# Задание

# 

# 

# Диаграмма классов

Cat

Класс

->Animal

->IAnimal

Поля

Playful

Методы

Cat()

~Cat()

Show()

Hide()

Sniff()

Play()

IsPlayful()

Eat()

Hunt()

Animal

Класс

->Point

Поля

Name

Fed

Phrase

Методы

Animal()

~Animal()

isFed()

GetName()

Say()

Eat()

Location

Класс

Поля

X

Y

Методы

Location

~Location

Get\_X

Get\_Y

public

public

IAnimal

Интерфейс

Методы

Show()=0;

Hide()=0;

Hunt( )=0;

Eat( )=0;

public

public

public

public

Mouse

Класс

->Animal

->IAnimal

Методы

Mouse()

~Mouse()

Eat()

Hunt()

Show()

Hide()

Point

Класс

->Location

Поля

Visible

Методы

Point()

~Point()

IsVisible()

MoveTo()

Drag()

Show()

Hide()

public

public

Flower

Класс

->Point

Поля

Opened

Методы

Flower()

~Flower()

Show()

Hide()

IsOpened()

Turn()

public

Ball

Класс

->Point

Методы

Ball()

~Ball()

Show()

Hide()

Mink

Класс

->Point

Поля

Full

Методы

Mink()

~Mink()

Show()

Hide()

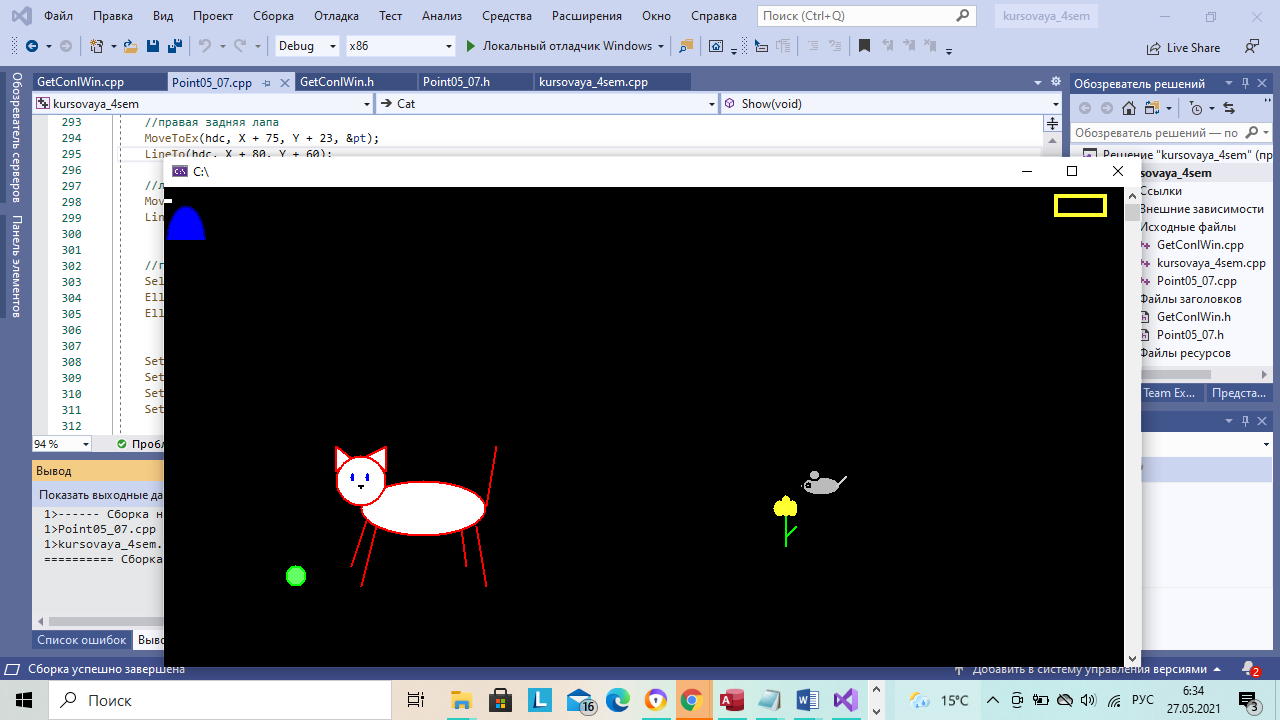
Fill()

IsFill()

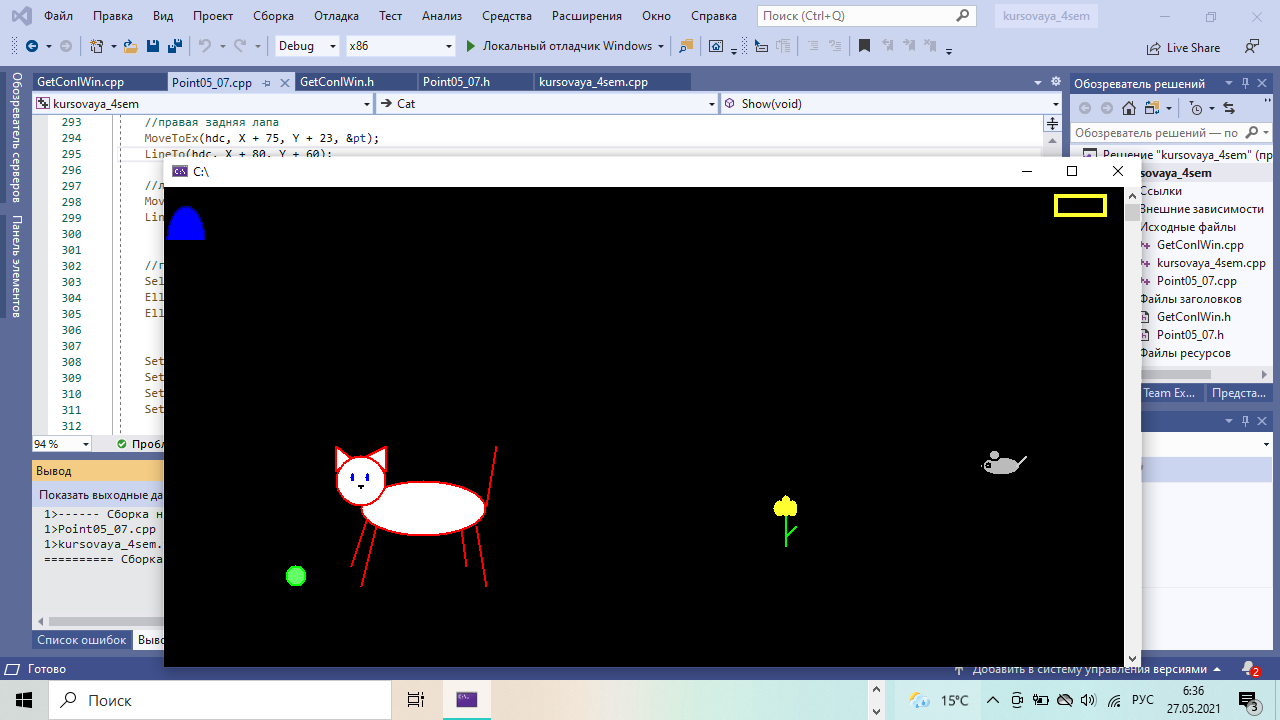
# 

# Результаты работы

Начало работы программы:

