Московский государственный технический университет и Н.Э. Баумана
Факультет ИУ «Информатика и системы управления»
Кафедра ИУ-5 «Системы обработки информации и управления»

ОТЧЕТ
по домашнему заданию

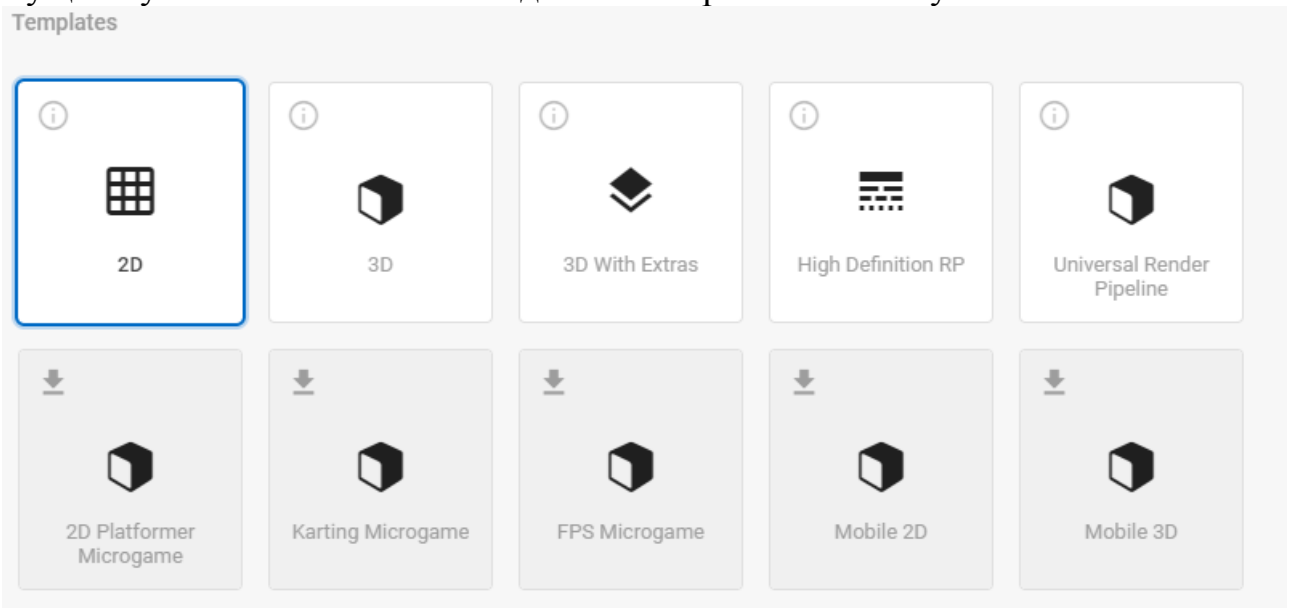
Выполнил:
Студент ИУ5-32Б
Ныркov Илья Алексеевич

Проверил:
Доцент Гапанюк Ю. Е.

МОСКВА 2020

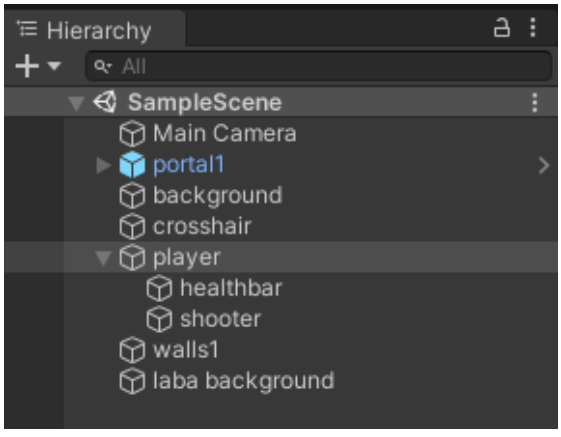
Описание задания – Освоить основные инструменты в игровом движке Unity. Изучить способы написания игровой логики с использованием языка программирования C# и изученных в лабораторных работах тем.

