МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский Авиационный Институт»
(Национальный Исследовательский Университет)

Институт: №8 «Информационные технологии и прикладная
математика»
Кафедра: 806 «Вычислительная математика и программирование»

Курсовая работа
по курсу «Фундаментальная
информатика» I семестр
Задание 4
«Процедуры и функции в качестве параметров»

| | |
|---|---|
| Группа | М8О-109Б-22 |
| Студент | Дударь Ю.М. |
| Преподаватель | Сысоев М.А. |
| Оценка | |
| Дата | |

Москва, 2022

**Содержание**



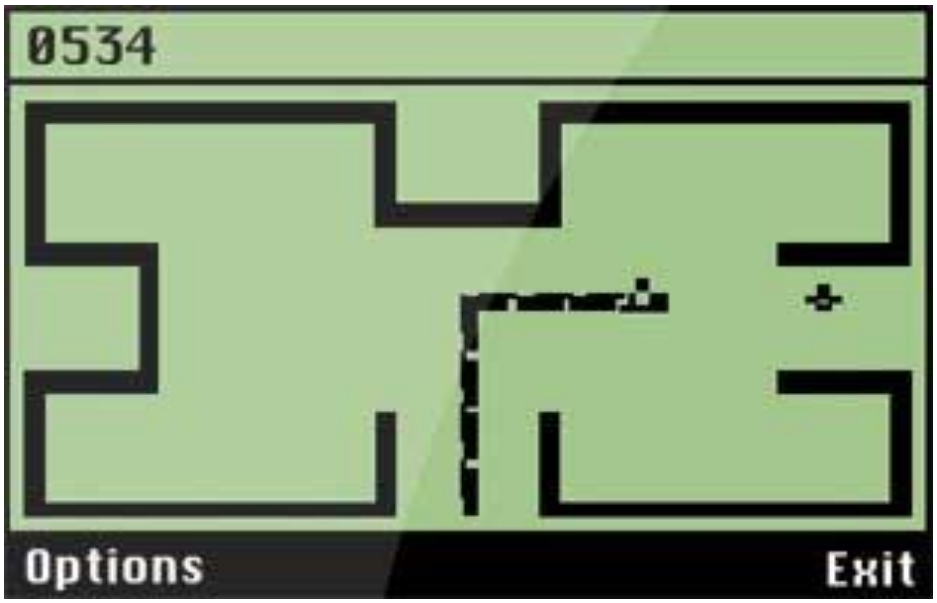
**План выполнения проекта**

1. Выбор идеи проекта
2. Поиск информации о змейке
3. Выбор движка для написания игры
4. Поиск моделек для игры
5. Изучение документации
6. Выполнение проекта
7. Тестирование итоговой версии игры и написание отчета

**История появления оригинальной змейки**

История игры «Змейка» началась за несколько лет до появления первых мобильных телефонов. В 1977 году компания Gremlin Industries выпустила игровой автомат Hustle, рассчитанный на одного или двух игроков, в которой нужно было управлять «змейками», направляя их на бессистемно появляющиеся цели. Для победы нужно было заполучить больше очков, чем у оппонента, преграждая по ходу игры ему путь к новым целям (в случае многопользовательской игры), или просто побить установленный на игровом автомате рекорд. В 1984 году Gremlin Industries была вынуждена закрыться, но игра Hustle начала набирать обороты: сначала появился порт для компьютеров TRS-80, затем для Commodore PET и Apple

Оригинальная «Змейка» (Snake) от Nokia появилась в 1997 году благодаря стараниями разработчика Танели Орманто. В том же году компания выпустила первый телефон с этой игрой — Nokia 6110. Уже тогда игра была многопользовательской: телефоны общались через ИК-порты, ведь ни




Bluetooth, ни тем более Wi-Fi в телефонах в то время не было. Сама змейка

состояла из чёрных квадратов и могла двигаться в четырёх направлениях. Игровая зона, по которой передвигалось пресмыкающееся, была ограничена размерами экрана телефона: при ударе головы змейки о край телефона игра завершалась. «Змейка» приобрела невероятную популярность, сравнимую разве что с популярностью современных хитов «Angry Birds» и «Cut the Rope».

