CG MINI-PROJECT

GAME : SASSY ON FIRE !!

TITLE:

A game using unity - Sassy on fire !!

STATEMENT:

Design and implement game/animation clip / Graphics Editor using an open-source graphics library. Make use of maximum features of Object-Oriented Programming.

PREREQUISITE:

1.Basic knowledge about unity and its functions

2.Basic programming skills of C-sharp

THEORY:

(1) GAME ENGINE: UNITY

Unity is a cross-platform game engine developed by Unity Technologies. The engine can be used to create three-dimensional, two-dimensional, virtual reality, and augmented reality games and other experiences like simulations.

The engine has been adopted in various other industries such as film, automotive, architecture, engineering, construction along the gaming industry.

(2) UNITY 2D:

Most famous for its 3-D capabilities, Unity can also help make 2D games too. When in 2D mode, an orthographic view is set; the camera looks along the Z-axis with the Y-axis increasing upward.

- **2D GRAPHICS:** Sprites 2D graphic objects. Unity provides a built-in SpriteEditor to combine and manage sprite textures for efficiency and convenience during development.
- **2D PHYSICS:** Unity has a separate physics engine for handling 2D physics so as to make use of optimizations only available with 2D. Some of the most common components are: Rigidbody2D, Collider2D, Constant Force 2D, 2D Joints, etc.

(3) GAMEOBJECT:

GameObjects are the fundamental objects in Unity that represent characters, props and scenery. They are the building blocks for the scenes and also act as containers for Components, which implement the real functionality.

For example, a Light object is created by attaching a Light component to a GameObject.

A GameObject always has a Transform component attached (to represent position and orientation) and it is not possible to remove this. The other components that gi ve the object its functionality can be added from the editor's Component menu or from a script.

(4) SCRIPTING:

The behavior of **GameObjects**, in Unity, is controlled by the **Components** that are attached to them. Unity allows you to create your own Components using **scripts**.

Scripting (or creating scripts) is writing your own additions to the Unity Editor's functionality in code, using the Unity Scripting API.

These allow you to trigger game events, modify Component properties over time and respond to user inputs.

In technical terms, any script that is made compiles as a type of component, so the Unity Editor treats the script like a built-in component and executes whatever functionality that is written.

TAGS: A Tag is a reference word which can be assigned to one or more GameObjects. Tags help to identify GameObjects for scripting purposes and are useful for triggers in Collider control scripts.

For example, a "Player" Tag can be defined for player-controlled characters and an "Enemy" Tag for non-player-controlled characters.

(5) CLASS : MONOBEHAVIOUR:

The MonoBehaviour class provides the framework which allows you to attach your script to a GameObject in the editor, as well as providing hooks into useful Events such as Start() and Update().

In other words, this class links the Unity Game Engine features and the C# programming language.

(6) TRANSFORMS:

The Transform is used to store a GameObject's position, rotation, scale, and parenting state and is thus very important. A GameObject will always have a Transform component attached - it is not possible to remove a Transform or to create a GameObject without one.

(7) SCENES:

Scenes contain the objects of the game designed. They can be used to create the main menu, individual levels, and anything else. Each unique Scene File can be thought of as a unique level where the required environments, obstacles and decorations can be placed. In other words, the game is developed in pieces.

MULTI SCENE EDITING: Multi Scene Editing allows you to have multiple scenes open in the editor simultaneously and makes it easier to manage scenes at runtime. The ability to have multiple scenes open in the editor allows to create large streaming worlds and improve the workflow when collaborating on scene editing.

(8) AUDIO:

A game would be incomplete without some kind of audio, be it sound effects or background music. Unity's Audio features include full 3D spatial sound, real-time mixing and mastering, hierarchies of mixers, snapshots, predefined effects and much more. The audio system, provided, is flexible, powerful and offers a wide range of options to play with the sound effect.