Описание работы и создания игровых приложений в программе Unity – Существует несколько типов создаваемых проектов в Unity:

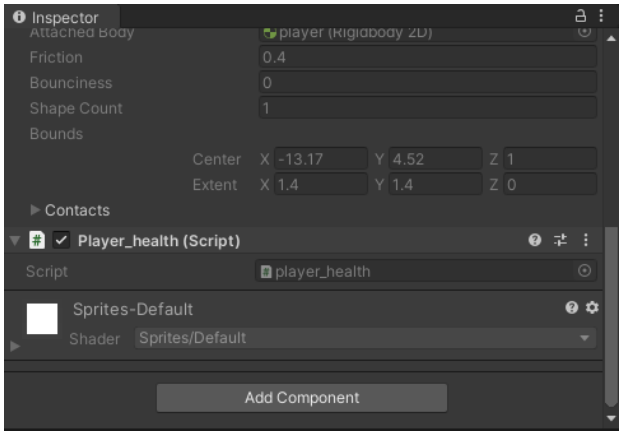


Каждый из этих типов можно реализовать через тип 3D, однако это займёт больше времени, чем использование готового шаблона игры (при профессиональной разработке используются 3D или 2D, так как ставиться цель создать новый оригинальны продукт). Проект 2D отличается от 3D тем, что ось Z направлена на пользователя (проецируется в точку).

Работа с объектами осуществляется в интуитивно понятном интерфейсе
В разделе **“иерархия” (рис. 1)** можно создавать объекты, которые будут находится на сцене и префабы (prefab) – готовые шаблоны объектов, которые можно использовать дальше. После создания объекта, ему можно устанавливать различные компоненты в разделе “inspector” (рис. 2) при помощи кнопки Add Component.



(рис. 1)



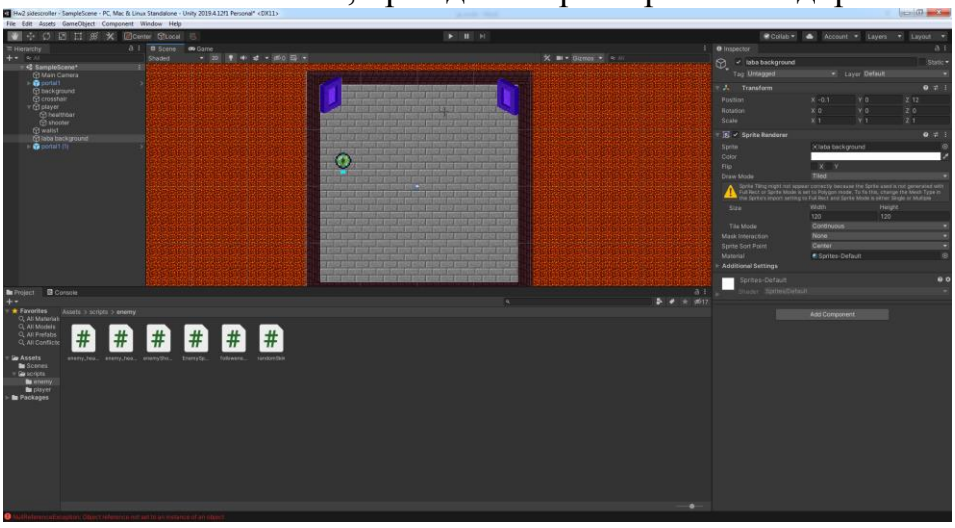
(рис. 2)

Существуют различные виды **компонентов**, отвечающих за физику объектов **(RigidBody2D)**, спрайты – графическое изображение для объекта **(SpriteRenderer)**, и другие свойства. Главным компонентом является - C# скрипт **(C# script)** – это файл с кодом на C#, в котором описываются свойства и поведение объекта на программном уровне. Возможно использование только скриптов при разработке игры, однако использование готовых и отлаженных программных средств для конкретных задач (компонентов) по типу **RigidBody (физика перемещения объектов)**, **SpriteRenderer (вид объекта)**, **BoxCollider (физика столкновения объектов)** гораздо эффективнее.

