**ФГБОУ ВО Уральский государственный горный университет**

**Инженерно-экономический факультет**

**Кафедра информатики**

Курсовой проект

По дисциплине «Программирование»

На тему «Space Invaders»»

**Выполнил:**  Студент гр. АУБП-18 Усольцев Г.К

**Проверила:**

ст. преп. каф. Информатики Волкова Е.А.

**Екатеринбург, 2020г.**

41

«Space invaders»

**Формулировка задания**

Реализовать классическую игру « Space invaders »

**Постановка задачи**

Реализовать алгоритм игры « Space invaders »

**Содержание**

1.Постановка задачи ...........................................................................................2

1.1. Формулировка задания .............................................................................. 4

1.2. Алгоритмическое разрешение задачи ...................................................... 5

1.3. Контрольные примеры ............................................................................... 6

2.Решение задачи .............................................................................................. 8   
2.1 Выбор средств реализации ......................................................................... 8

2.2. Описание основных классов ...................................................................... 9

2.3. Интерфейс приложения ..............................................................................9

3. Тестирование приложения ............................................................................9

3.1. Работа приложения на контрольных тестах .............................................11

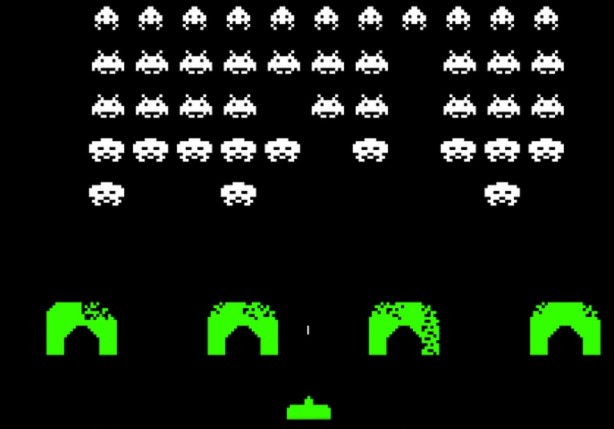
3.2. Результаты работы .......................................................................................9

1.Постановка задачи

**1.1. Формулировка задания**

Написать код реализующий игру « Space invaders »

Пример программной области:



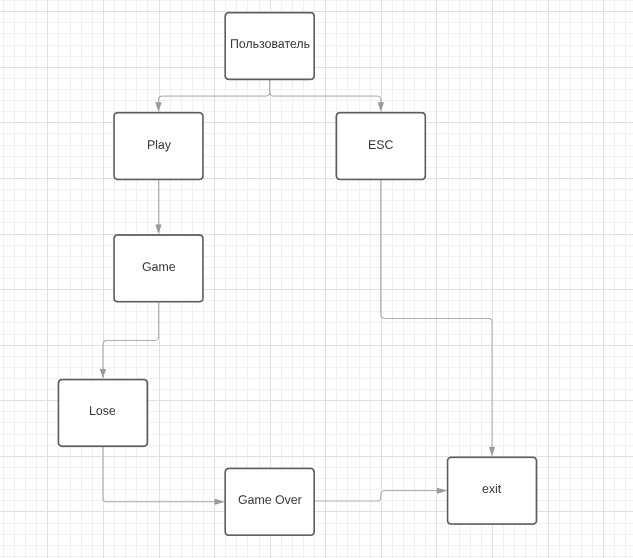
**Постановка задачи**

Реализовать алгоритм игры « Space invaders »

**1.2. Алгоритмическое разрешение задачи**

Алгоритм проверяет какие клавиши нажимает пользователь , на основе этих данных исполняет движение и выстрелы

Алгоритм пользователя, чтобы играть в игру



**1.3. Контрольные примеры**

**Пример №1**  
Выбираем режим “Play”

Нажимаем “space” для выстрела

Нажимаем “RiGTH\_K” чтобы доска передвинулась направо

Нажимаем “LEFT\_K” чтобы доска передвинулась налево

Изменяется кол-во очков

Изменяется уровень

Корабль имеет определенное кол-во очков

**Пример №2**

Регистрация попадание по нескольким enemy

Проверка работы, проггессии в очках за одного enemy

Проверка правильности работы жизней в игре

2.Решение задачи   
2.1. Выбор средств реализации

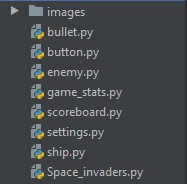
В качестве языка программирования для решения этой задачи был выбран язык Python.