CODES:

(1) PROJECTILE SPAWN:

```
using System.Collections;
using UnityEngine;

public class ProjectileSpawn : MonoBehaviour
{
    public GameObject[] enemyList;

    private float maxTime = 3.0f;
    private float minTime = 1.0f;


    void Start()
    {
        SpawnObstacle();
    }

    void SpawnObstacle()
    {
        int randomObstacle = Random.Range(0,enemyList.Length);
        Instantiate(enemyList[randomObstacle], transform.position, Quaternion.identity);
        float randTime = Random.Range(minTime, maxTime);
        if (!GameManager.isGameOver)
        {
            StartCoroutine(Delay());
            Invoke("SpawnObstacle", randTime);
        }

    }

    IEnumerator Delay()
    {
        yield return new WaitForSeconds(1);
```

```
    }
}
```

(2) REMOVE BG:

```
using UnityEngine;

public class RepeatBG : MonoBehaviour
{
    private Vector3 startPos;
    private float repeatWidth;

    void Start()
    {
        startPos = transform.position;
        repeatWidth = GetComponent<SpriteRenderer>().bounds.size.x/2;
    }

    void Update()
    {
        if (transform.position.x < startPos.x - repeatWidth)
            transform.position = startPos;
    }
}
```

(3) PLAYER CONTROLS:

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

public class PlayerControl : MonoBehaviour
{
    private Rigidbody2D playerRB;
    private float jumpForce=375;
    private bool isGround = true;

    public Animator animator;

    public AudioSource bgMusic;
    public AudioSource lobbyMusic;

    public ParticleSystem dirtParticle;
    public ParticleSystem deathExplosion;

    void Start()
```

```
    {
        playerRB = GetComponent<Rigidbody2D>();
    }

    void Update()
    {
        PlayerMovement();

        if(GameManager.isGameOver)
            bgMusic.enabled = false;
        if (!GameManager.isGameOver)
            lobbyMusic.enabled = false;
        animator.SetBool("isJumping", !isGround);

    }

    void PlayerMovement()
    {
        if((Input.GetKeyDown(KeyCode.Space) || Input.GetKeyDown(KeyCode.UpArrow))
&& isGround)
        {
            dirtParticle.Stop();
            playerRB.AddForce(Vector2.up * jumpForce);
            isGround = false;
        }
    }

    private void OnCollisionEnter2D(Collision2D collision)
    {
        if (collision.gameObject.tag == "BG")
        {
            isGround = true;
            dirtParticle.Play();
        }

        if (collision.gameObject.tag == "Avoid")
        {
            GameManager.isGameOver = true;
            dirtParticle.Stop();
            StartCoroutine("Delay");
            SceneManager.LoadScene("Outro");
        }

    }

    IEnumerator Delay()
    {
        yield return new WaitForSeconds(5f);
        deathExplosion.Play();
```

```
        animator.SetBool("GameOver", true);
    }
}
```

(4) MOVE LEFT:

```
using UnityEngine;

public class MoveLeft : MonoBehaviour
{
    public float speed = 5;
    private float leftBound = -30;
    private float bottomBound = -50;

    void Update()
    {
        if(!GameManager.isGameOver)
        {
            transform.Translate(Vector3.left * Time.deltaTime * speed);
        }
        if (transform.position.x < leftBound || transform.position.y < bottomBound)
        {
            Destroy(gameObject);
            ScoreCount.scoreValue += 5;
        }
    }
}
```

(5) SCORE COUNT:

```
using UnityEngine;
using UnityEngine.UI;

public class ScoreCount : MonoBehaviour
{
    public static int scoreValue = 0;
    public Text scoreText;