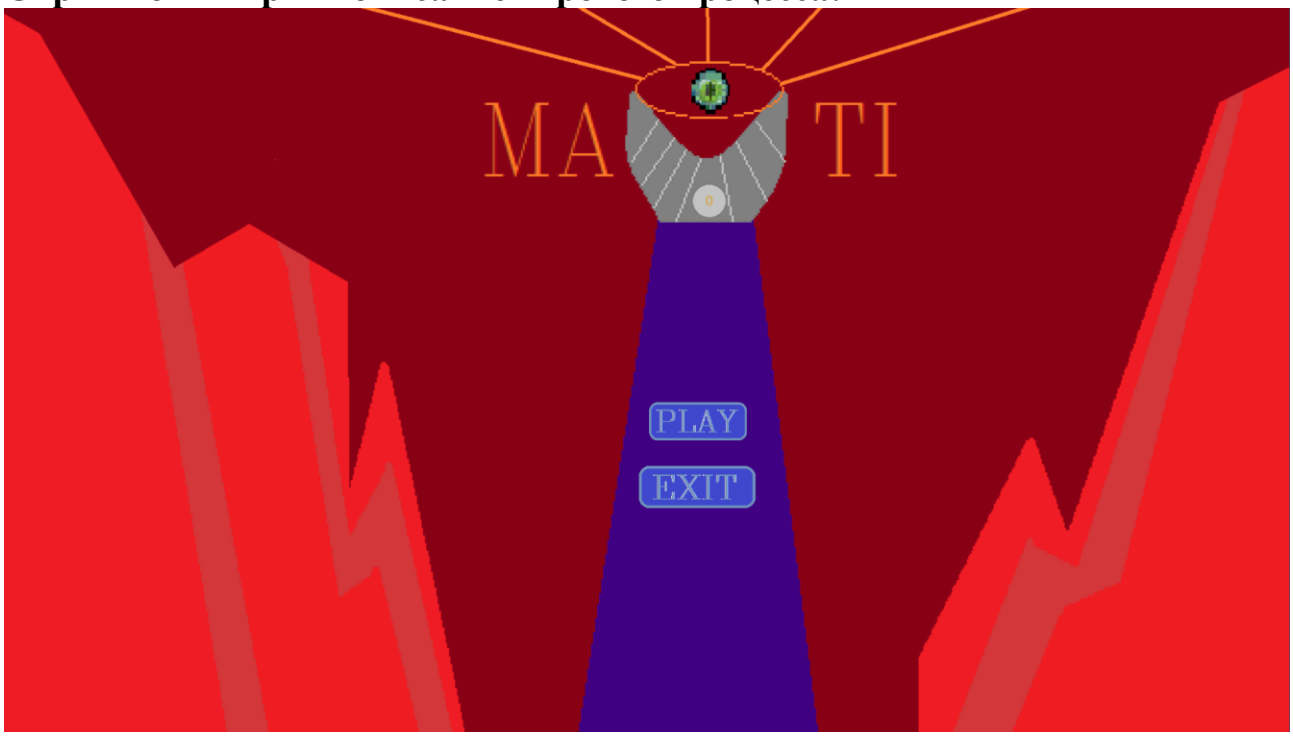
Начальный код C# скрипта:

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine; // Встроенная библиотека для работы с компонентами Unity

public class NewBehaviourScript : MonoBehaviour
{
    //Переменные

    // Метод запускающийся при запуске игры (используется для назначения переменных,
    // создания объекта и т. п.
    void Start()
    {

    }

    // Метод обновления по частоте такта процессора
    // Проигрывает код в теле метода каждый такт
    // Существует похожий метод void fixedUpdate(),
    // работающий также как Update, однако обновление
    // происходит в зависимости от заданного fps (количество кадров в секунду)
    void Update()
    {

