|  |  |
| --- | --- |
| Министерство образования и науки Республики Казахстан Северо-Казахстанский университет им. М. Козыбаева | |
| МОДУЛЬНЫЙ ПРОЕКТ 1  На тему: «**Объектно-ориентированное программирование и модуль turtle**» | |
| Выполнили студенты группы АПО-19 | Битанов А.С  Бейсек Б.Ж ...Отеген К.Р Мамлютов В.Р |
| Научный руководитель доцент, PhD | Астапенко Н.В. |
| Петропавловск, 2022 | |

**СОДЕРЖНИЕ**

[**Введение** 3](#_Toc95294841)

[**1.** **Аналитическая часть** 6](#_Toc95294842)

[**1.1** **Постановка задачи** 8](#_Toc95294843)

[**1.2 Характеристика объекта исследования** 28](#_Toc95294844)

[**2.** **Проектная часть** 31](#_Toc95294845)

[**2.1 Описание используемых алгоритмов** 31](#_Toc95294846)

[**Заключение** 33](#_Toc95294847)

[**Cписок используемой литературы** 35](#_Toc95294848)

**Введение**

Программная инженерия (промышленное программирование) обычно ассоциируется с разработкой больших и сложных программ коллективами разработчиков. Становление и развитие этой области деятельности было вызвано рядом проблем, связанных с высокой стоимостью программного обеспечения, сложностью его создания, необходимостью управления и прогнозирования процессов разработки.

Концепция объектно-ориентированного программирования подразумевает, что основой управления процессом реализации программы является передача сообщений объектам. Поэтому объекты должны определяться совместно с сообщениями, на которые они должны реагировать при выполнении программы. В этом состоит главное отличие ООП от процедурного программирования, где отдельно определённые структуры данных передаются в процедуры (функции) в качестве параметров. Таким образом, объектно-ориентированная программа состоит из объектов – отдельных фрагментов кода, обрабатывающего данные, которые взаимодействуют друг с другом через определённые интерфейсы.

Объектно-ориентированный язык программирования должен обладать следующими свойствами:

* **абстракции** – формальное о качествах или свойствах предмета путем мысленного удаления некоторых частностей или материальных объектов;
* **инкапсуляции** – механизма, связывающего вмести код и данные, которыми он манипулирует, и защищающего их от внешних помех и некорректного использования;
* **наследования** – процесса, с помощью которого один объект приобретает свойства другого, т.е. поддерживается иерархической классификации;
* **полиморфизма**– свойства, позволяющего использовать один и тот же интерфейс для общего класса действий.

Разработка объектно-ориентированных программ состоит из следующих последовательных работ:

* определение основных объектов, необходимых для решения данной задачи;
* определение закрытых данных (данных состояния) для выбранных объектов;
* определение второстепенных объектов и их закрытых данных;
* определение иерархической системы классов, представляющих выбранные объекты;
* определение ключевых сообщений, которые должны обрабатывать объекты каждого класса;
* разработка последовательности выражений, которые позволяют решить поставленную задачу;
* разработка методов, обрабатывающих каждое сообщение;
* очистка проекта, то есть устранение всех вспомогательных промежуточных материалов, использовавшихся при проектировании;
* кодирование, отладка, компоновка и тестирование.

Объектно-ориентированное программирование позволяет программисту моделировать объекты определённой предметной области путем программирования их содержания и поведения в пределах класса. Конструкция «класс» обеспечивает механизм инкапсуляции для реализации абстрактных типов данных. Инкапсуляция как бы скрывает и подробности внутренней реализации типов, и внешние операции, и функции, допустимые для выполнения над объектами этого типа.

Суть данного проекта заключается в объединении нескольких человек для получения готового результата. Этот проект мы назвали «МОВВ».

В нем мы познакомились с технологией объектно-ориентированное программирование и способами её реализации с помощью python и познакомились с модулем turtle.

Всю командную работу мы разделили поровну.

* Битанов Асанали взял на себя разработку и написание кода.
* Отеген Казбек – знакомство и разработка GitHub.
* Бейсек Бакустар и Мамлютов Вадим – составление отчёта.