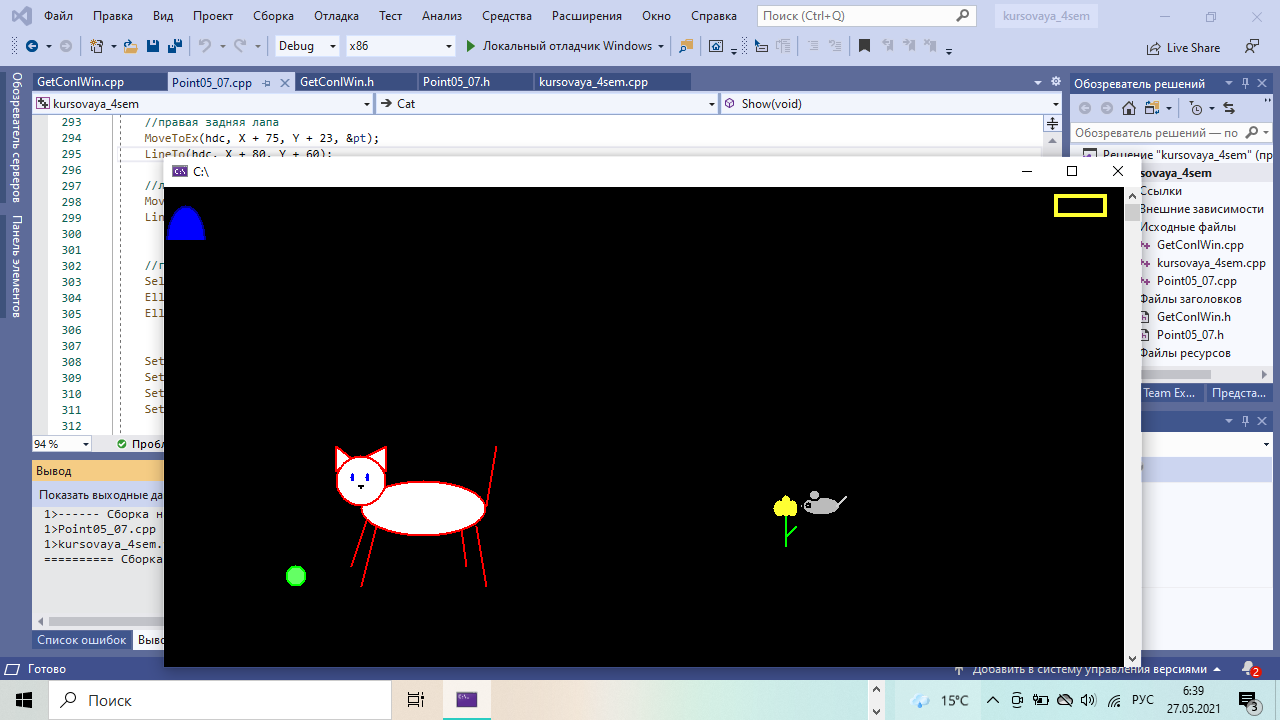


Нажали Enter(наш выбор для продолжения), а затем цифру 2 – выбрали мышку

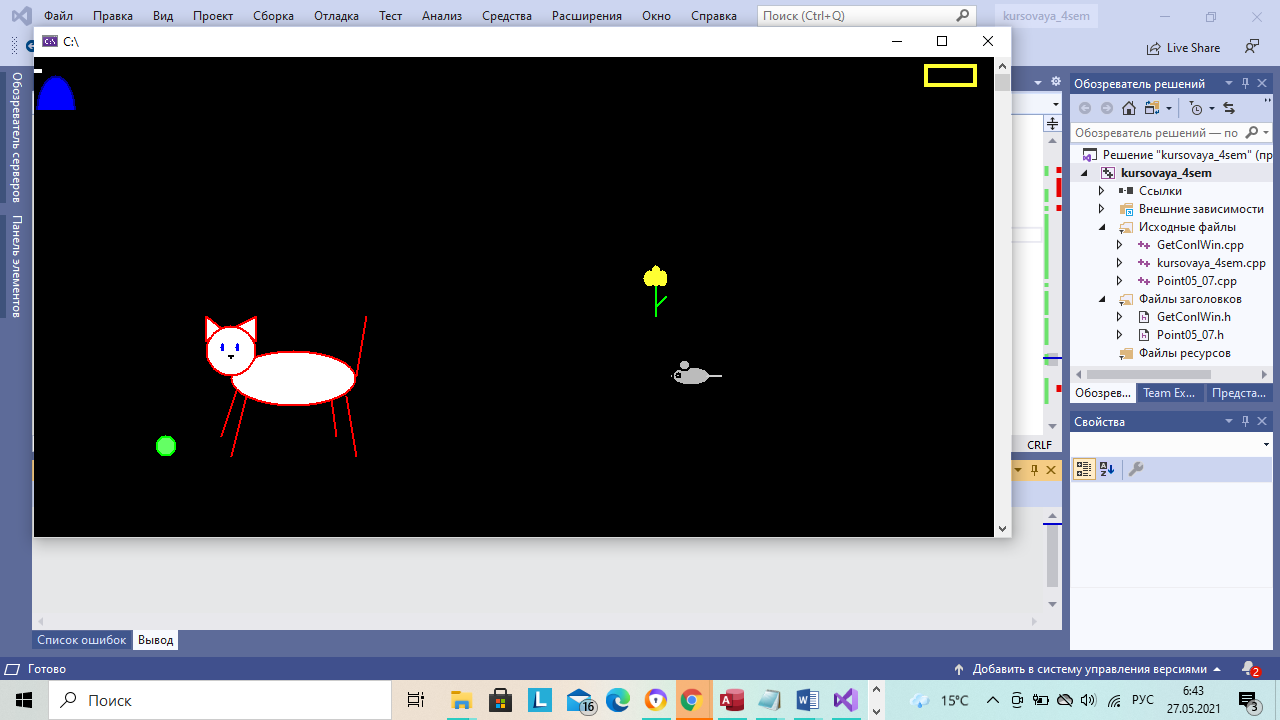


Переместили ее стрелками, нажали цифру “0”, и ничего не произошло – рядом нет ни кошки, ни цветка