Python это язык программирования общего назначения, нацеленный в первую очередь наповышение продуктивности самого программиста, нежели кода, который он пишет. Говоря простым человеческим языком, на Python можно написать практически что угодно (веб-/настольные приложения, игры, скрипты по автоматизации, комплексные системы расчёта, системы управления жизнеобеспечением и многое многое другое) без ощутимых проблем.

Также я ипользовал библиотеку pygame, для более удобной реализации данного проекта

**2.2. Описание основных классов**

При решении задачи было разработано 8 классов, показанных ниже:



Главный цикл реализующий все действие в игре под названием “Space\_indavers”

Разберем его по подробнее:

1. Добавление основных библиотек и импортирование различных классов

import sys  
from time import sleep  
  
import pygame  
  
from settings import Settings  
from game\_stats import GameStats  
from scoreboard import Scoreboard  
from button import Button  
from ship import Ship  
from bullet import Bullet  
from enemy import Enemy

1. Класс показывающий все объекты в игре  
   class SpaceIndavers:  
     
    def \_\_init\_\_(self):  
     
    pygame.init()  
    self.screen = pygame.display.set\_mode([800, 600])  
    pygame.display.set\_caption("SpaceIndavers")  
     
    self.settings = Settings()  
    self.stats = GameStats(self)  
    self.sb = Scoreboard(self)  
    self.ship = Ship(self)  
    self.bullets = pygame.sprite.Group()  
    self.enemy = pygame.sprite.Group()  
     
    self.\_create\_fleet()  
    # create button play  
    self.play\_button = Button(self, "Play")
2. Функция, котороя создает и выщитывает мах короблей в рядах

def \_create\_fleet(self):  
 # create enemy  
 enemy = Enemy(self)  
 enemy\_width, enemy\_height = enemy.rect.size  
 enemy\_width = enemy.rect.width  
 aviilable\_space\_x = self.settings.screen\_width - (2 \* enemy\_width)  
 number\_enemy\_x = aviilable\_space\_x // (2 \* enemy\_width)  
  
 ship\_heignt = self.ship.rect.height  
 aviilable\_space\_y = (self.settings.screen\_height -  
 (3 \* enemy\_height) - 2 \* ship\_heignt)  
 number\_rows = aviilable\_space\_y // (2 \* enemy\_height)  
  
 # create first enemy fleet  
 for row\_number in range(number\_rows):  
 for enemy\_number in range(number\_enemy\_x):  
 self.\_create\_enemy(enemy\_number, row\_number)  
  
 def \_create\_enemy(self, enemy\_number, row\_number):  
 # create enemy  
 enemy = Enemy(self)  
 enemy\_width, enemy\_height = enemy.rect.size  
 enemy.x = enemy\_width + 2 \* enemy\_width \* enemy\_number  
 enemy.rect.x = enemy.x  
 enemy.rect.y = enemy.rect.height + 2 \* enemy.rect.height \* row\_number  
 self.enemy.add(enemy)

1. Функция, которая определяет края экрана и заставляет enemy двигаться в другом направлений  
     
    def \_change\_fleet\_edges(self):  
    for enemy in self.enemy.sprites():  
    if enemy.check\_edges():  
    self.\_change\_fleet\_direction()  
    break  
     
    def \_change\_fleet\_direction(self):  
    for enemy in self.enemy.sprites():  
    enemy.rect.y += self.settings.fleet\_drop\_speed  
    self.settings.fleet\_direction \*= -1  
     
    self.screen = pygame.display.set\_mode(  
    (self.settings.screen\_width, self.settings.screen\_height))
2. Основная функция запускающая игру  
     
    def run\_game(self):  
    # start main round game  
    while True:  
    self.\_update\_enemy()  
    self.\_check\_events()  
    if self.stats.game\_active:  
    self.ship.update()  
    self.\_update\_bullets()  
     
