Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Факультет электронно-информационных систем

Кафедра ИИТ

Лабораторная работа №1-2

за 5 семестр

По дисциплине: «ОСиСП»

Выполнил:

студентк 3 курса

группы ПО-4(1)

Ковальчук В. В.

Проверила:

Дряпко А.В.

2021

Лабораторная работа №1-2

Цель работы: приобрести практические навыки проектирования и разработки приложений с графическим пользовательским интерфейсом в ОС Windows средствами Qt.

Вариант 12

**Задание:**

Игра «Space Invaders». Реализовать один уровень игры с 3 типами кораблей инопланетян. Первый тип находится в первых двух рядах и представляет собой легкий класс кораблей, которые уничтожаются одним выстрелом. Второй тип занимает следующие 2 ряда и представляет класс кораблей с улучшенной броней. Они уничтожаются с двух выстрелов. Наконец, третий тип – тяжелые корабли – уничтожается с трех выстрелов и занимает один ряд. Игрок может прятаться за четырьмя импровизированными щитами-заслонами, которые постепенно разрушаются от каждого выстрела захватчиков. Игрок может стрелять и уничтожать корабли космических захватчиков. Они в свою очередь тоже могут стрелять по игроку. Корабли инопланетян постепенно приближаются к игроку. После уничтожения каждого типа скорость их движения увеличивается.

**Код:**

**Player.cs**

using UnityEngine;

using UnityEngine.SceneManagement;

public class Player : MonoBehaviour

{

    public Projectile lasetPrefab;

    public float speed = 5.0f;

    private bool \_laserActive;

    public int Health;

    private void Update()

    {

        if(Input.GetKey(KeyCode.A) || Input.GetKey(KeyCode.LeftArrow)){

            this.transform.position += Vector3.left \* this.speed \* Time.deltaTime;

        } else if(Input.GetKey(KeyCode.D) || Input.GetKey(KeyCode.RightArrow)){

            this.transform.position += Vector3.right \* this.speed \* Time.deltaTime;

        }

        if(Input.GetKeyDown(KeyCode.Space) || Input.GetMouseButtonDown(0))

        {

            Shoot();

        }

    }

    private void Shoot()

    {

        if(!\_laserActive){

            Projectile projectile = Instantiate(this.lasetPrefab, this.transform.position, Quaternion.identity);

            projectile.destroyed += LaserDestroyed;

            \_laserActive = true;

        }

    }

    private void LaserDestroyed()

    {

        \_laserActive = false;

    }

    private void OnTriggerEnter2D(Collider2D other)

    {

        if(other.gameObject.layer == LayerMask.NameToLayer("Invader")||

        other.gameObject.layer == LayerMask.NameToLayer("Missile"))

        {

            Helper.Helperdll.PLayerRecieveDamage(ref this.Health);

            if(this.Health <= 0)

            {

                SceneManager.LoadScene(SceneManager.GetActiveScene().name);

            }

        }

    }

}

**Invader.cs**

using UnityEngine;

using Helper;

public class Invader : MonoBehaviour

{

    public Sprite[] animationSprites;

    public float animationTime = 1.0f;

    public System.Action killed;

    private SpriteRenderer \_spriteRenderer;

    private int \_animationFrame;

    public int health;

    private void Awake()

    {

        \_spriteRenderer = GetComponent<SpriteRenderer>();

    }

    private void Start()

    {

        InvokeRepeating(nameof(AnimateSprite), this.animationTime, this.animationTime);

    }

    private void AnimateSprite()

    {

        \_animationFrame++;

        if(\_animationFrame >= this.animationSprites.Length)

        {

            \_animationFrame = 0;

        }

        \_spriteRenderer.sprite = this.animationSprites[\_animationFrame];

    }

    private void OnTriggerEnter2D(Collider2D other)

    {

        if(other.gameObject.layer == LayerMask.NameToLayer("Laser"))

        {

            Helper.Helperdll.EnemyRecieveDamage(ref this.health);

        }

        if(health<=0)

        {

            this.killed.Invoke();

            this.gameObject.SetActive(false);

        }

    }

}

**Invaders.cs**

using UnityEngine;

using UnityEngine.SceneManagement;

using Helper;

public class Invaders : MonoBehaviour

{

    public Invader[] prefabs;

    public int rows = 5;

    public int columns = 11;

    public AnimationCurve speed;

    public Projectile missilePrefab;

    public float missileAttackRate = 1.0f;

    public int amountKilled = 0;

    public int amountAlive => this.totalInvaders - this.amountKilled;

    public int totalInvaders => this.rows \* this.columns;

    public float percentKilled => (float)this.amountKilled / (float)this.totalInvaders;

    private Vector3 \_direction = Vector3.right;

    private void Awake()

    {

        for(int row = 0; row < this.rows; row++)

        {

            float width = 2.0f \* (this.columns - 1);

            float height = 2.0f \* (this.rows - 1);

            Vector2 centering = new Vector2(-width/2, -height / 2);

            Vector3 rowPosition = new Vector3(centering.x ,centering.y + (row \* 2.0f), 0.0f);

            for(int col = 0; col < this.columns; col++)

            {

                Invader invader = Instantiate(this.prefabs[row], this.transform);

                Vector3 position = rowPosition;

                invader.killed += InvaderKilled;

                position.x += col \* 2.0f;

                invader.transform.localPosition = position;

            }

        }

    }

    private void Start()

    {

        InvokeRepeating(nameof(MissileAttack), this.missileAttackRate, this.missileAttackRate);

    }

    private void Update()