Вторая часть «Змейки» — "Snake II" — обзавелась «бесшовной» игровой зоной, так что змейка не «умирала», врезаясь в край экрана, а выползала из противоположной части дисплея. Также были добавлены карты с препятствиями, врезаться в которые было нельзя. В новой версии был и чит: если успевать ставить игру на паузу в момент поглощения пищи змейкой, она не увеличивалась в размерах. С помощью этой уловки усидчивые, но не очень честные геймеры зарабатывали умопомрачительный счёт, чтобы потом похвастаться перед друзьями. В итоге, не вполне честным путём люди зарабатывались и 20, и 30 тысяч очков — это при том, что в первой «Змейке» больше 4500 баллов набрать было невозможно.

Впоследствии Nokia выпустила игру Snake Xenzia для монохромных и бюджетных цветных телефонов, Snake EX и Snake EX2 с улучшенной


графикой и мультиплеером через Bluetooth для цветных телефонов на S60 и S40.

В январе 2005 года специально для N-Gage компанией IOMO (являющейся крупнейшим разработчиком мобильных игр в Европе на то время) по заказу Nokia была разработана новая, шестая, версия игры — «Змейки», или же "Snakes". На этот раз игра была уже трёхмерной, а графика соответствовала уровню Sony PlayStation. В «Snakes» присутствовал мультиплеер через Bluetooth на 4-х человек, была реализована «вирусная» быстрая передача самой игры на N-Gage друзей. В последствии игра «Snakes» стала доступна для всех смартфонов на S60, правда, уже без мультиплеера.

Следующей версией игры стала трёхмерная "Snake III". Она повторяла игру «Snakes» и отличалась лишь реалистичностью графики — всё-таки в «Snakes» змеи были довольно абстрактные. В остальном же игра содержала те же режимы игры и тот же мультиплеер через Bluetooth.

Наконец, в 2008 году вышла последняя версия «Змейки» — "Snakes Subsonic". «Сабсоник» являлся продолжением игры «Snakes», но уже для игрового сервиса «N-Gage», доступного на двух десятках устройств Nokia.

После выхода «Snakes Subsonic» легендарная «Змейка» уже не развивалась — эра кнопочных телефонов прошла, а для сенсорных мобильных устройств игра не вполне подходила. Тем не менее в «Змейку» продолжают играть и сейчас.

Что случилось с создателями Змейки
Создателю «Змейки» на Nokia Танели Армато сейчас 57 лет, он продолжает работать программистом. Но мобильное подразделение Nokia, проданное Microsoft, а позже — китайцам, пришлось сменить на небольшую компанию Into oy, которая специализируется на информационных технологиях и работе с данными.

Изначально в Nokia Армато занимался разработкой интерфейсов и рингтонов — игры не были его специализацией. Поэтому неудивительно, что сейчас он занят другой, более интересующей его областью информационных технологий. Тем не менее, в 2015 году финский разработчик объединился с компанией Rumilus Design и представил миру обновленную версию культовой игры под названием Snake Rewind для iOS, Android и Windows Phone. В ней больше уровней, современная графика и звуки – убедитесь сами!

Что до программистов из компании Gremlin, которые придумали саму идею «Змейки», то о них известно мало. Есть информация лишь о том, что в 1978 году компанию купила Sega, и спустя пять лет решила закрыть ее — к тому времени начиналась эра персональных компьютеров, и аркадные автоматы начали уходить в прошлое.

**Информация о выбранном движке**

Ursina Engine — это полноценный движок под Windows, Linux и Mac, написанный на Panda3D, Pillow и Pyperclip. Его можно использовать для создания 2D- и 3D-игр. В комплекте библиотеки — готовые шейдеры, геометрические примитивы и анимации.

**Реализация собственной змейки**

Описание кода
Для упрощения разработки игры, раздробим дерево проекта на два файла, файла с основной логикой игры, такой как отрисовка карты, считывание поедания яблок, критерии “геймовера”, начало новой игры и остальными механиками. Во втором файле будет отрисовка змейки и высчитывание появления яблок.