    self.\_update\_screen()
3. Удаление пуль вышедших за края карты, и проверка дошла ли пуля до цели, также добаление level и обнавление enemy  
     
    def \_update\_bullets(self):  
    self.bullets.update()  
    # delete bullet  
    for bullet in self.bullets.copy():  
    if bullet.rect.bottom <= 0:  
    self.bullets.remove(bullet)  
    print(len(self.bullets))  
    # delete enemy  
    self.\_check\_bullet\_enemy\_collisions()  
     
    def \_check\_bullet\_enemy\_collisions(self):  
    # delete ,bullet and enemy  
    collisions = pygame.sprite.groupcollide(  
    self.bullets, self.enemy, True, True)  
    if collisions:  
    for enemy in collisions.values():  
    self.stats.score += self.settings.enemy\_points \* len(enemy)  
    self.sb.prep\_score()  
    if not self.enemy:  
    # create new fleet  
    self.bullets.empty()  
    self.\_create\_fleet()  
    self.settings.increase\_speed()  
     
    # + level  
    self.stats.level += 1  
    self.sb.prep\_level()  
     
    def \_update\_enemy(self):  
    self.\_change\_fleet\_edges()  
    self.enemy.update()  
    if pygame.sprite.spritecollideany(self.ship, self.enemy):  
    self.\_ship\_hit()  
    self.\_check\_enemy\_bottom()  
     
    def \_ship\_hit(self):  
    if self.stats.ships\_left > 0:  
    self.stats.ships\_left -= 1  
    self.sb.prep\_ships()  
     
    self.enemy.empty()  
    self.bullets.empty()  
     
    self.\_create\_fleet()  
    self.ship.center\_ship()  
     
    # pause  
    sleep(0.5)  
    else:  
    self.stats.game\_active = False  
    pygame.mouse.set\_visible(True)  
     
    def \_check\_enemy\_bottom(self):  
    screen\_rect = self.screen.get\_rect()  
    for enemy in self.enemy.sprites():  
    if enemy.rect.bottom >= screen\_rect.bottom:  
      
    self.\_ship\_hit()  
    break
4. действие с клавиатурой  
     
    def \_check\_events(self):  
    # history screen and keyboard  
     
    for event in pygame.event.get():  
    if event.type == pygame.QUIT:  
    sys.exit()  
    elif event.type == pygame.MOUSEBUTTONDOWN:  
    mouse\_pos = pygame.mouse.get\_pos()  
    self.\_check\_play\_button(mouse\_pos)  
    elif event.type == pygame.KEYDOWN:  
    self.\_check\_keydown\_events(event)  
    elif event.type == pygame.KEYUP:  
    self.\_check\_keyup\_events(event)  
     
    def \_check\_play\_button(self, mouse\_pos):  
     
    if self.play\_button.rect.collidepoint(mouse\_pos):  
    # delete static  
    button\_clicked = self.play\_button.rect.collidepoint(mouse\_pos)  
    if button\_clicked and not self.stats.game\_active:  
    self.settings.initialize\_dynamic\_settings()  
    self.stats.reset\_stats()  
    self.stats.game\_active = True  
    self.sb.prep\_level()  
    self.sb.prep\_score()  
     
    pygame.mouse.set\_visible(False)  
     
    self.enemy.empty()  
    self.bullets.empty()  
     
    self.\_create\_fleet()  
    self.ship.center\_ship()  
     
    def \_check\_keydown\_events(self, event):  
    # reaction on keystroke button  
    if event.key == pygame.K\_RIGHT:  
    self.ship.moving\_right = True  
    elif event.key == pygame.K\_LEFT:  
    self.ship.moving\_left = True  
    # exit with button esc  
    elif event.key == pygame.K\_ESCAPE:  
    sys.exit()  
    elif event.key == pygame.K\_SPACE:  
    self.\_fire\_bullet()  
     
    def \_check\_keyup\_events(self, event):  
    # reaction on don't keystroke button  
    if event.key == pygame.K\_RIGHT:  
    self.ship.moving\_right = False  
    elif event.key == pygame.K\_LEFT:  
    self.ship.moving\_left = False
5. Создание пули  
     