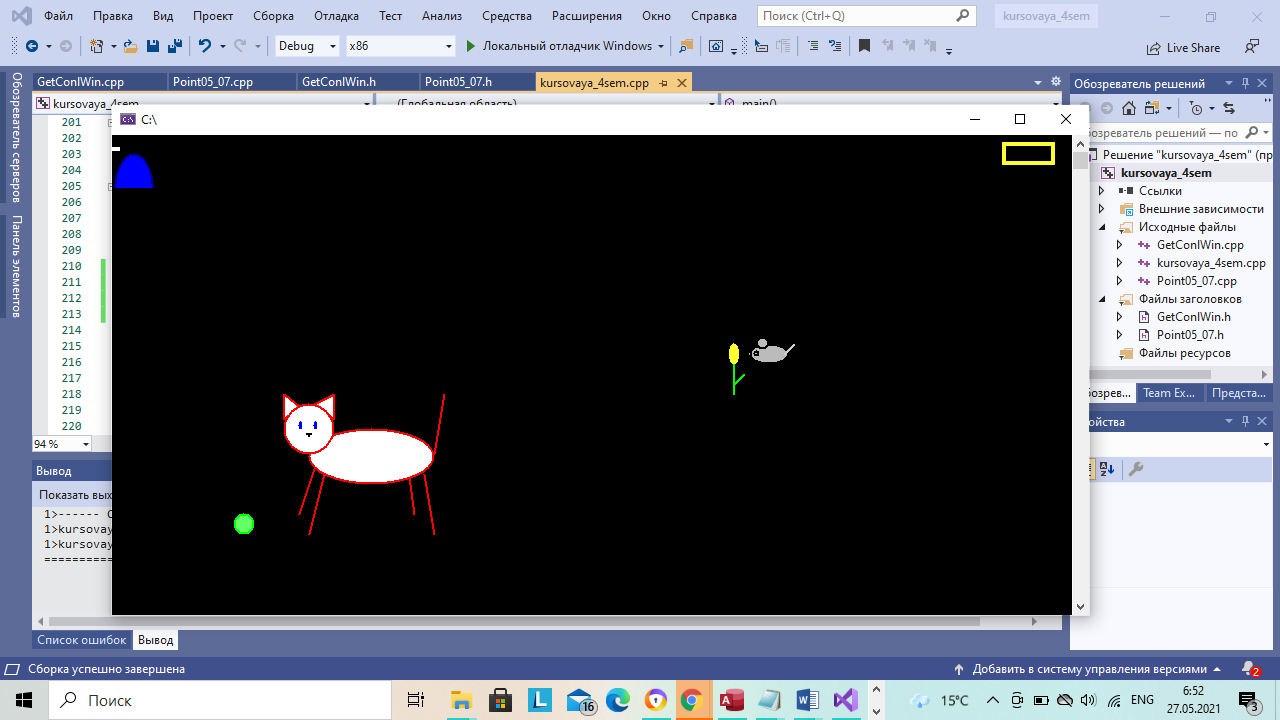
Далее опять выбираем мышку



Подошли к раскрытому цветку, мышка голодная

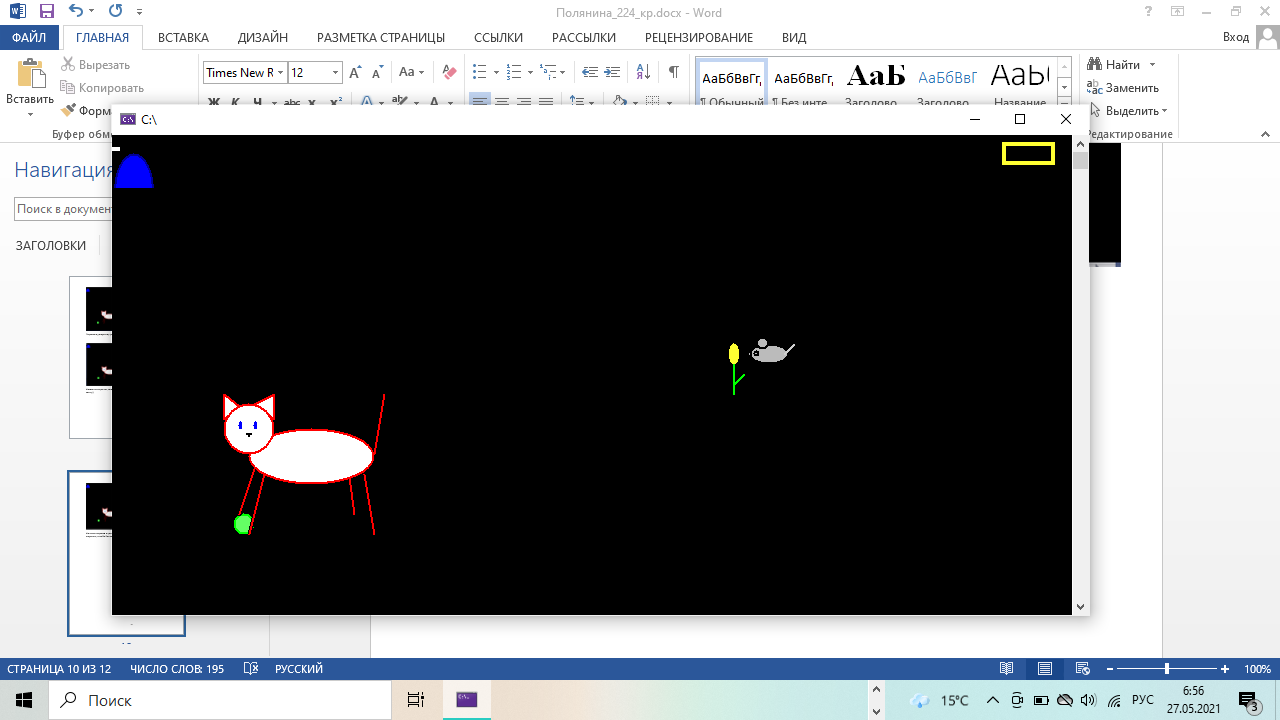


Мышка съела цветок, появился новый на другом месте, а мышка теперь сытая(видно по хвосту)

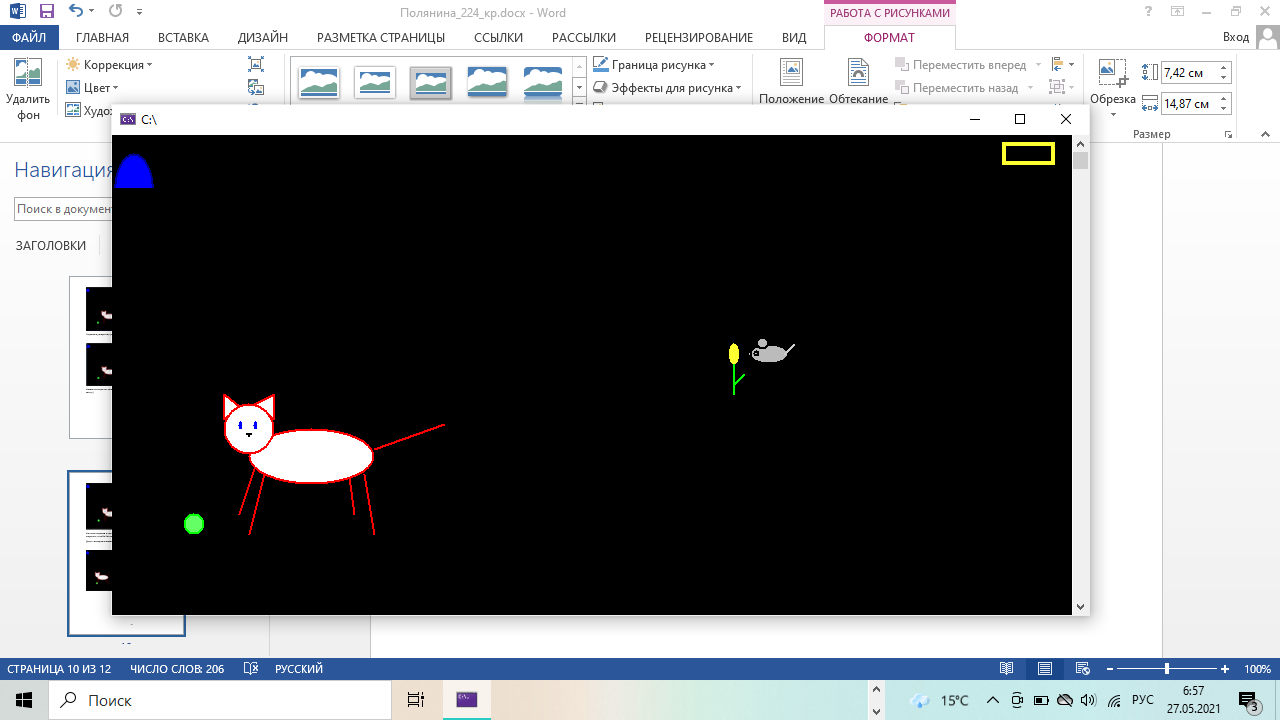


Мы опять подошли к цветку и мышка его “понюхала”, и теперь он закрылся(мог и открыться, если бы был закрыт), а мышка опять хочет есть