    {

        this.transform.position += \_direction \* this.speed.Evaluate(this.percentKilled) \* Time.deltaTime;

        Vector3 leftEdge = Camera.main.ViewportToWorldPoint(Vector3.zero);

        Vector3 rightEdge = Camera.main.ViewportToWorldPoint(Vector3.right);

        foreach(Transform invader in this.transform)

        {

            if(!invader.gameObject.activeInHierarchy)

            {

                continue;

            }

            if(\_direction == Vector3.right && invader.position.x >= (rightEdge.x - 1.0f))

            {

                AdvanceRow();

            }

            else if(\_direction == Vector3.left && invader.position.x <= (leftEdge.x + 1.0f))

            {

                AdvanceRow();

            }

        }

    }

    private void AdvanceRow()

    {

        \_direction.x \*= -1.0f;

        Vector3 position = this.transform.position;

        position.y -= 1.0f;

        this.transform.position = position;

    }

    private void MissileAttack()

    {

        foreach(Transform invader in this.transform)

        {

            if(!invader.gameObject.activeInHierarchy)

            {

                continue;

            }

            if(Random.value < (1.0f / (float)this.amountAlive))

            {

                Instantiate(this.missilePrefab, invader.position, Quaternion.identity);

                break;

            }

        }

    }

    private void InvaderKilled()

    {

        this.amountKilled++;

        Helper.Helperdll.EnemyKilled(ref this.amountKilled);

        if(this.amountKilled >= this.totalInvaders)

        {

            SceneManager.LoadScene(SceneManager.GetActiveScene().name);

        }

    }

}

**Projectile.cs**

using UnityEngine;

public class Projectile : MonoBehaviour

{

    public Vector3 direction;

    public float speed;

    public System.Action destroyed;

    private void Update()

    {

        this.transform.position += this.direction \* this.speed \* Time.deltaTime;

    }

    private void OnTriggerEnter2D(Collider2D other)

    {

        if(this.destroyed != null)

        {

            this.destroyed.Invoke();

        }

        Destroy(this.gameObject);

    }

}

**Bunker.cs**

using UnityEngine;

public class Bunker : MonoBehaviour

{

    private SpriteRenderer sp;

    private int health = 3;

    private void Awake()

    {

        sp = GetComponent<SpriteRenderer>();

    }

    private void OnTriggerEnter2D(Collider2D other)

    {

        if(other.gameObject.layer == LayerMask.NameToLayer("Missile")) {

            Color color = sp.material.color;

            color.a -= 0.3f;

            sp.material.color = color;

            health--;

        }

        if(health <= 0) {

            this.gameObject.SetActive(false);

        }

        if(other.gameObject.layer == LayerMask.NameToLayer("Invader")) {

            this.gameObject.SetActive(false);

        }

    }

}

**MainMenu.cs**

using UnityEngine;

using UnityEngine.SceneManagement;

using UnityEngine.UI;

using About;

public class MainMenu : MonoBehaviour

{

    public GameObject MessageBox;

    public Text textAboutinf;

    private AboutL information = new AboutL();

    public Text textPlay;

    public Text textExit;

    public Text textAbout;

    public void Play(int index)

    {

        SceneManager.LoadScene(index);

    }

    public void Exit()

    {

        Application.Quit();

    }

    public void About()

    {

        textAboutinf.text = information.InformationAbout();

        MessageBox.SetActive(!MessageBox.activeSelf);

    }

    public void Normal()

    {

        textPlay.fontStyle = FontStyle.Normal;

        textExit.fontStyle = FontStyle.Normal;

        textAbout.fontStyle = FontStyle.Normal;

    }

    public void Bold()

    {

        textPlay.fontStyle = FontStyle.Bold;

        textExit.fontStyle = FontStyle.Bold;

        textAbout.fontStyle = FontStyle.Bold;

    }

    public void Italic()

    {

        textPlay.fontStyle = FontStyle.Italic;

        textExit.fontStyle = FontStyle.Italic;

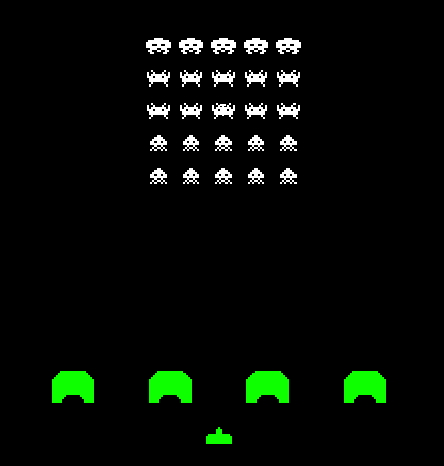
        textAbout.fontStyle = FontStyle.Italic;

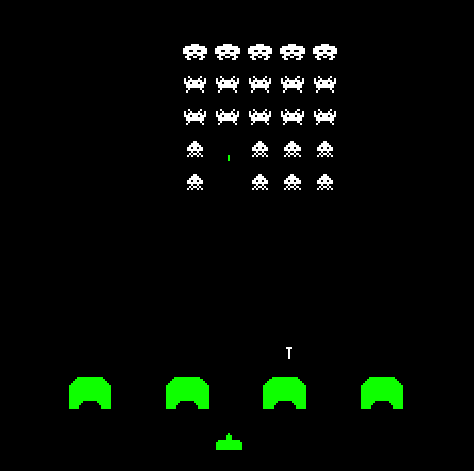
    }

}

**Результаты тестирования программы:**







**Вывод:** приобрёл практические навыки проектирования и разработки приложений с графическим пользовательским интерфейсом в ОС Windows средствами Qt.