    def \_fire\_bullet(self):  
    # create new bullet  
    new\_bullet = Bullet(self)  
    self.bullets.add(new\_bullet)
6. Постоянное обновление экрана и объектов  
     
    def \_update\_screen(self):  
     
    self.screen.fill(self.settings.bg\_color)  
    self.ship.blitme()  
    for bullet in self.bullets.sprites():  
    bullet.draw\_bullet()  
    self.enemy.draw(self.screen)  
    self.sb.show\_score()  
    if not self.stats.game\_active:  
    self.play\_button.draw\_button()  
     
    pygame.display.flip()
7. Запуск игры  
     
     
   if \_\_name\_\_ == '\_\_main\_\_':  
    # create and start game  
    ai = SpaceIndavers()  
    ai.run\_game()

Также можно было бы рассмотреть и другие классы

**2.3. Интерфейс приложения**

Кнопка “play” - начало игры

Кнопка “space” – выстрел

Кнопки “Right, Left” – движение

Кнопка “ESC” - выход

Также на экране есть ряд показателей:

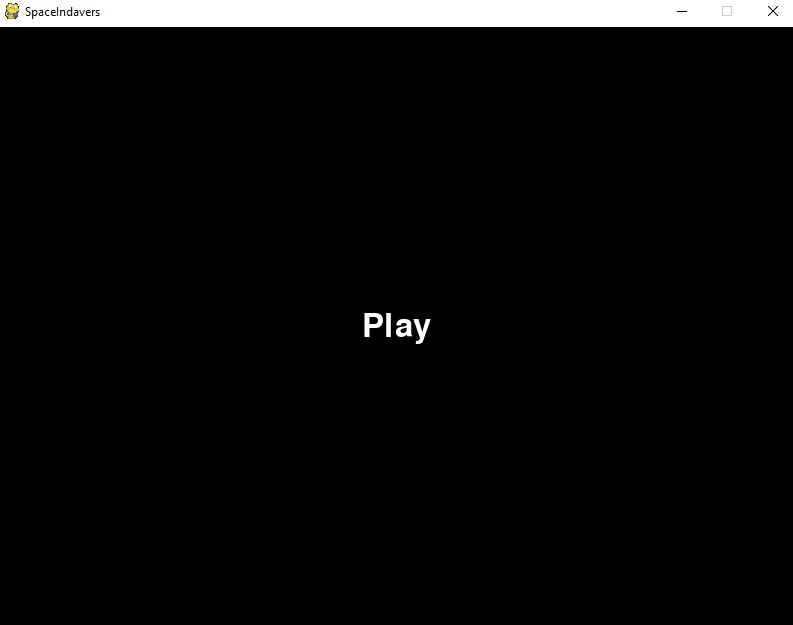
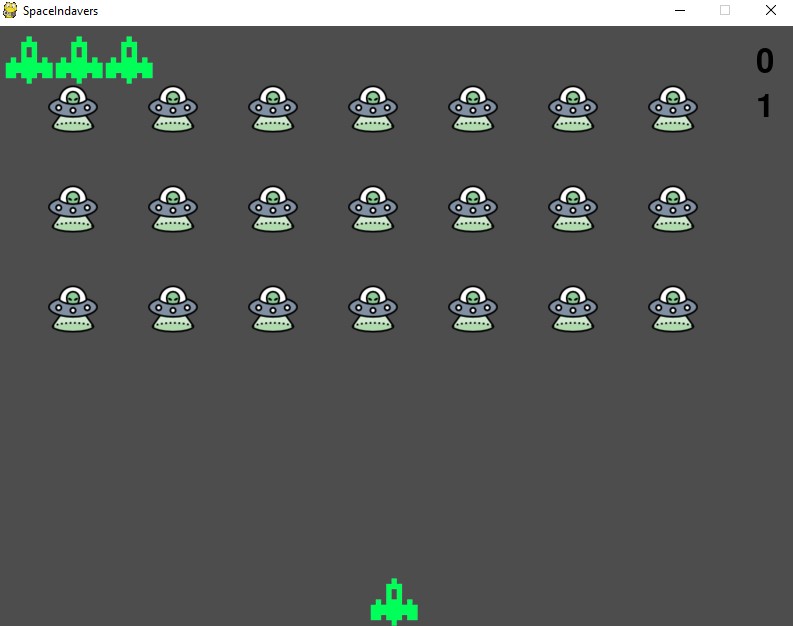
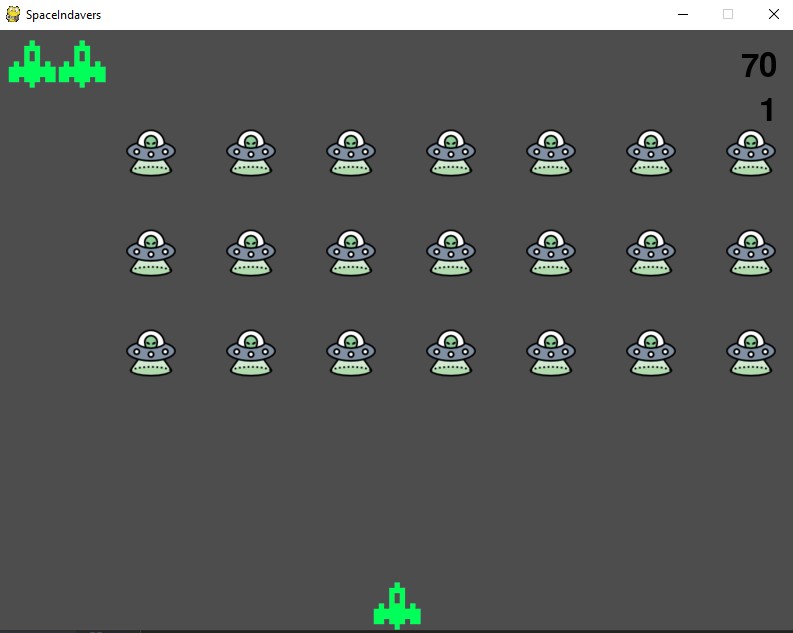
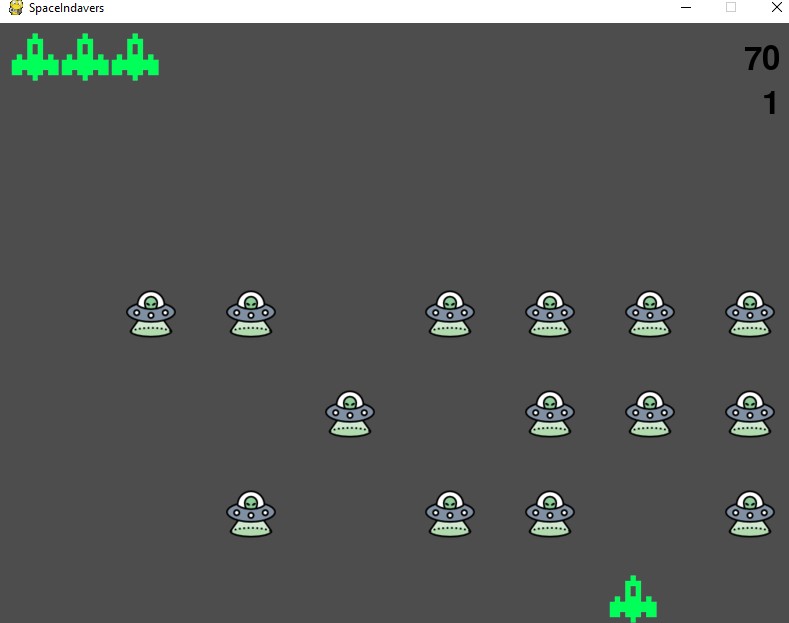
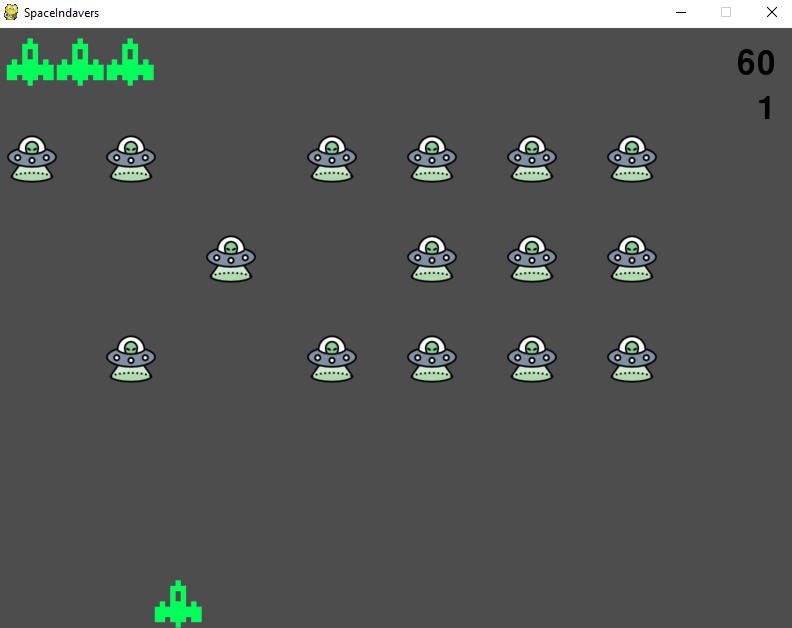
Кол-во жизней