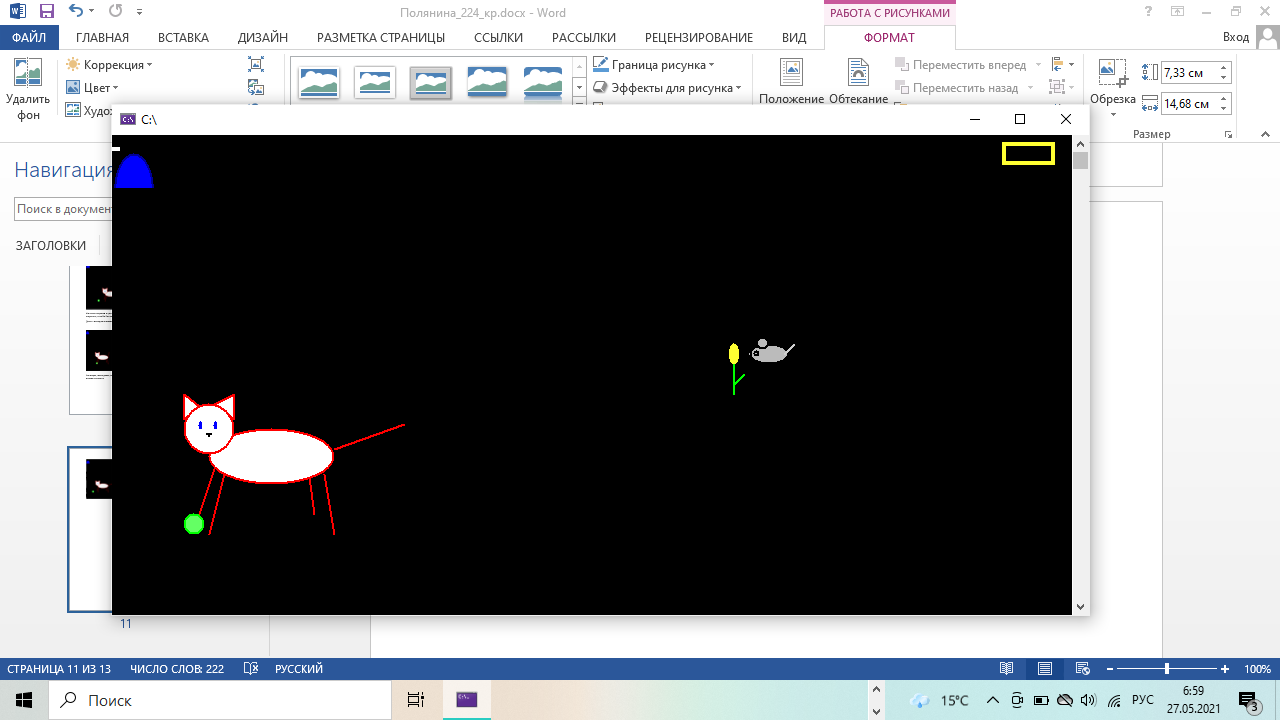
Далее с помощью клавиши “1” выбираем кошку и приближается к мячу



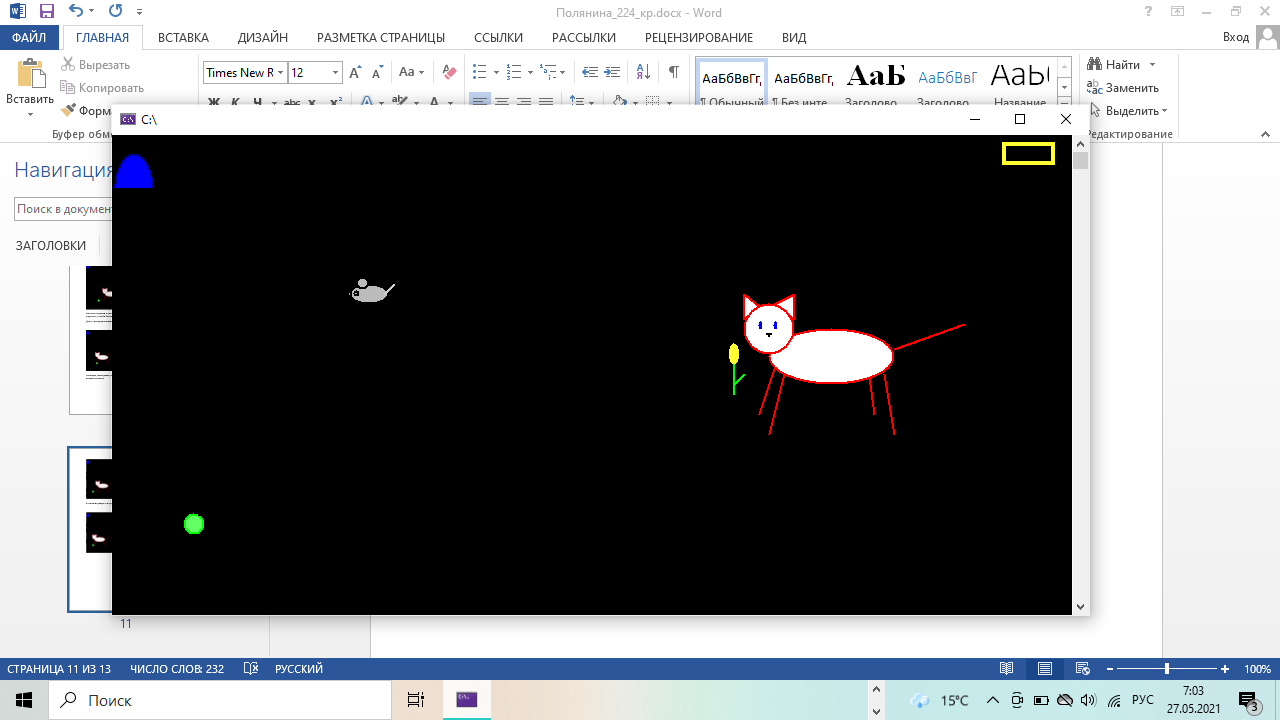
Как видно, она игривая, но когда мы закончим ходить(нажали “0”), она станет неиггривой, а мячик откатится