1. **Аналитическая часть**

Программная инженерия — это интегрирование принципов математики, информатики и компьютерных наук с инженерными подходами, разработанными для производства осязаемых материальных артефактов. Программная инженерия определяется как системный подход к анализу, проектированию, оценке, реализации, тестированию, обслуживанию и модернизации программного обеспечения, то есть применение инженерии к разработке программного обеспечения.

Концепция объектно-ориентированного программирования подразумевает, что основой управления процессом реализации программы является передача сообщений объектам.  Поэтому объекты должны определяться   совместно   с сообщениями, на которые они должны реагировать при выполнении программы.  В этом состоит главное отличие объектно-ориентированного программирования от процедурного программирования, где отдельно определённые структуры данных передаются в процедуры (функции) в качестве параметров.  Таким образом, объектно-ориентированная   программа состоит из объектов – отдельных фрагментов кода, обрабатывающего данные, которые взаимодействуют друг с другом через определённые интерфейсы.

Разработка объектно-ориентированных программ   состоит   из:

* определение основных объектов, необходимых для решения данной задачи;
* определение второстепенных объектов и их закрытых данных;
* определение иерархической системы   классов, представляющих выбранные объекты;
* определение ключевых сообщений, которые должны обрабатывать объекты каждого класса;
* разработка последовательности выражений, которые   позволяют решить поставленную задачу;
* разработка методов, обрабатывающих каждое сообщение;
* кодирование, отладка, компоновка и тестирование.

Объектно-ориентированное программирование позволяет   моделировать объекты определённой предметной области путем программирования их содержания и поведения в   пределах   класса. Конструкция «класс» обеспечивает механизм инкапсуляции для реализации абстрактных типов данных. Инкапсуляция как бы скрывает и подробности внутренней реализации типов, и внешние операции, и функции, допустимые для выполнения над объектами этого типа.

* 1. **Постановка задачи**
* Задание: разработать два класса фигур, расположенных в отдельных модулях. Минимальные требования к каждому классу: наличие конструктора, выполняющего отрисовку фигуры (например, изображение смайлика или мухи); метод, выполняющий отрисовку фигуры в другом положении (например, появление улыбки у смайлика или взмах крыльев мухи).
* Разработать внешний вид основного окна, содержащего название проекта, фамилии исполнителей, фон.
* Разработать алгоритм создания одного или нескольких объектов первого класса, движущихся по заранее заданной траектории (например, одна или несколько мух, передвигающихся по синусоиде, меняющие взмах крыльев через определенный интервал).
* Разработать алгоритм создания объекта другого класса, передвижения его по нажатию клавиш, вызов метода при определенных условиях (например, передвижение смайлика при нажатии клавиши и появление улыбки через определенное число перемещений).
* Создать исполняемый файл.

Для решения задачи опишем следующие классы:

Рик (Rick), Морти (Morty), Космический корабль (Space shuttle). Экземпляры данного класса должен уметь обрисовывать себя в области отображения, передвигаться по траектории заданной, в виде окружности, иметь изменять свой радиус и скорость движения, а также направление движения. Для этого он должен знать свое расположение, иметь ссылку на область отображения.

Код отрисовки персонажа Морти (Morty)