Уровни

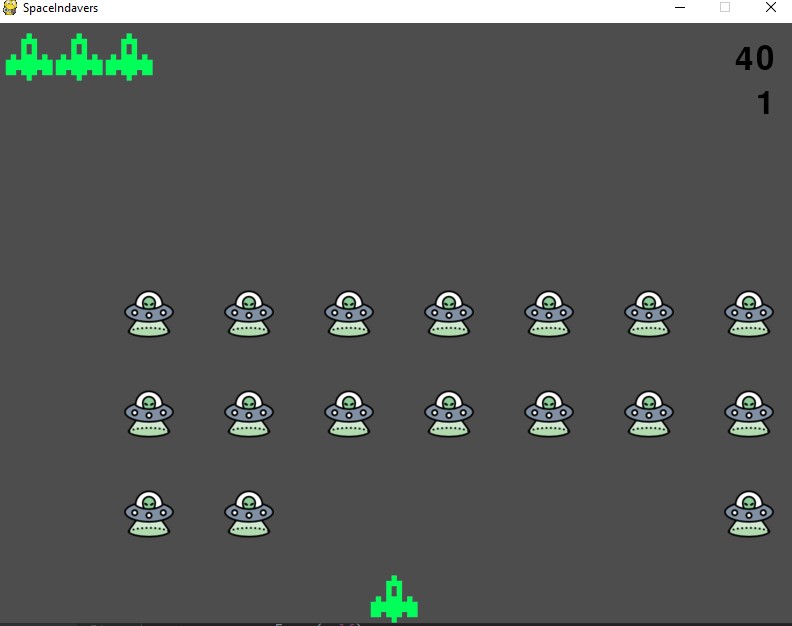
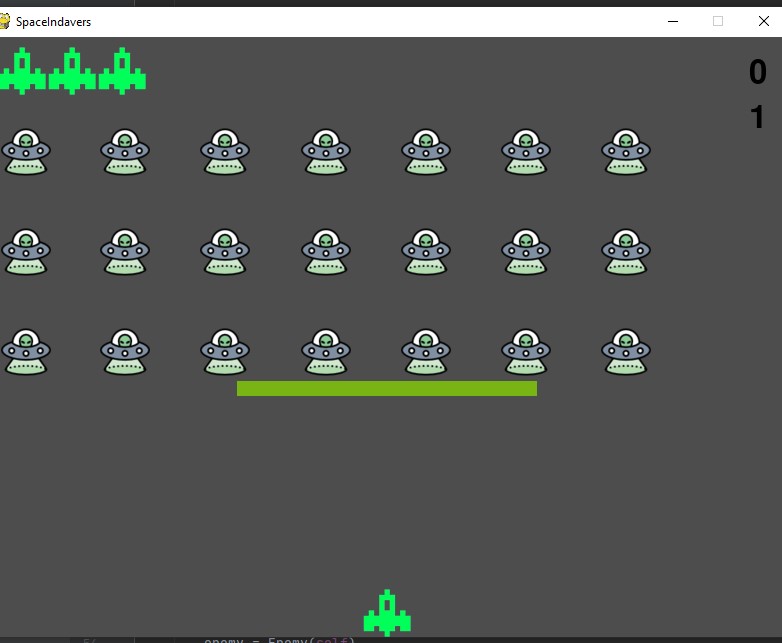
Кол-во очков