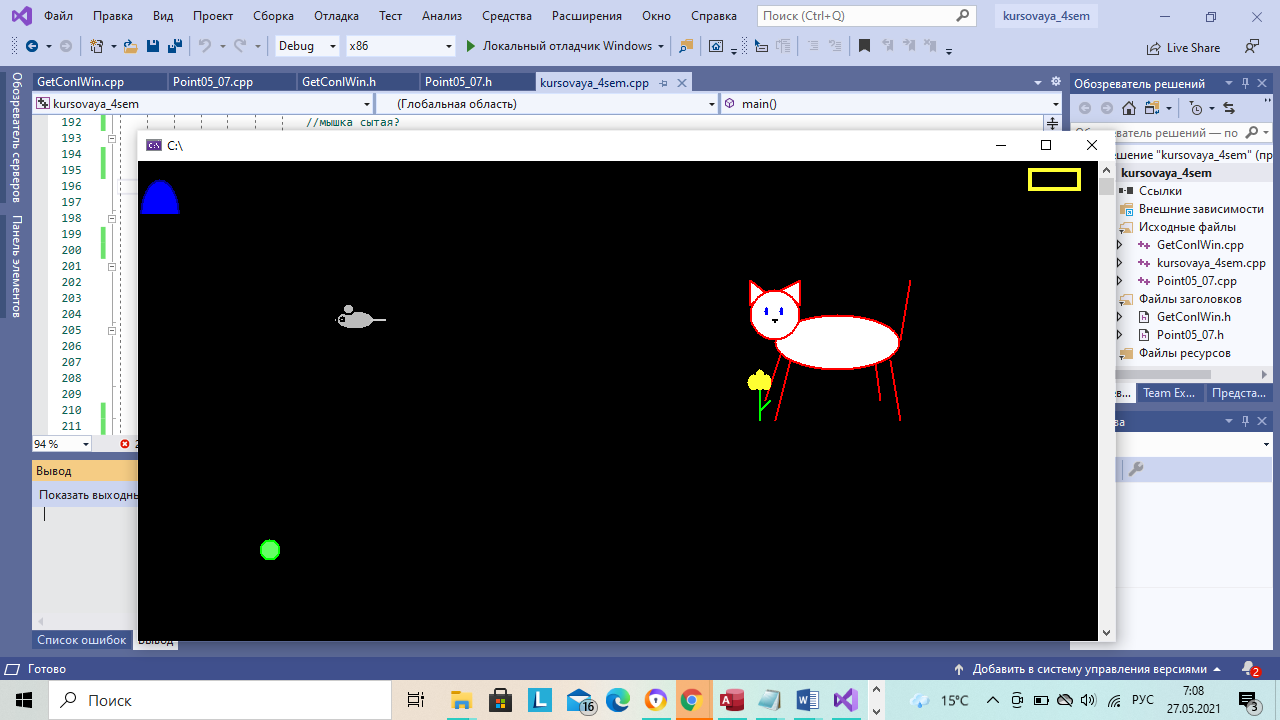
Если мы подойдем к мячу еще раз, ничего не произойдет

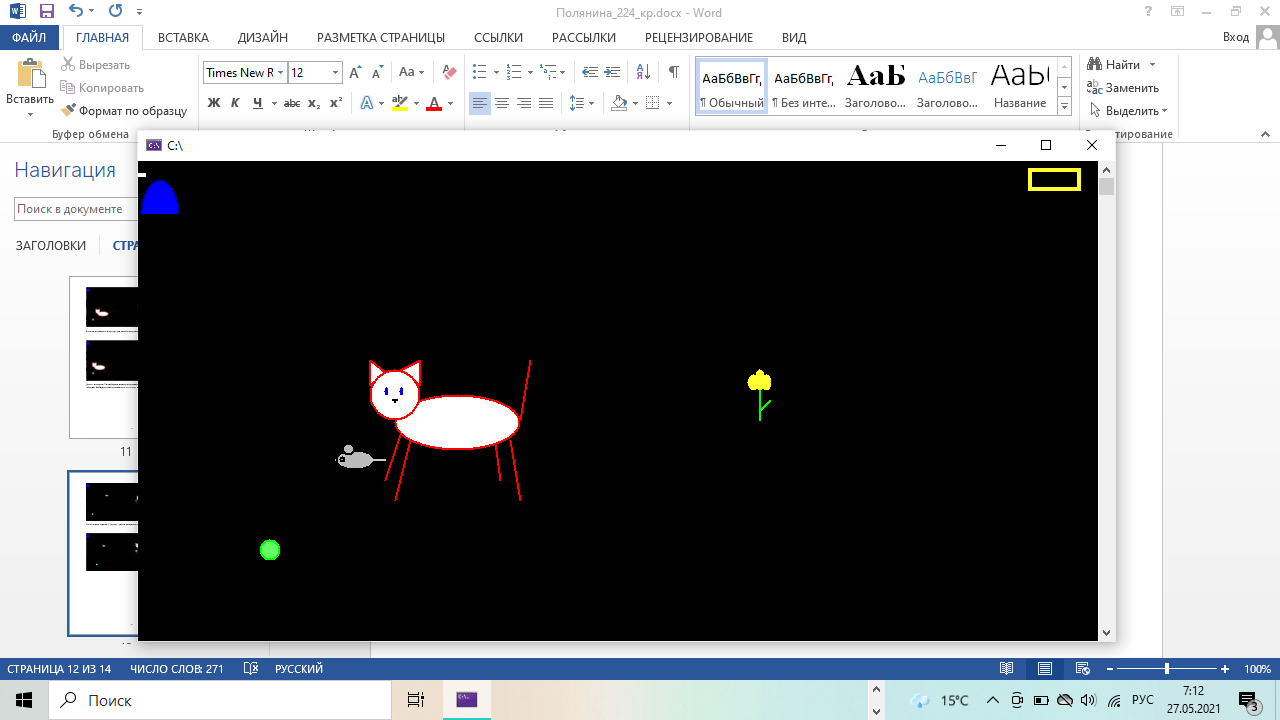


Далее с помощью Tab выбираем мышку и передвигаем ее туда, где она не сможет вызвать событий. Выбираем кошку и движемся то тех пор, пока не будем около цветка