    private void FixedUpdate()
    {
        scoreText.text = "Score: " + scoreValue;
    }
}
```

(6) PLAYER SLIDE:

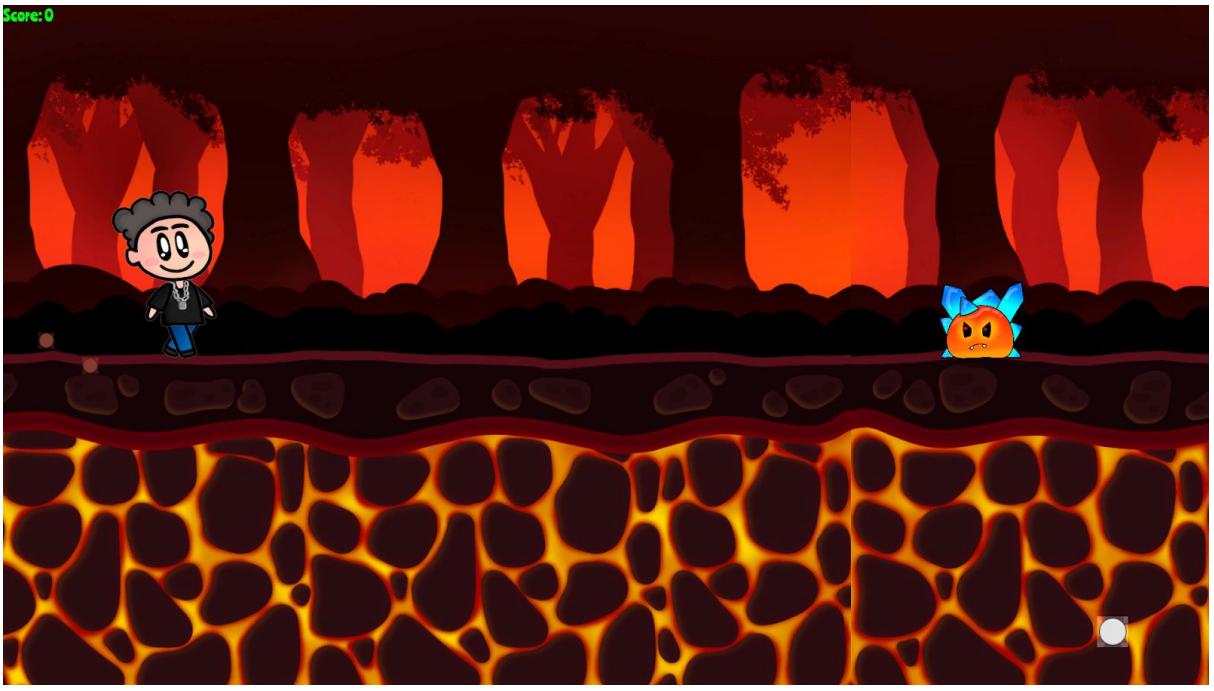
```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class PlayerSlide : MonoBehaviour
{
    private bool isSliding = false;
    public Animator anim;

    public Rigidbody2D rb2D;
    public BoxCollider2D regularColl;
    public BoxCollider2D slideColl;
    private float slideSpeed = 5f;

    private void Update()
    {
        if (Input.GetKeyDown(KeyCode.DownArrow))
            PerformSlide();

        anim.SetBool("isSlide", isSliding);

    }

    void PerformSlide()
    {
        isSliding = true;
        regularColl.enabled = false;
        slideColl.enabled = true;
        rb2D.AddForce(Vector2.right * slideSpeed);

        StartCoroutine("StopSlide");
    }

    IEnumerator StopSlide()
    {
        yield return new WaitForSeconds(.8f);
        isSliding = false;
        regularColl.enabled = true;
        slideColl.enabled = false;
    }