**import turtle**

**skin\_color = '#FFDEAD'**

**hair\_color = '#8B4513'**

**class Morty:**

**turtle.title("MOBB, Мамлютов, Отеген, Бейсек, Битанов") # фон, маркер**

**turtle.bgpic("frame1.gif")**

**m = turtle.Turtle()**

**m.shape("turtle")**

**m.pensize(4)**

**m.pencolor('black')**

**m.up() # волосы**

**m.goto(-400, -170)**

**m.setheading(90)**

**m.down()**

**m.color('black', hair\_color)**

**m.begin\_fill()**

**m.circle(250, 180)**

**m.left(90)**

**m.forward(500)**

**m.end\_fill()**

**m.up() # лицо**

**m.home()**

**m.setheading(180)**

**m.forward(650)**

**m.down()**

**m.color('black', skin\_color)**

**m.begin\_fill()**

**m.circle(200, 360)**

**m.end\_fill()**

**m.up() # глаза**

**m.goto(-575, -100)**

**m.down()**

**m.color('black', 'white')**

**m.begin\_fill()**

**m.circle(60, 360)**

**m.up()**

**m.forward(140)**

**m.down()**

**m.circle(60, 360)**

**m.end\_fill()**

**m.up()**

**m.left(90)**

**m.forward(60)**

**m.down()**

**m.dot(10, 'black')**

**m.up()**

**m.left(90)**

**m.forward(140)**

**m.down()**

**m.dot(10, 'black')**

**m.up() # уши**

**m.goto(-455, -250)**

**m.down()**

**m.setheading(0)**

**m.color('black', skin\_color)**

**m.begin\_fill()**

**m.circle(50, 180)**

**m.left(90)**

**m.forward(107)**

**m.end\_fill()**

**m.right(90)**

**m.up()**

**m.goto(-850, -160)**

**m.down()**

**m.color('black', skin\_color)**

**m.begin\_fill()**

**m.circle(50, 195)**

**m.end\_fill()**

**m.up() # брови**

**m.goto(-800, -140)**

**m.down()**

**m.left(35)**

**m.forward(110)**

**m.setheading(0)**

**m.up()**

**m.forward(150)**

**m.down()**

**m.left(315)**

**m.forward(110)**

**m.up () # нос**

**m.goto(-650, -250)**

**m.down()**

**m.forward(15)**

**m.circle(15, 180)**

**m.forward(15)**

**m.up() # рот**

**m.goto(-650, -335)**

**m.down()**

**m.setheading(270)**

**m.left(25)**

**m.forward(15)**

**m.circle(12, 180)**

**m.forward(15)**

**m.setheading(0)**

**m.right(55)**

**m.forward(25)**

**m.circle(14, 180)**

**m.forward(30)**

**Код Рика (Rick)**

**import turtle**

**class Rick:**

**turtle.title("MOBB, Мамлютов, Отеген, Бейсек, Битанов") # фон, маркер**

**turtle.bgpic("frame1.gif")**

**t = turtle.Turtle()**

**t.shape("turtle")**

**t.pensize(4)**

**t.pencolor('black')**

**t.penup() # правый глаз**

**t.forward(75)**

**t.pendown()**

**t.forward(70)**

**t.left(110)**

**t.forward(25)**

**t.circle(30, 125)**

**t.left(22)**

**t.forward(34)**

**t.left(25)**

**t.forward(30)**

**t.circle(30, 150)**

**t.left(13)**

**t.forward(40)**

**t.up() # зрачок**

**t.left(98)**

**t.forward(30)**

**t.down()**

**t.color('black', 'black')**

**t.begin\_fill()**

**t.circle(4, 360)**

**t.end\_fill()**

**t.up() # левый глаз**

**t.forward(160)**

**t.down()**

**t.setheading(0)**

**t.forward(70)**

**t.left(110)**

**t.forward(25)**

**t.circle(30, 125)**

**t.left(22)**

**t.forward(34)**

**t.left(25)**

**t.forward(30)**

**t.circle(30, 150)**

**t.left(13)**

**t.forward(40)**

**t.up() # зрачок**

**t.left(98)**

**t.forward(30)**

**t.down()**

**t.color('black', 'black')**

**t.begin\_fill()**

**t.circle(4, 360)**

**t.end\_fill()**

**t.up() # лицо**

**t.forward(110)**

**t.down()**

**t.left(90)**

**t.forward(180)**

**t.circle(150, 180)**

**t.left(1)**

**t.forward(230)**

**t.circle(150, 150)**

**t.left(22)**

**t.forward(120)**

**t.up() # левое ухо**

**t.forward(75)**

**t.right(90)**

**t.down()**

**t.forward(25)**

**t.circle(25, 180)**

**t.left(340)**

**t.forward(50)**

**t.up() # правое ухо**

**t.setheading(0)**

**t.forward(300)**

**t.down()**

**t.left(25)**

**t.forward(50)**

**t.circle(25, 155)**

**t.forward(45)**

**t.up() # рот**

**t.forward(200)**

**t.left(90)**

**t.forward(100)**

**t.down()**

**t.color('black')**

**t.begin\_fill()**

**t.forward(10)**

**t.left(90)**

**t.forward(150)**

**t.left(90)**

**t.forward(10)**

**t.left(90)**

**t.forward(150)**

**t.end\_fill()**