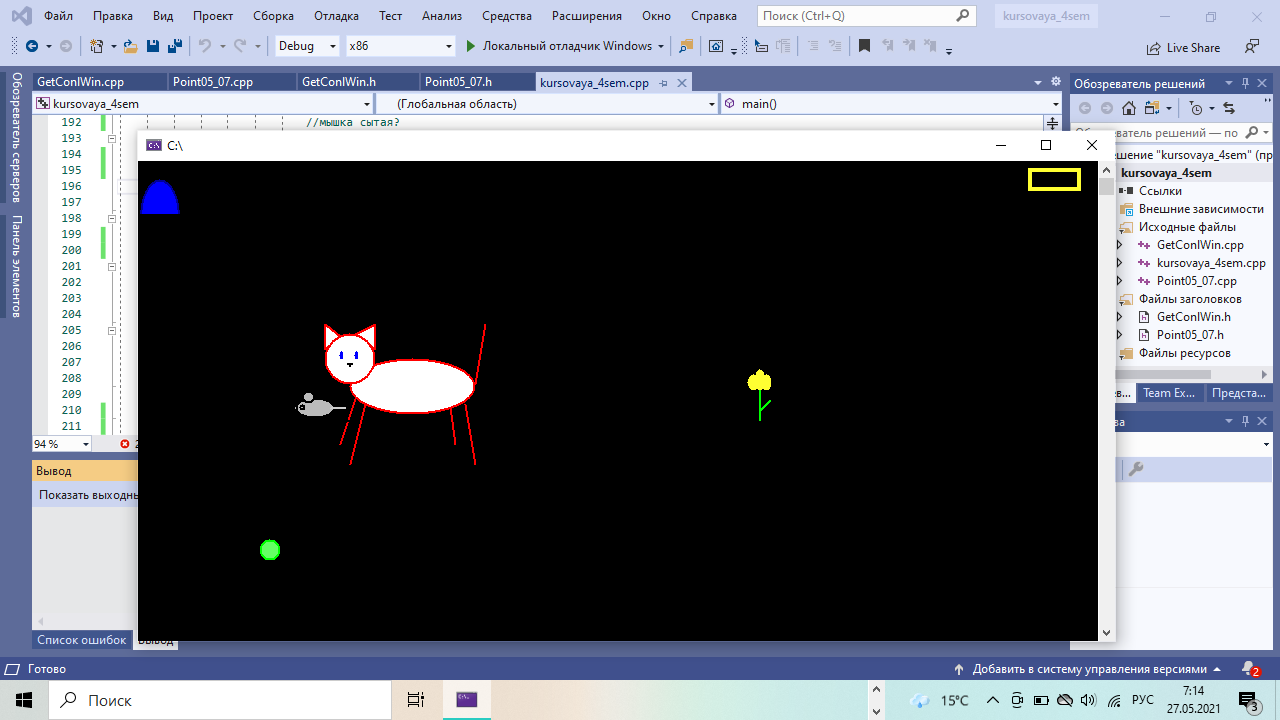


После кошка “вдыхает” цветок , цветок раскрылся, а кошка стала игривой

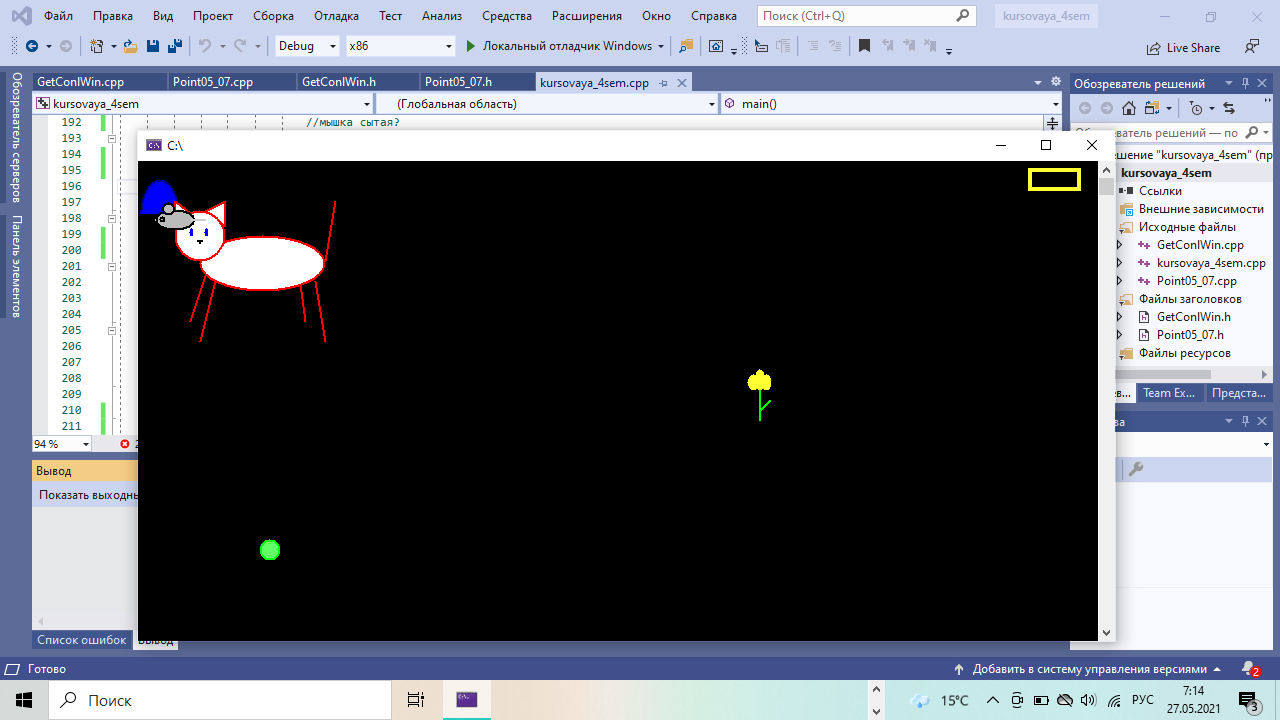


После передвинем кошку и мышку так, чтобы они были близко друг к другу

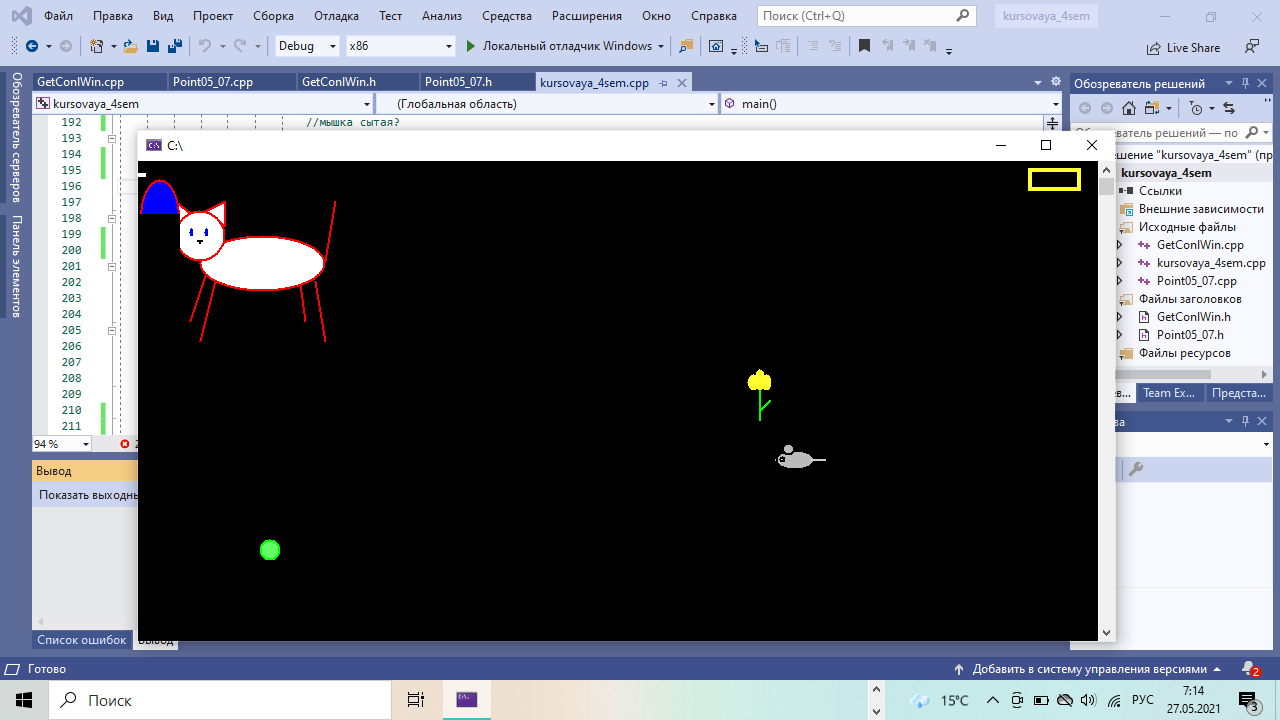
Началась охота: мышка и кошка побежали к норке