Основным игровым пространством будет поле 20 на 20 клеток, для красоты под это поле добавим скайбокс и установим правильную изометрию камеры для создания эффекта 3D. Причем я реализовал механику смены вида камеры на вид сверху, что приводит к созданию эффекта 2D игры, что выгодно отличает мою змейку от уже написанных.
По умолчанию будем открывать игру в полноэкранном режиме, добавим технологию vsync для синхронизации fps и частоты обновления экрана.

Сверху рабочего пространства выведем информацию о набранном количестве очков, изначально оно будет нулевое, но при поедании яблок, будут начисляться баллы по одному за каждый съеденный фрукт.

При поедании яблок тело змейки должно расти, считывать попадание головы змейки, переднего элемента, на яблоко будем по координатам.

Если змейка выходит за пределы игрового поля или наезжает на сому себя, выводим надпись “Game over” как дань памяти оригинальной змейке.

Движение змейки будем осуществлять следующем образом, создаем два списка, первый Snake_head – для хранения координат перемещения головы змейки, а во втором snake_body будем хранить сегменты змейки. Объявим переменную для длины змейки snake_length. Причем голова змейки – snake_head[-1], snake_head(snake_head+direction) – обработка следующего шага змейки, а для выявления координат сегментов делаем срез, snake_head = snake_head[-snake_length:] и перезаписываем координаты всех сегментов на новые. Изначально голова змейки и яблоко будут располагаться на карте рандомно.

Для управления змейкой создадим три словаря с векторами приращения движения для головы змейки при нажатии клавиш wasd, словари на разрешение использование клавиш и противоположными по движению клавишами для исключения прохождения змейки через сому себя.

Для фиксирования наезжания змейки для себя, используем логику
Lenn_snake = self.snake.snake_head
len(Lenn_snake) == len(set(Lenn_snake)), ведь если змейка наедет на себя, то появятся дублирующие координаты и это равенство будет нарушено.

Код змейки

pysnake.py

```
from ursina import *
from snake_apple import *


class Game(Ursina):
    def __init__(self):
        super().__init__()
        window.fullscreen = True
        window.vsync = True
        self.size_of_map = 20
        camera.position = (self.size_of_map // 2, -21, -24)
        camera.rotation_x = -52
        self.start_new_game()

    def start_new_game(self):
        scene.clear()
        self.start_map(self.size_of_map)
        self.apple = Apple(self.size_of_map, model='sphere', rotation=(-90,
90, 0), texture='apple.jpg')
        self.snake = Snake(self.size_of_map)

    def start_map(self, size_of_map):
        Entity(model='quad', scale=size_of_map, position=(size_of_map // 2,
size_of_map // 2, 0),
               color=color.white66, texture='white_cube', texture_scale=(20,
20))
        Entity(model='quad', scale=size_of_map, position=(size_of_map // 2,
size_of_map // 2, 0),
               texture_scale=(20, 20), texture="трава_10.jpg")
        Entity(model='quad', scale=size_of_map * 3, position=(size_of_map //
2, size_of_map // 2, 1),
               color=color.white66, texture='closeupcloudysky.mp4')
        Entity(model=Grid(size_of_map, size_of_map), scale=size_of_map, posi-
tion=(size_of_map // 2, size_of_map // 2, -0.01), color=color.black)

    def apple_eaten(self):
        if self.snake.snake_head[-1] == self.apple.position:
            self.snake.add_segment()
            self.apple.new_position()

    def game_over(self):
        lens_snake = self.snake.snake_head
        if 0 < lens_snake[-1][0] < self.size_of_map and 0 < lens_snake[-1][1]
< self.size_of_map and len(lens_snake) == len(set(lens_snake)):
            return
        print_on_screen('GAME OVER', position=(-0.7, 0.1), scale=10, dura-
tion=1)
        self.snake.direction = Vec3(0, 0, 0)
        self.snake.permissions = dict.fromkeys(self.snake.permissions, 0)
        invoke(self.start_new_game, delay=1)

    def input(self, key, is_raw=False):
        super().input(key)
        self.snake.control(key)
        if key == '2':
            camera.rotation_x = 0
            camera.position = (self.size_of_map // 2, self.size_of_map // 2, -
50)
        elif key == '3':
            camera.position = (self.size_of_map // 2, -21, -24)
            camera.rotation_x = -52

    def update(self):
```

```
        print_on_screen(f'Очки: {self.snake.score}', position=(-0.05, 0.45),
scale=1, duration = 0.05)
        self.apple_eaten()
        self.game_over()
        self.snake.run()

if __name__ == '__main__':
    game = Game()
    update = game.update
    game.run()
```