    }
}
```

Описание, разработанной в ходе выполнения домашней работы, игры – Игра разработана в жанре “Top-down shooter”. Игрок может передвигаться во всех направлениях и стрелять в направлении и прицела. Враги появляются на арене из 2х порталов, расположенных по бокам. Задача игрока набрать максимальное количество очков, прежде чем растерять всё здоровье.



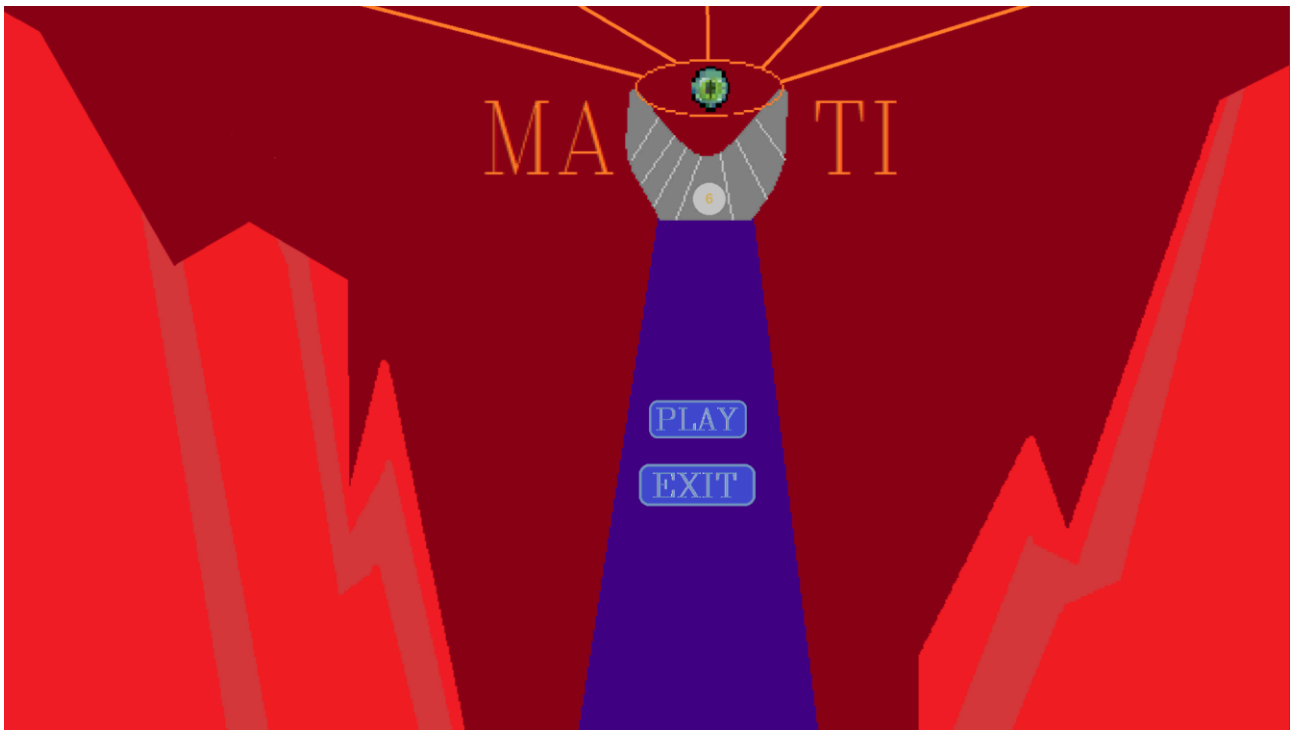
Скриншоты игры и описание игрового процесса:



меню игры

игровой процесс

В левом верхнем углу указывается количество набранных очков. Игрок перемещается по арене и стреляет в случайно появляющихся врагов и набирает очки. В случае потери всего здоровья от касаний врагов игрок возвращается в главное меню, где может выйти из игры или начать заново, для того чтобы побить свой рекорд.



Меню игры, вызываемое при запуске игры и после потери всего здоровья игроком

В сером круге на вершине башни указан рекорд игрока.

С целью усложнения игры, со временем увеличивается скорость появления врагов, враги появляются в случайных позициях. Данные возможности реализованы только при помощи скриптов C#.

Примеры C# скриптов (все файлы скриптов будут указаны в репазитории Github, ввиду их большого количества)

PointAndShot (стрельба и прицел)

```
using System.Collections;
using System.Collections.Generic;
using System.IO;
using UnityEngine;

public class pointANDshoot : MonoBehaviour
{
    // Start is called before the first frame update
    public GameObject crosshairs;
    public GameObject player;
    //blazerod
    public GameObject blazerodPrefab;
    public float blazerodSpeed = 60.0f;
    public int ShootCoolDown = 100;
    public GameObject rodStart;
    //player
    private Vector3 target;
    void Start()
    {
        Cursor.visible = false;
    }

    // Update is called once per frame
    int shootCD = 0;