**t.up()**

**t.right(90)**

**t.forward(20)**

**t.left(90)**

**t.forward(20)**

**t.down()**

**for i in range(2):**

**t.left(65)**

**t.forward(35)**

**t.setheading(0)**

**t.up()**

**t.forward(180)**

**t.down()**

**for i in range(2):**

**t.left(65)**

**t.forward(35)**

**t.setheading(180)**

**t.up()**

**t.forward(40)**

**t.left(90)**

**t.forward(30)**

**t.down()**

**t.color('black', '#07e63f')**

**t.begin\_fill()**

**t.forward(35)**

**t.right(34)**

**t.forward(20)**

**t.left(34)**

**t.forward(25)**

**t.setheading(90)**

**t.left(90)**

**t.forward(10)**

**t.right(45)**

**t.forward(50)**

**t.setheading(90)**

**t.forward(40)**

**t.end\_fill()**

**t.up() # нос**

**t.forward(120)**

**t.left(90)**

**t.forward(15)**

**t.down()**

**t.left(90)**

**t.forward(65)**

**t.circle(20, 180)**

**t.forward(65)**

**t.up() # бровь**

**t.forward(180)**

**t.right(90)**

**t.forward(50)**

**t.down()**

**t.color('black', '#00BFFF')**

**t.begin\_fill()**

**t.setheading(270)**

**t.right(90)**

**t.forward(25)**

**t.left(90)**

**t.forward(5)**

**t.right(90)**

**t.forward(100)**

**t.right(90)**

**t.forward(5)**

**t.left(90)**

**t.forward(55)**

**t.circle(15, 180)**

**t.right(20)**

**t.forward(15)**

**t.setheading(0)**

**t.forward(125)**

**t.left(25)**

**t.forward(45)**

**t.circle(10, 160)**

**t.end\_fill()**

**t.up()**

**t.setheading(0)**

**t.forward(77)**

**t.right(90)**

**t.forward(320)**

**t.setheading(0)**

**t.down()**

**t.left(55)**

**t.forward(175)**

**t.left(90)**

**t.forward(65)**

**t.right(90)**

**t.forward(130)**

**t.left(90)**

**t.forward(100)**

**t.right(90)**

**t.forward(100)**

**t.setheading(180)**

**t.forward(150)**

**t.right(90)**

**t.forward(145)**

**t.left(145)**

**t.forward(140)**

**t.right(90)**

**t.forward(160)**

**t.left(100)**

**t.forward(130)**

**t.setheading(180)**

**t.forward(170)**

**t.left(120)**

**t.forward(150)**

**t.setheading(180)**

**t.forward(120)**

**t.left(150)**

**t.forward(170)**

**t.setheading(180)**

**t.left(40)**

**t.forward(120)**

**t.left(110)**

**Код Космического корабля (Space shuttle)**

**import turtle**

**class Spaceship:**

**turtle.title("MOBB, Мамлютов, Отеген, Бейсек, Битанов") # фон, маркер**

**turtle.bgpic("frame1.gif")**

**shuttle\_roof\_color = '#B0C4DE'**

**shuttle\_main\_color = '#C0C0C0'**

**f = turtle.Turtle()**

**f.shape("turtle")**

**f.pensize(4)**

**f.pencolor('black')**

**def draw(self):**

**Spaceship.f.up() # космический шатл**

**Spaceship.f.goto(800, 180)**

**Spaceship.f.down()**

**Spaceship.f.setheading(90)**

**Spaceship.f.color('black', Spaceship.shuttle\_main\_color)**

**Spaceship.f.begin\_fill()**

**Spaceship.f.circle(60, 90)**

**Spaceship.f.forward(80)**

**Spaceship.f.end\_fill()**

**Spaceship.f.setheading(90)**

**Spaceship.f.color('black', Spaceship.shuttle\_roof\_color)**

**Spaceship.f.begin\_fill()**

**Spaceship.f.circle(100, 180)**

**Spaceship.f.end\_fill()**

**Spaceship.f.color('black', Spaceship.shuttle\_main\_color)**

**Spaceship.f.begin\_fill()**

**Spaceship.f.right(90)**

**Spaceship.f.forward(80)**

**Spaceship.f.circle(60, 90)**

**Spaceship.f.left(90)**

**Spaceship.f.forward(480)**

**Spaceship.f.up()**

**Spaceship.f.goto(800, 180)**

**Spaceship.f.down()**

**Spaceship.f.setheading(90)**

**Spaceship.f.circle(60, 90)**

**Spaceship.f.forward(80)**

**Spaceship.f.end\_fill()**

**Spaceship.f.setheading(90)**

**Spaceship.f.end\_fill()**

**Сбор готового результата происходит в исполнительном файле**

**Сам код сбора**

**from module\_Rick import Rick**

**from module\_Morty import Morty**

**from module\_space\_shuttle import Spaceship**

**import turtle**

**r = Rick()**

**m = Morty()**

**s = Spaceship()**

**turtle.onkey(s.draw, 'Up')**

**turtle.listen()**

**turtle.done()**

**В нём происходит само рисование, также при нажатии на кнопку Up происходит рисование Космического корабля (Space shuttle)**

# **1.2 Характеристика объекта исследования**

Базовым в объектно-ориентированном программировании является понятие объекта. Объект имеет определённые свойства.  Состояние объекта задаётся значениями его признаков. Объект «знает», как решать определённые задачи, то   есть   располагает   методами   решения.   Программа, написанная    с использованием ООП, состоит из объектов, которые могут взаимодействовать между собой.