snake_apple.py

```
from ursina import *
from random import randrange


class Apple(Entity):
    def __init__(self, size_of_map, **kwargs):
        super().__init__(**kwargs)
        self.size_of_map = size_of_map
        self.new_position()

    def new_position(self):
        self.position = (randrange(self.size_of_map) + 0.5,
randrange(self.size_of_map) + 0.5, -0.5)


class Snake:
    def __init__(self, size_of_map):
        self.MAP_SIZE = size_of_map
        self.snake_length = 1
        self.snake_head = [Vec3(randrange(size_of_map) + 0.5,
randrange(size_of_map) + 0.5, -0.5)]
        self.snake_body = []
        self.create_segment(self.snake_head[0])
        self.directions = {'a': Vec3(-1, 0, 0), 'd': Vec3(1, 0, 0), 'w':
Vec3(0, 1, 0), 's': Vec3(0, -1, 0)}
        self.direction = Vec3(0, 0, 0)
        self.permissions = {'a': 1, 'd': 1, 'w': 1, 's': 1}
        self.block_movement = {'a': 'd', 'd': 'a', 'w': 's', 's': 'w'}
        self.speed, self.score = 20, 0
        self.frame_counter = 0

    def add_segment(self):
        self.snake_length += 1
        self.score += 1
        self.create_segment(self.snake_head[0])

    def create_segment(self, position):
        entity = Entity(position=position)
        Entity(model='cube', texture='krasivo-8.jpg', position=posi-
tion).add_script(
            SmoothFollow(speed=20, target=entity))
        self.snake_body.insert(0, entity)

    def run(self):
        self.frame_counter += 1
        if not(self.frame_counter % self.speed):
            self.snake_head.append(self.snake_head[-1] + self.direction)
            self.snake_head = self.snake_head[-self.snake_length:]
            for segment, segment_position in zip(self.snake_body,
self.snake_head):
                segment.position = segment_position

    def control(self, key):
        for pressed_key in 'wasd':
```

```
            if pressed_key == key and self.permissions[pressed_key]:
                self.direction = self.directions[pressed_key]
                self.permissions = dict.fromkeys(self.permissions, 1)
                self.permissions[self.block_movement[pressed_key]] = 0
                break
```

Проблемы, с которыми я столкнулся

Во время выполнения проекта я столкнулся с множеством проблем, их всех я преодолел в особенности StackOverflow. Благодаря этому я хорошо подтянул уровень своего английского языка, так как большинство информации об используемом движке на иностранном языке. Множество багов было связано с неумением использовать ооп. Некоторые проблемы возникали из-за неправильной работы со словарями, я неправильно прописывал ограничения на управления змейкой.

**Заключение**

Мне очень понравилась идея введения подобного реферата, ведь это дает свободу полета и обширный выбор технологий для реализации идеи. Я прокачал навыки программирования на всеми любимом языке python. Поностальгировал про времена сдачи егэ. Весело провел время над устранением возникающих багов и продумыванием алгоритма змейки и механик.
В итоге я получил готовую игру, которую с гордостью могу выставлять на торговые площадки и стоить на ней крупный бизнес. Благодарю за изменение стандартного варианта реферата своего лабника Максима. С наступающим новым годом!
Очень жаль, что не смогу подготовить красивое и обширное выступление с презентацией змейки, желаю Максиму побольше свободного времени в новом году.

Задание 100/10