МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
Высшего профессионального образования
«Вятский государственный университет»
Факультет автоматики и вычислительной техники
Кафедра электронных вычислительных машин

Отчет по лабораторной работе дисциплины «Технологии программирования» «Реализация и документирования программного обеспечения»

Выполнил студент группы ИВТб-32 _____/Серебряков М.А/
Проверил преподаватель_____/ /

Киров 2020

1. Листинг программы.

AnimationController:

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

public class AnimationController : MonoBehaviour
{
    public Animator anim;
    void Start()
    {
        anim = GetComponent<Animator>();
    }

    void Update()
    {
        if (Input.GetButtonDown("Fire1"))
        {
            anim.SetBool("fireBool",true);
        }
        else
        {
            anim.SetBool("fireBool", false);
        }
        if (Input.GetButton("Horizontal") || Input.GetButton("Vertical"))
        {
            anim.SetBool("runBool", true);
        }
        else
        {
            anim.SetBool("runBool", false);
        }
    }
}
```

CameraController:

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

public class CameraController : MonoBehaviour
{
    CharacterController PlayerBody;

    public float Sens = 5;
    float MoveY, MoveX;

    public Vector2 MinMax_Y = new Vector2(-40, 40),    //ограничение по оси Y
        MinMax_X = new Vector2(-360, 360);  //ограничение по оси X
    static float ClampAngle(float angle, float min, float max)
    {
        //если угол прошел расстояние от 0 до -360 то обнуляем его
        if (angle < -360F) angle += 360F;
        //если угол прошел расстояние от 0 до 360 то обнуляем его
        if (angle > 360F) angle -= 360F;
        //рассчитываем среднее значение поворота относительно угла
        return Mathf.Clamp(angle, min, max);
    }
    void Start()
    {
        PlayerBody = this.GetComponent<CharacterController>();
    }

    void Update()
    {
        //получаем угол на который мышь улетела от центра экрана по Y
        MoveY -= Input.GetAxis("Mouse Y") * Sens;
        //ограничиваем угол поворота камеры чтобы она не вращалась под ноги
        MoveY = ClampAngle(MoveY, MinMax_Y.x, MinMax_Y.y);
        //получаем угол на который мышь улетела от центра экрана по X
        MoveX += Input.GetAxis("Mouse X") * Sens;
        //ограничиваем угол поворота камеры чтобы она не вращалась по оси X
        MoveX = ClampAngle(MoveX, MinMax_X.x, MinMax_X.y);
        //поворачиваем тело персонажа по осям
        PlayerBody.transform.rotation = Quaternion.Euler(MoveY, MoveX, 0);
    }
}
```

EnemyController:

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

public class EnemyController : MonoBehaviour
{
    private NavMeshAgent agent;
    private GameObject playerObj;
    public GameObject mark;

    public int botDamage = 20;
    public float botFireRate = .25f;
    public float botShootRange = 40f;
    private float botNextFire;
    void Start()
    {
        agent = GetComponent<NavMeshAgent>();
        playerObj = GameObject.FindGameObjectWithTag("Player");
    }

    void Update()
    {
        agent.SetDestination(playerObj.transform.position);

        if (Time.time > botNextFire)
        {
            botNextFire = Time.time + botFireRate;

            if (Physics.Raycast(mark.transform.position,
mark.transform.forward, out RaycastHit hit, botShootRange))
            {
                PlayerController health =
hit.collider.GetComponent<PlayerController>();
                if (health != null)
                {
                    health.DamagePlayer(botDamage);
                }
            }
        }
    }
}
```

PlayerController:

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

public class PlayerController : MonoBehaviour
{
    public CharacterController controller;

    public int healthPlayer = 100;
    public float speed = 4f;
    public float horizontal;
    public float vertical;
    public GameObject MyCamera;
    Vector3 DirCam;

    public GameObject guiTextLink;

    void Update()
    {
        horizontal = Input.GetAxisRaw("Horizontal"); //A = -1 D = +1
        vertical = Input.GetAxisRaw("Vertical"); //S = -1 W = +1
        DirCam = MyCamera.transform.forward;
        DirCam.y = 0;
        Vector3 direction = new Vector3(horizontal, 0f, vertical).normalized;
        direction = Quaternion.LookRotation(DirCam) * direction;

        if (direction.magnitude >= 0.1f)
        {
            controller.Move(direction * speed * Time.deltaTime);
        }

    }

    public void DamagePlayer(int damageAmount)
    {
        healthPlayer -= damageAmount;
        guiTextLink.GetComponent<Text>().text = healthPlayer.ToString();

        if (healthPlayer <= 0)
        {
            SceneManager.LoadScene("Demo 1");
            Cursor.visible = true;
        }
    }
}
```

SceneController:

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

public class SceneController : MonoBehaviour
{
    public void LoadScene(string sceneName)
    {
        SceneManager.LoadScene(sceneName);
    }
    public void Exit()
    {
        Application.Quit();
    }
}
```

ScoreUpdate:

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

public class ScoreUpdate : MonoBehaviour
{
    void Start()
    {
        GetComponent<Text>().text = ShootableEnemy.score.ToString();
    }

}
```

ShootableEnemy:

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

public class ShootableEnemy : MonoBehaviour
{
    public int enemyHealth = 100;
    public static int score;

    public void getDamage(int damageAmount)
    {
        enemyHealth -= damageAmount;

        if (enemyHealth <= 0)
        {
            gameObject.SetActive(false);
            score++;
        }
    }
}
```

ShootingController:

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

public class ShootingConroller : MonoBehaviour
{
    public int playerDamage = 30;
    public float fireRate = .25f;
    public float shootRange = 40f;

    private Camera thirdCam;
    private float nextFire;
    void Start()
    {
        thirdCam = GetComponent<Camera>();
        Cursor.visible = false;
        ShootableEnemy.score = 0;
    }

    void Update()
    {
        if (Input.GetButtonDown("Fire1") && Time.time > nextFire)
        {
            nextFire = Time.time + fireRate;

            Vector3 rayOrigin = thirdCam.ViewportToWorldPoint(new
Vector3(.5f, .5f, 0)); //Центральная точка

            //RaycastHit hit;

            if(Physics.Raycast(rayOrigin,thirdCam.transform.forward,out
RaycastHit hit,shootRange))
            {
                ShootableEnemy health =
hit.collider.GetComponent<ShootableEnemy>();
                if (health != null)
                {
                    health.getDamage(playerDamage);
                }
            }
        }
    }
}
```

Spawner:

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

public class Spawner : MonoBehaviour
{
    public GameObject enemy;
    public float dTime = 5f;
    private float nextSpawn;

    void Start()
    {
        nextSpawn = Time.time + dTime;
        Instantiate(enemy, this.transform.position, this.transform.rotation);
    }
    void Update()
    {
        if (Time.time > nextSpawn)
        {
            nextSpawn = Time.time + dTime;

Instantiate(enemy,this.transform.position,this.transform.rotation);
        }
    }
}
```