC() // Метод установки коэффициентов

{

cout << "Введите A:" << endl;

InputDouble(&A); // Вызов функции проверки ввода коэффициента

cout << "Введите B:" << endl;

InputDouble(&B); // Вызов функции проверки ввода коэффициента

cout << "Введите C:" << endl;

InputDouble(&C); // Вызов функции проверки ввода коэффициента

cout << "Коэффициенты уравнени заданы!" << endl;

}

void FindX() // Функция поиска корней уравнения

{

if (A == 0 && B == 0 && C != 0) // Условия случая 0 0 C

{

cout << "Уравнение не имеет решения!" << endl;

flag = false;

}

if ((B != 0) && (A == 0) && (C == 0) && (flag == true)) // Условия случая 0 B 0

{

X1 = 0;

count = 1;

}

if ((B == 0) && (C == 0) && (A != 0) && (flag == true)) // Условия случая A 0 0

{

X1 = 0;

count = 1;

}

if ((C == 0) && (A != 0) && (B != 0) && (flag == true)) // Условия случая A B 0

{

X1 = 0;

X2 = -B / A;

count = 2;

}

if ((A == 0 && B == 0 && C == 0) && (flag == true)) // Условия случая 0 0 C

{

cout << "Нет решения!" << endl;

flag = false;

}

if ((B == 0) && (A != 0) && (C != 0) && (flag == true)) // Условия случая A 0 C

{

if ((-C / A) < 0)

{

cout << "Уравнение не имеет решений!" << endl;

flag = false;

}

else

{

X1 = sqrt((-C / A));

X2 = -X1;

count = 2;

}

}

if ((A == 0) && (B != 0) && (C != 0) && (flag == true)) // Условия случая 0 B C

{

X1 = -C / B;

count = 1;

}

if ((A != 0) && (B != 0) && (C != 0) && (flag == true)) // Условия случая A B C

{

double D = 0; // Поле дискриминант

D = sqrt((B \* B) - (4 \* A \* C)); // Поиск дискриминанта

if (D > 0) // Проверка на наличие корней

{

X1 = (-B + D) / (2 \* A);

X2 = (-B - D) / (2 \* A);

count = 2;

}

else

{

if (D == 0)

{

X1 = -B / (2 \* A);

count = 1;

}

else

{

cout << "D < 0, мы не имеем решение уравнения!" << endl;

flag = false;

}

}

}

}

public:

void Output() // Функция вывода ответа на задачу

{

if (flag == true)

{

cout << "A = " << A << endl;

cout << "B = " << B << endl;

cout << "C = " << C << endl;

cout << "Ваше уравнение:" << endl;

cout << A << "x^2 + " << B << "x + " << C << " = 0" << endl;

if (count == 2)

{

cout << "Уравнение имеет 2 корня:" << endl;

cout << "Ваши корни:" << endl;

cout << "X1" <<" = " << X1 << endl;

cout << "X2" << " = " << X2 << endl;

}

else

{

if (count == 1)

{

cout << "Уравнение имеет 1 корень:" << endl;

cout << "Ваш корень:" << endl;

cout << "X" << " = " << X1 << endl;

}

}

}

}

};

class Second : public Equation // Класс-наследник

{

double A1; // Коэффициент А

double B1; // Коэффициент В

double C1; // Коэффициент С

public:

void Sum(Equation x1, Equation x2) // Метод сложения

{

A1 = x1.A + x2.A;

B1 = x1.B + x2.B;

C1 = x1.C + x2.C;

}

void Dif(Equation x1, Equation x2) // Метод разности

{

A1 = x1.A - x2.A;

B1 = x1.B - x2.B;

C1 = x1.C - x2.C;

}

void Mult(Equation x1, Equation x2) // Метод умножения

{

A1 = x1.A \* x2.A;

B1 = x1.B \* x2.B;

C1 = x1.C \* x2.C;

}

void OutputB() // Метод вывода наследников после преобразований

{

cout << "A = " << A1 << endl;

cout << "B = " << B1 << endl;

cout << "C = " << C1 << endl;

cout << "Ваше уравнение:" << endl;

cout << A1 << "x^2 + " << B1 << "x + " << C1 << " = 0" << endl;

}

};

int main(void)

{

setlocale(LC\_ALL, "Russian"); // Установка русской кодировки в консоли

Equation N; // Создание базового представителя для первой части задания

N.SetABC(); // Вызов метода установки коэффициентов

N.FindX(); // Вызов метода поиска решения уравнения

N.Output(); // Вызов функции вывода результата

cout << "Теперь создадим нового представиля базового класса для проведений операций наследника" << endl;

Equation M; // Второй базовый представитель для работы методов наследника

M.SetABC(); // Вызов метода установки коэффициентов

cout << "Теперь создадим представителей наследника, чтобы показать результат работы функций:" << endl;

Second n1, n2, n3; // Создания представителей для сохранения результатов операций методов наследника

cout << "Операция суммы:" << endl;

n1.Sum(N, M); // Вызов метода суммы

n1.OutputB();

cout << "Операция разности:" << endl;

n2.Dif(N, M); // Вызов метода разности

n2.OutputB();

cout << "Операция произведения:" << endl;

n3.Mult(N, M); // Вызов метода произведения

n3.OutputB();

return 0;

}

void InputDouble(double\* tmp) // Функция проверки на корректность ввода значения коэффициентов

{

while (!(cin >> \*tmp) || (cin.peek() != '\n'))

{

cin.clear();

while (cin.get() != '\n');

cout << "Некорректный ввод!" << endl;

}

}