Программная реализация объекта представляет собой объединение данных и процедур их обработки.  Переменные объектного типа называют экземплярами объекта. Здесь требуется уточнение – экземпляр можно лишь формально назвать переменной. Его описание даётся в предложение описания переменных, но в действительности экземпляр – нечто большее, чем обычная переменная.

В отличие от типа «запись», объектный тип содержит не только поля, описывающие данные, но также процедуры и функции, описания   которых содержится в описании объекта. Эти процедуры и функции называют методами.

Методам объекта доступны его поля.  Следует отметить, что методы и их параметры определяются в описании объекта, а их реализация даётся вне этого описания, в том мест программы, которое предшествует вызову данного метода.

В описании объекта фактически содержаться лишь шаблоны обращения к методам, которые необходимы компилятору для   проверки   соответствия   количества параметров и их типов при обращении к методам.

Основная цель ООП, как и большинства других подходов к программированию - повышение эффективности разработки программ. Идеи ООП оказались плодотворными и нашли применение не только в языках программирования, но и в других областях Computer Science, например, в области разработки операционных систем.

Появление ООП было связано с тем наблюдением, что компьютерные программы представляют собой описание действий, выполняемых над различными объектами. В роли последних могут выступать, например, графические объекты, записи в базах данных или совокупности числовых значений. В традиционных методах программирования изменение данных или правил и методов обработки часто приводило к необходимости значительного изменения программы.

Использование ООП позволяет выйти из такой ситуации с минимальными потерями, сводя необходимую модификацию программы к её расширению и дополнению. Необходимо заметить, что ООП не является панацеей от всех программистских бед, но его ценность как передовой технологии программирования несомненна. Изучение идей и методов ООП может существенно упростить разработку и отладку сложных программ.

Мы уже привыкли использовать в своих программах процедуры и функции для программирования тех сложных действий по обработке данных, которые приходится выполнять многократно.

Подпрограмма может иметь формальные предметы, которые при обращении к ней заменяются фактическими предметами. В этом случае есть опасность вызова подпрограммы с неправильными данными, что может привести к сбою программы и её аварийному завершению при выполнении. Поэтому естественным обобщением традиционного подхода к программированию является объединение данных и подпрограмм (процедур и функций), предназначенных для их обработки.

Модуль Turtle

Turtle (черепашка) – простой графический пакет, который так же

может быть использован для создания несложного пользовательского

графического интерфейса. Модуль turtle входит в стандартную поставку библиотеки Python. Исходно этот модуль позиционируется как средство для обучения компьютерной графике детей. Модель этого модуля следующая. Имеется прямоугольная поверхность, по которой ползает черепашка.

Черепашка может перемещаться на заданное расстояние прямо, назад, под углом или по заданным координатам. Черепашку можно клонировать, создавая группу черепашек. При этом каждая черепашка живет своей жизнью. Для рисования черепашка использует цветовое перо (карандаш), которое может быть поднято или опущено. Если перо опущено, то остается след. Можно изменять цвет и толщину линии.