    void Update()
    {
        target = transform.GetComponent<Camera>().ScreenToWorldPoint
            (new Vector3(Input.mousePosition.x, Input.mousePosition.y, transform.posi-
tion.z));
        //определение положения мыши в самой игре
        crosshairs.transform.position = new Vector2(target.x, target.y);
        Vector3 difference = target - player.transform.position;
        float rotationZ = Mathf.Atan2(difference.y, difference.x) * Mathf.Rad2Deg;
        player.transform.rotation = Quaternion.Euler(0.0f, 0.0f, rotationZ);
        shootCD++;
        if (Input.GetMouseButtonDown(0))
        {
            float distance = difference.magnitude;
            Vector2 direction = difference / distance;
            direction.Normalize();
            if (shootCD >= ShootCoolDown)
            {
                throwRod(direction, rotationZ);
                shootCD = 0;
            }
        }
    }
    void throwRod(Vector2 direction, float rotationZ)
    {
        GameObject b = Instantiate(blazerodPrefab);
        b.transform.position = rodStart.transform.position;
        b.transform.rotation = Quaternion.Euler(0.0f, 0.0f, rotationZ);
        b.GetComponent<Rigidbody2D>().velocity = direction * blazerodSpeed;
        Destroy(b, 1);
    }
}
```

PlayerHealth (система здоровья и получения повреждений от врагов)

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine.SceneManagement;
```

```
using UnityEngine;

public class player_health : MonoBehaviour
{
    Rigidbody2D rb;
    public static float healthAmount;
    void Start()
    {
        healthAmount = 0.3f;
        rb = GetComponent<Rigidbody2D>();
    }
    void Update()
    {
        if (healthAmount <= 0)
        {
            Cursor.visible = true;
            SceneManager.LoadScene("menu");
            Destroy(gameObject);
        }
    }
    private void OnCollisionEnter2D(Collision2D col)
    {
        if (col.gameObject.name.Equals("enemyRandomSkin(Clone)"))
        {
            healthAmount -= 0.1f;
        }
    }
}
```

RandomSkin (случайная текстура врага при создании)

```
using System;
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class randomSkin : MonoBehaviour
{
    public List<Sprite> sprites;
    public SpriteRenderer spriteRenderer;
    void Start()
    {
        System.Random random = new System.Random();
        int index = random.Next(10);
        spriteRenderer = gameObject.GetComponent<SpriteRenderer> ();
        spriteRenderer.sprite = sprites[random.Next(sprites.Count)];
    }
}
```

EnemySpawn (создание врагов в случайных точках с ускорением скорости появления врагов)

```
using System.Collections;
using System.Collections.Generic;
using System;
using UnityEngine;

public class EnemySpawner : MonoBehaviour
{
    public float timeForSpawn = 200;
    public float spawnAcceleration = 0;
    public GameObject enemyPrefab;
```

```
    public List<GameObject> placesToSpawn;
    private float timeForSpawnAndAcceleration;
    [SerializeField]
    private float spawnTimer = 0;
    private System.Random random;
    private void Start()
    {
        random = new System.Random();
        timeForSpawnAndAcceleration = timeForSpawn;
    }
    void FixedUpdate()
    {
        spawnTimer += Time.deltaTime;
        if (player_health.healthAmount <= 0)
        {
            timeForSpawnAndAcceleration = timeForSpawn;
        }
        if (spawnTimer >= timeForSpawnAndAcceleration && player_health.healthAmount > 0)
        {
            spawnTimer = 0;
            Instantiate(enemyPrefab, placesToSpawn[random.Next(placesToSpawn.Count)].trans-
form);
            if (timeForSpawnAndAcceleration > 0.5f)
            {
                timeForSpawnAndAcceleration -= spawnAcceleration;
            }
        }
    }
}
```

followenemy (Преследование игрока врагами)

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class followenemy : MonoBehaviour
{
    Transform player;
    public float moveSpeed = 5f;
    private Rigidbody2D rb;
    private Vector2 movement;
    void Start()
    {
        rb = this.GetComponent<Rigidbody2D>();
        player = GameObject.Find("player").GetComponent<Transform>();
    }
    void Update()
    {
        Vector3 direction = player.position - transform.position;
        float angle = Mathf.Atan2(direction.y, direction.x) * Mathf.Rad2Deg;
        rb.rotation = angle;
        direction.Normalize();
        movement = direction;
    }
    private void FixedUpdate()
    {
        moveCharacter(movement);
    }
    void moveCharacter(Vector2 direction)
    {
        rb.MovePosition((Vector2)transform.position + (direction * moveSpeed * Time.del-
taTime));
    }
```

```
}
```

MainMenu (главное меню игры с двумя методами для нажимаемых кнопок)

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using UnityEngine.SceneManagement;
public class mainMenu : MonoBehaviour
{
    public void PlayGame ()
    {
        SceneManager.LoadScene("Game");
    }
    public void ExitGame ()
    {
        Debug.Log("Quit");
        Application.Quit();
    }
}
```