**3. Тестирование приложения**

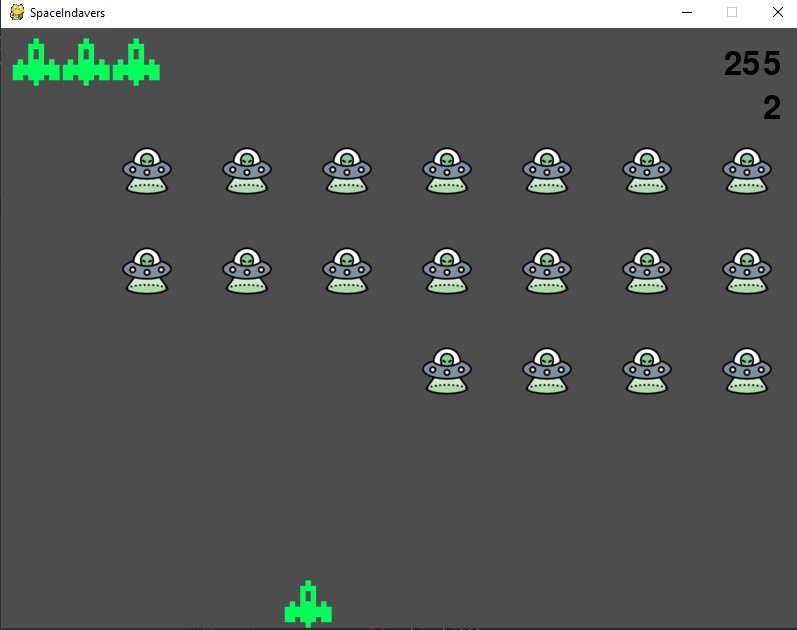
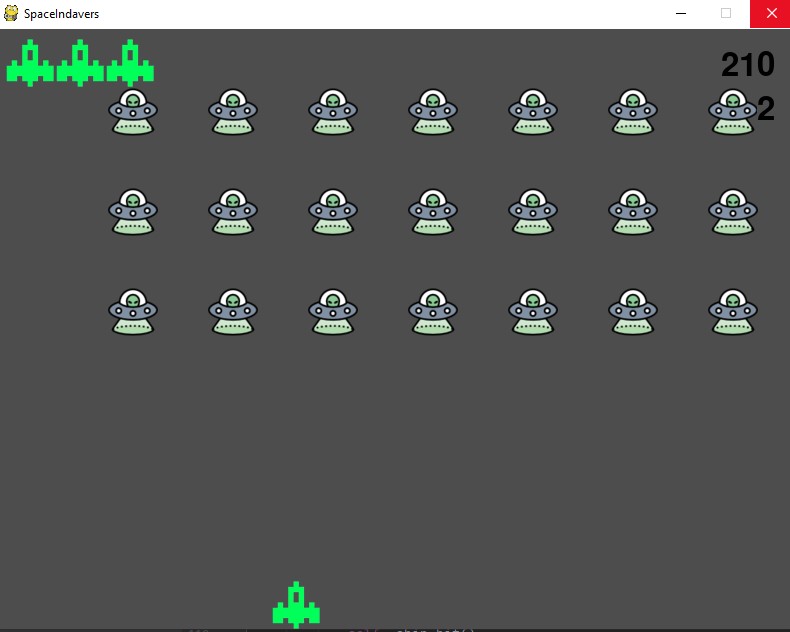
**3.1. Работа приложения от начало и до конца**  
**Полный цикл работы игры**

1. Начало игры Экран “Play”.
2. Пользователь начинает играть. 
3. Когда ship унижтовает enemy ему начисляют очки..
4. Если enemy уничтожают ship, то кол-во жизней уменьшается, если жизни не остается игра заканчивается. 
5. Есть прогрессия в уровнях(больше очков и скорость enemy).
6. И да корабль перемещатся влево и вправо.