Черепашка понимает команды, с помощью которых можно нарисовать окружность заданного радиуса и цвета, дугу с заданным углом, залить

фигуру определенным цветом, получить текущее состояние настроек или

изменить их. Форма черепашки может быть изменена пользователем и использована как штамп, после которого на холсте остается рисунок.

В модуле turtle реализованы и интерактивные способы взаимодействия с черепашкой. События, связанные с кликом кнопки мыши или нажатия \ отпускания клавиши клавиатуры, могут обрабатываться пользовательскими функциями, привязанными к этим событиям.

1. **Проектная часть**

**2.1 Описание используемых алгоритмов**

В приложении рисуется несколько рисунков. Далее необходимо нажать на кнопку «Up», при нажатии начинается отрисовка еще одного рисунка. При этом всё вся рисование происходит на готовом изображении для более качественного вывода. Готовый рисунок представлен на Рис. 2 «Рик и Морти»

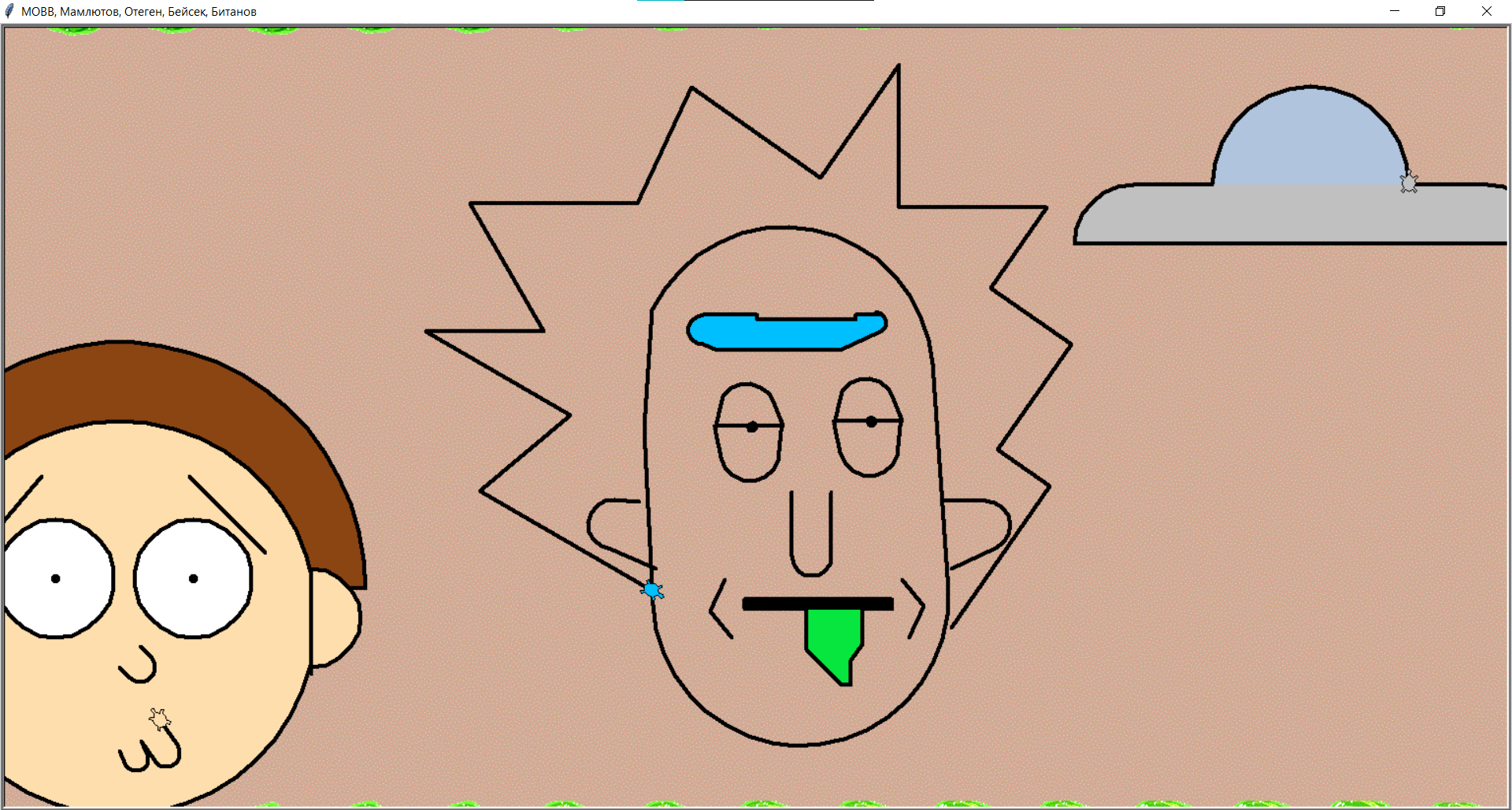
****

Рис. 2 «Рик и Морти»

**Заключение**

В данной работе удалось разработать все поставленные задачи в программе на Python.

Выполнение работы было разделено на пять этапов, как и в самом задании:

* Разработать два класса фигур, расположенных в отдельных модулях. Минимальные требования к каждому классу: наличие конструктора, выполняющего отрисовку фигуры (например, изображение смайлика или мухи); метод, выполняющий отрисовку фигуры в другом положении (например, появление улыбки у смайлика или взмах крыльев мухи).
* Разработать внешний вид основного окна, содержащего название проекта, фамилии исполнителей, фон.
* Разработать алгоритм создания одного или нескольких объектов первого класса, движущихся по заранее заданной траектории (например, одна или несколько мух, передвигающихся по синусоиде, меняющие взмах крыльев через определенный интервал).
* Разработать алгоритм создания объекта другого класса, передвижения его по нажатию клавиш, вызов метода при определенных условиях (например, передвижение смайлика при нажатии клавиши и появление улыбки через определенное число перемещений).