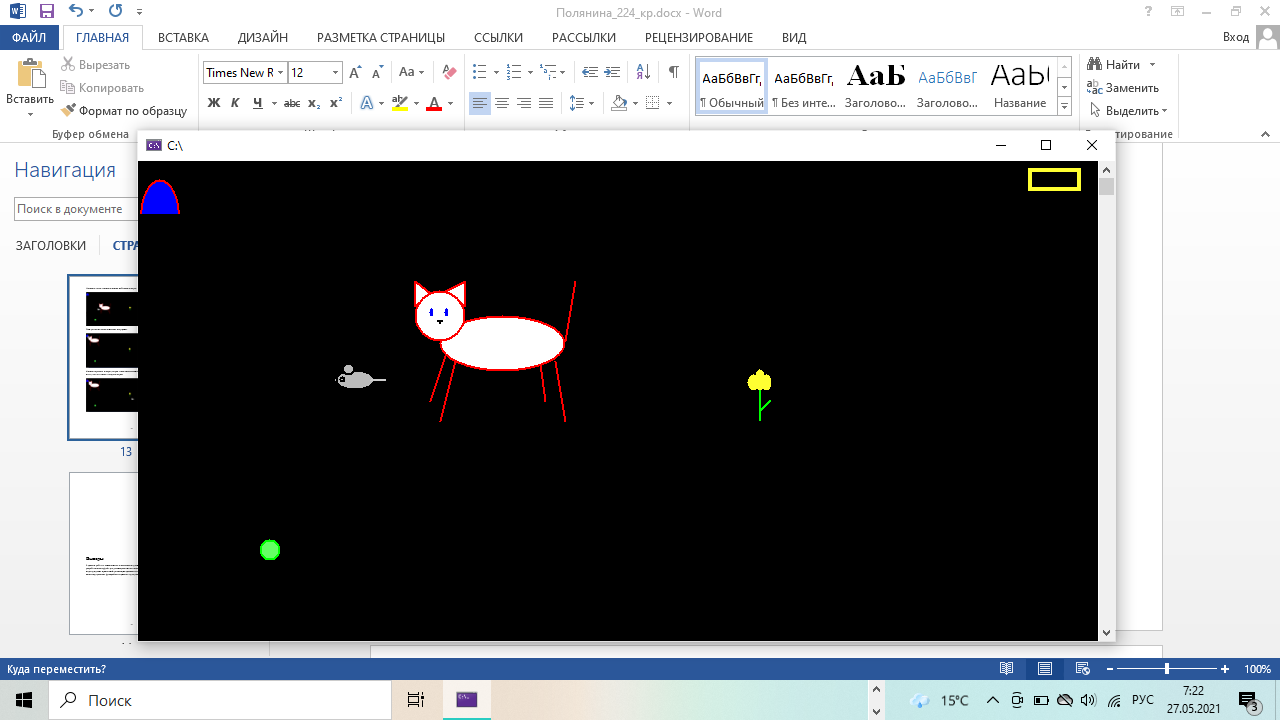
Они достигли своих конечных координат:



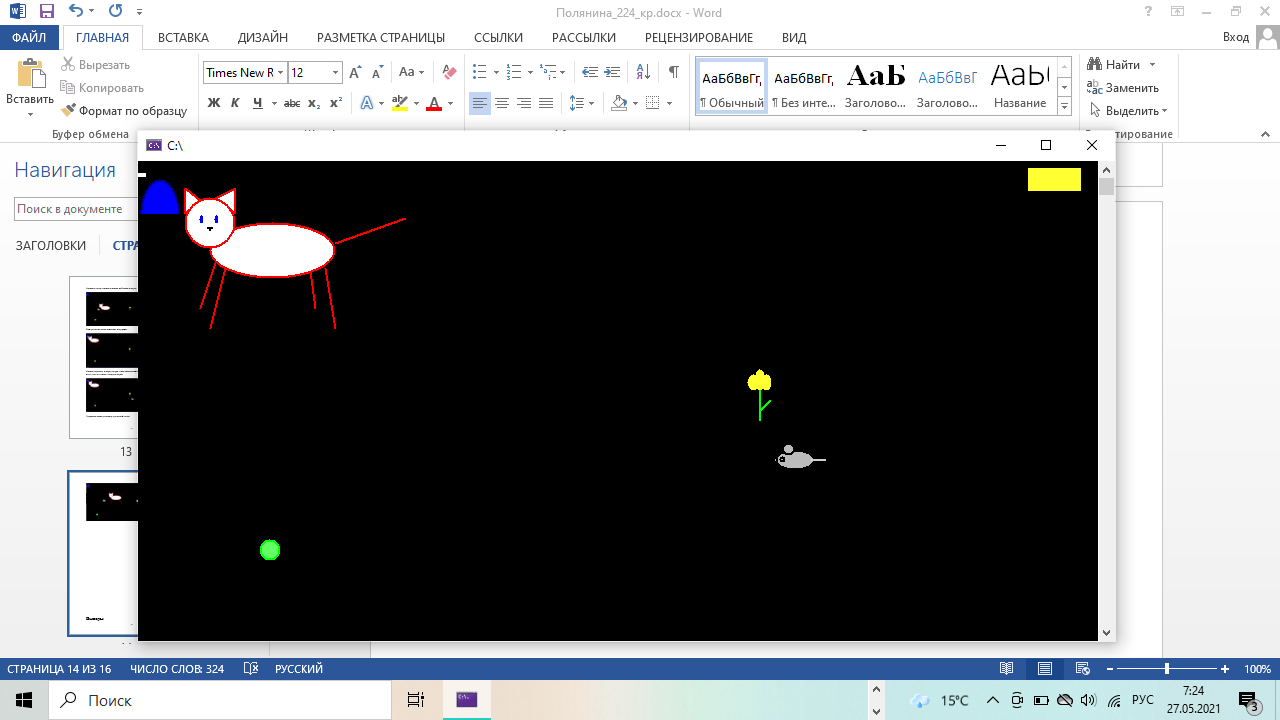
Мышка скрылась в норке, норка стала заполоненной, а новая мышка появилась в другом месте, кот все также голоден и игрив