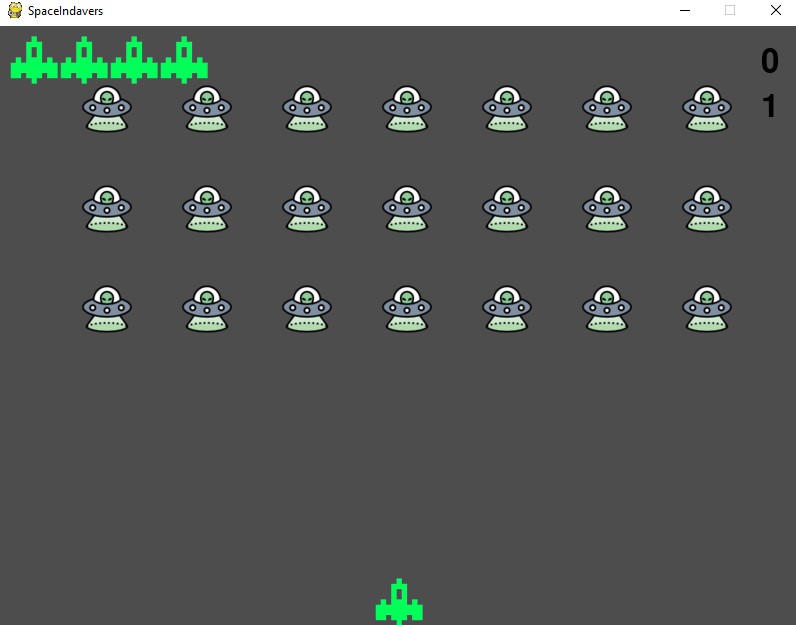
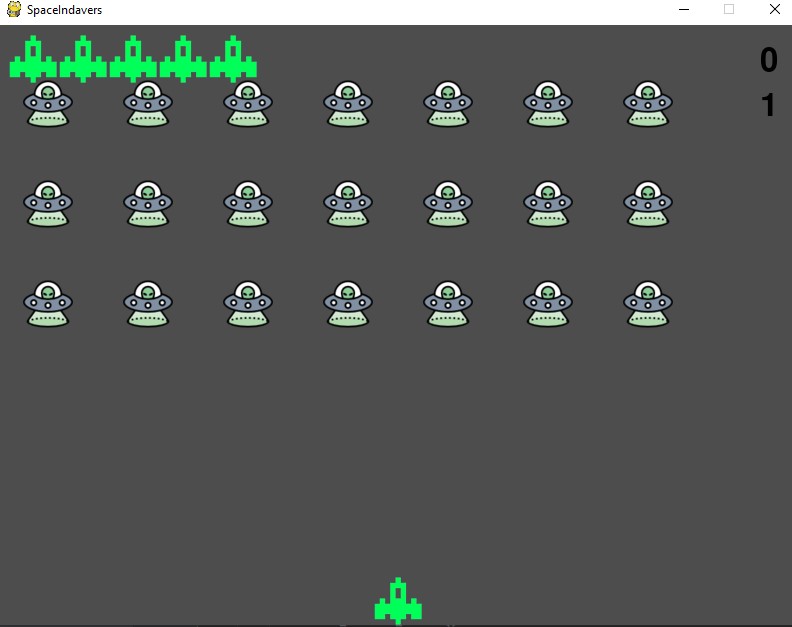
**3.2. Работа приложения на контрольных тестах**

* 1. Тест: регистрация попадания по нескольким enemy. Для этого увеличим наш снаряд и выстрелим в несколько enemy сразу. self.bullet\_width = 300

Вывод: все попадание регестрируются

* 1. Тест: проверим увеличиваются кол-во очков за одного enemy каждый след.уровень. Начальное кол-во очков 10 постовим прогрессию в 1.5 раза. self.score\_scale = 1.5

Вывод: Как мы видим число некруглое, значит прогрессия очков работает верно.

* 1. Тест: проверим правильность работы жизней в игре: для этого увеличим кол-во жизней (до 5) self.ship\_limit = 5

Вывод: Жизни в данном приложении работают корректно.

**3.3. Результаты работы приложения**   
Данная работа предоставила возможность научиться разрабатывать приложения на выбранном мной языке программирования.  
  
Данное приложение позволяет поиграть в игру Space Invaders.