* Создать исполняемый файл.

Итогом работы можно считать закрепление знаний в области создания алгоритмов, навыков программирования на языке Python и модуле turtle, полученных в ходе курса "объектно-ориентированное программирование".

Во время выполнения курсовой работы среда Microsoft Visual Basic работала без неполадок, была проста в обращении, возникающие ошибке при программировании легко исправлялись.

В заключение можно отметить, что модуль turtle является не таким и сложным, если объяснять обычным языком, это как рисование в paint, но только через коды строк.

Таким образом, можно сделать вывод, что первоначальные задачи, поставленные были полностью решены. Практическая часть работы успешно реализована в среде Python

# **Cписок используемой литературы**

1. Хорев, П.Б. Объектно-ориентированное программирование / П.Б. Хорев. - М.: Academia, 2018.
2. Бадд, Т. Объектно-ориентированное программирование в действии / Т. Бадд. - СПб.: Питер, 2017.
3. Бадд, Т. Объектно-ориентированное программирование в действии / Т. Бадд. - СПб.: Питер, 2018.
4. Черпаков, И.В. Основы программирования: Учебник и практикум для прикладного бакалавриата / И.В. Черпаков. - Люберцы: Юрайт, 2016.
5. Юдин, Д.Б. Задачи и методы линейного программирования: Математические основы и практические задачи / Д.Б. Юдин, Е.Г. Гольштейн. - М.: КД Либроком, 2020.
6. Зыков, С. В. Программирование. Объектно-ориентированный подход: учебник и практикум для академического бакалавриата / С. В. Зыков. — М.: Издательство Юрайт, 2019.
7. Федоров, Д. Ю. Программирование на языке высокого уровня python: учеб. пособие для прикладного бакалавриата / Д. Ю. Федоров. — 2-е изд., перераб, и доп. — М.: Издательство Юрайт, 2019.
8. Мэтиз Э. Изучаем Python. Программирование игр, визуализация данных, веб-приложения. — СПб.: Питер, 2017.
9. Федоров Д.Ю. Основы программирования на примере языка Python: учебное пособие. – СПб.: Питер, 2019.
10. Рядченко, В.П. Программирование на языке высокого уровня Python: учебно-методическое пособие / В.П. Рядченко, Л.М. Эльканова, Л.М. Шавтикова. – Черкесск: БИЦ СевКавГГТА, 2018.