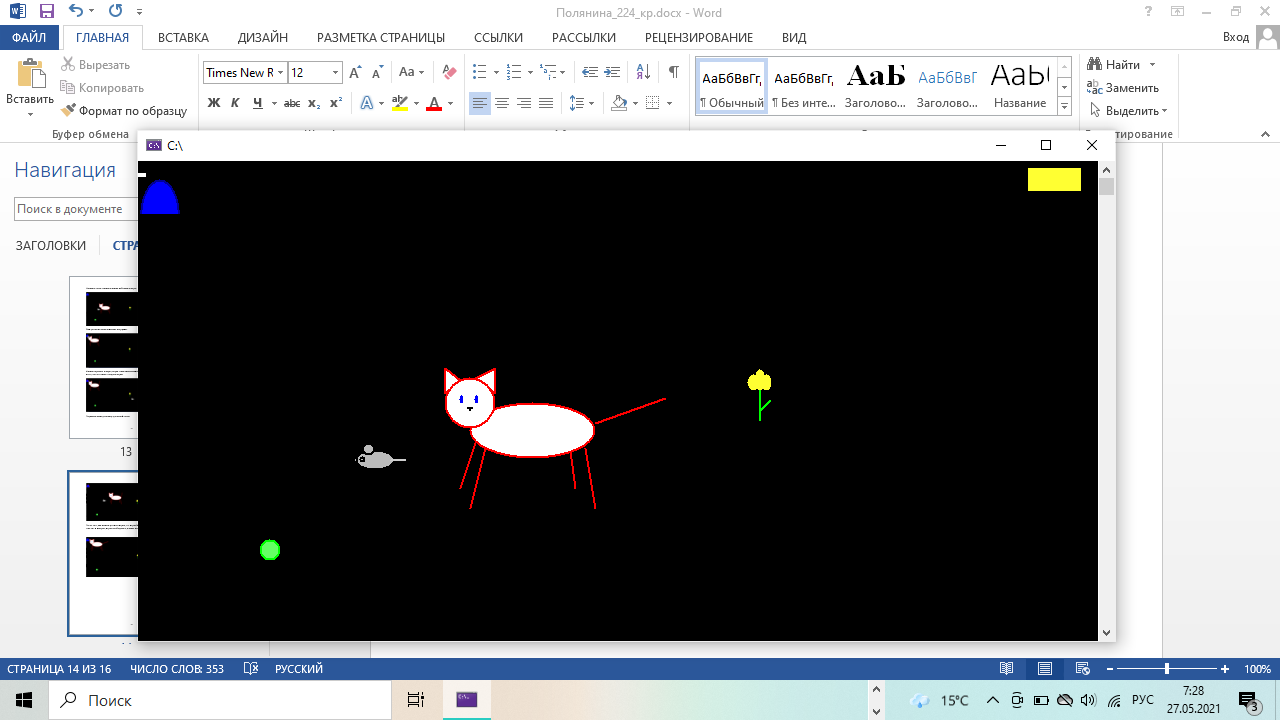
Подвинем мышку и кошку для новой охоты



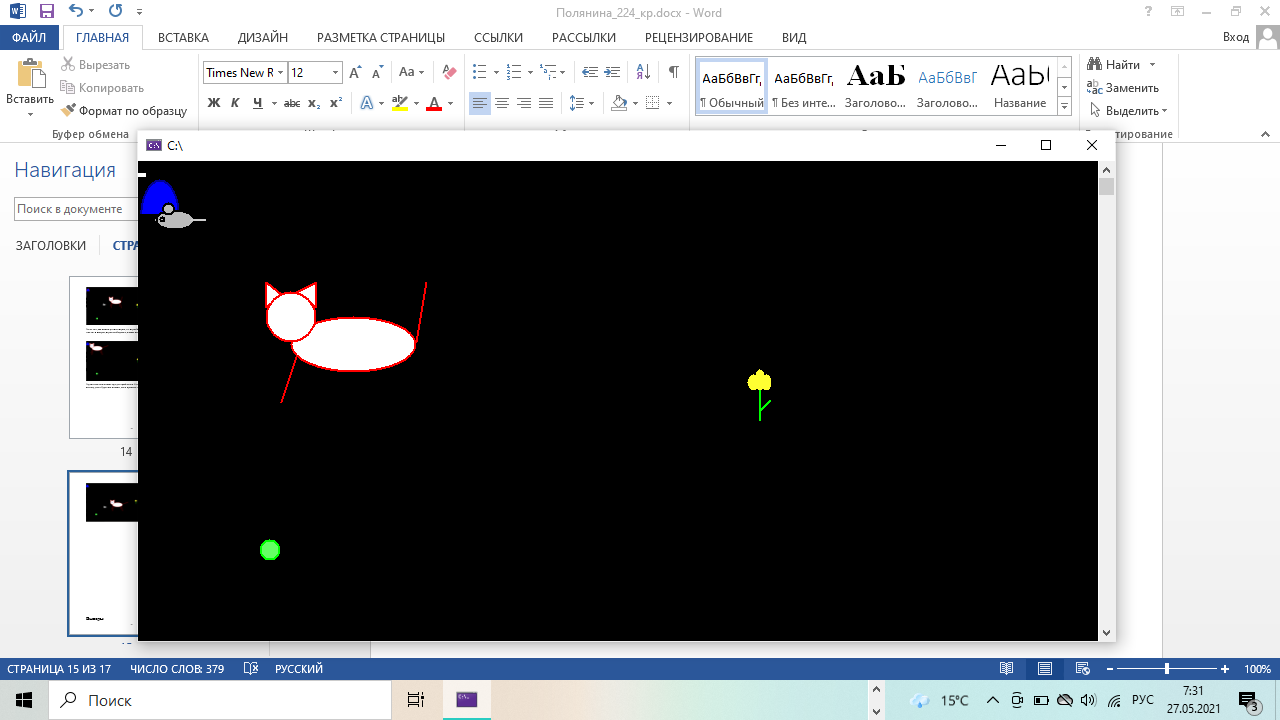
После того, как мышка достигла норки, т.к. норка была занята, а кот голоден, ее съели, кот стал сыт и неигрив, норка освободилась, а новая мышка появилась на новом месте



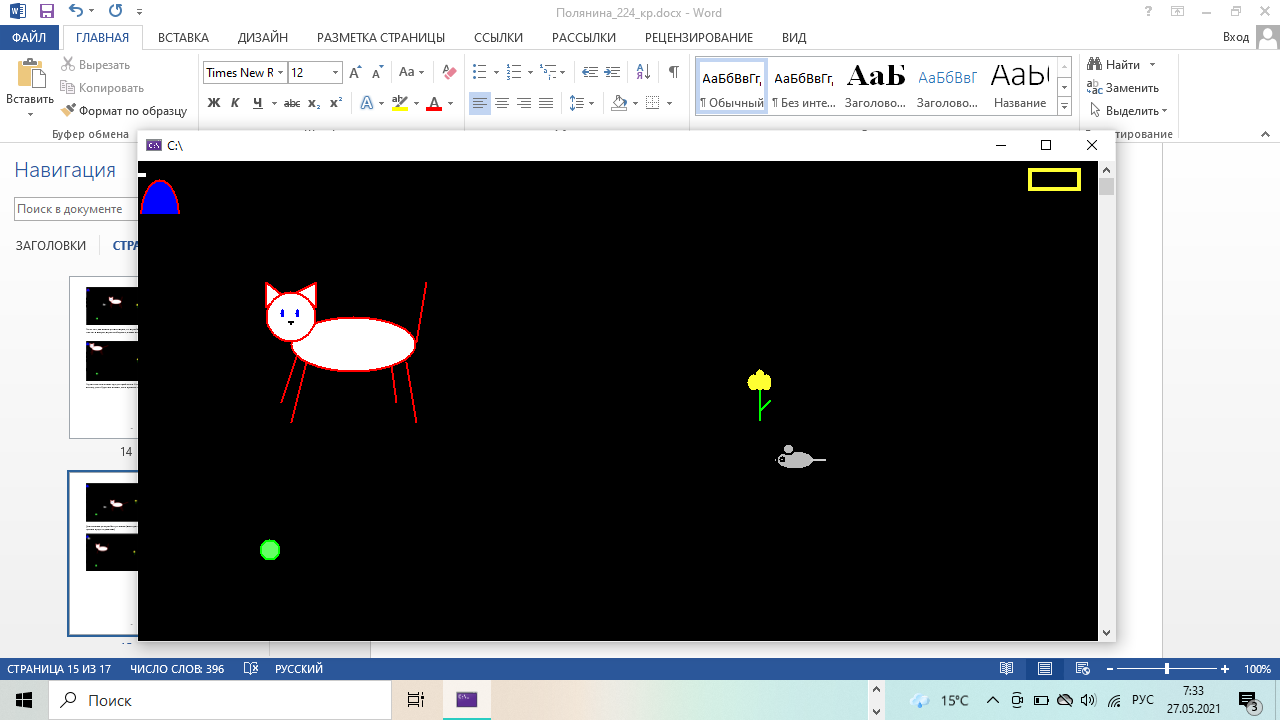
Переместим кота и мышь еще для одной охоты. В этот раз кот сытый и не игривый, поэтому у него будет шаг меньше, чем в прошлых случаях



Дошла мышка до норки быстрее кошки (некоторые детали кошки, в том числе и шкала, не видны, т.к. скрин сделан в процессе движения)



Норка стала заполнена, новая мышь появилась в другом месте, кошка оказалась в начальной позиции, стала голодна и игрива



# Выводы

В данном работе я ознакомилась и выполнила организацию наследования классов, разработала интерфейс для реализации множественного наследования, а для подтверждения правильной реализации динамического полиморфизма использовала вызов виртуальных функций наследников через указатель базового класса.