}
```

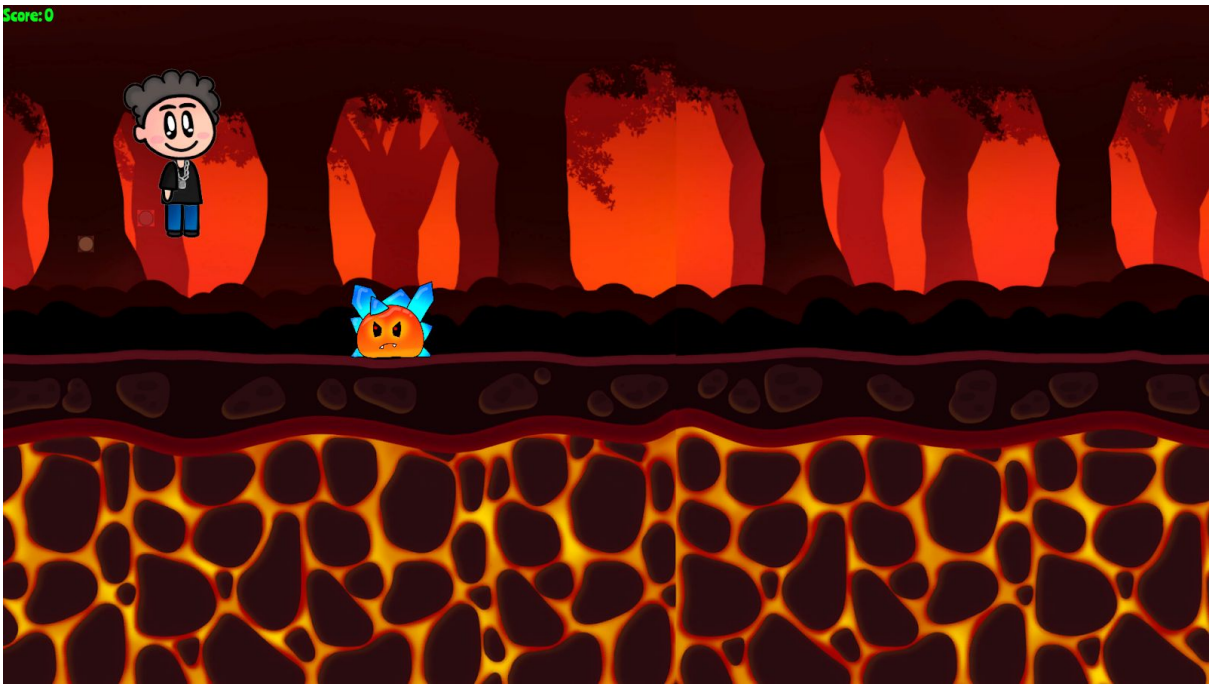
(7) GAME MANAGER:

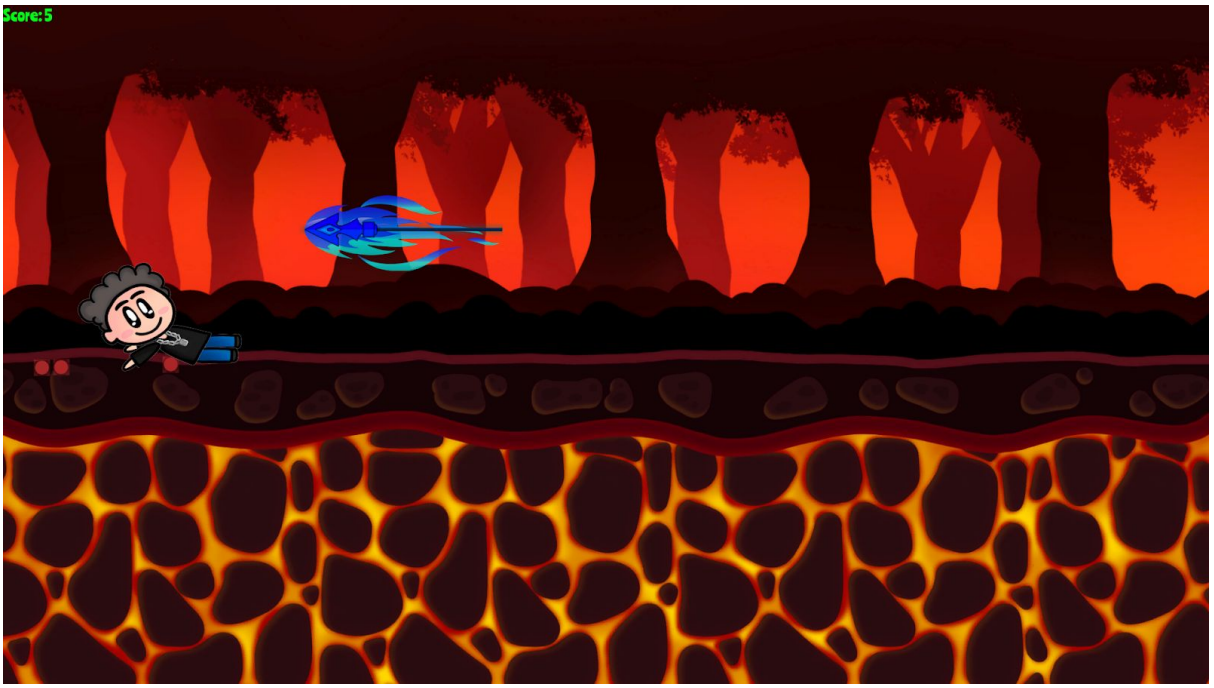
```
using UnityEngine;
using UnityEngine.SceneManagement;
using UnityEngine.UI;

public class GameManager : MonoBehaviour
{
    public GameObject mainScreen;
    public Text scoreText;
    public static bool isGameOver = true;

    void Update()
    {
        if (Input.GetKeyDown(KeyCode.R) && isGameOver)
        {
            SceneManager.LoadScene("MainGame");
            mainScreen.SetActive(false);
            scoreText.enabled = true;
            isGameOver = false;
        }
    }
}
```

OUTPUT:


SASSY ON FIRE
What are you waiting for?!
Space to jump and slide with down arrow.
Are you fast enough escape the fires and traps
of Mordor.....
Press 'R' to start!!!!


Score: 0


Score: 0



CONCLUSION:

A PROJECT BY -

SR NO	NAME	ROLL NO

1	Shashwat Tandon	21174
2	Shreya Chaudhary	21177
3	Gauri Takawane	21181