A blue and white logo

Description automatically generated

A logo of a university

Description automatically generatedA logo of a university

Description automatically generated**COMSATS University Islamabad (Lahore**A logo of a university

Description automatically generated **Campus)**

**<Assignment 4> – Fall 2023**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Course Title: | Game Development | Course Code: | CSC495 | Credit Hours: | 3 |
| Course Instructor/s: | Saira Aslam | Program Name: | BCS/BSE | | |
| **Submission Deadline** | **12 PM, 24th May 2024** | **Maximum Marks:** | **150** | | |
| **Important Instructions / Guidelines:**   * Submit a 10 seconds video, 4 screenshots, and all your scripts in Assignment 4 folder at google classroom * Paste your screenshots and scripts inside Assignment 4 Answer Template provided | | | | | |

***Question No 1.***

**Marks distribution:**

1. UI and Game flow =50
2. Instruction, Game semantics and error handling = 50
3. Authenticity and Novelty =50

***CLO: <4>; Bloom Taxonomy Level: <****Apply****>***

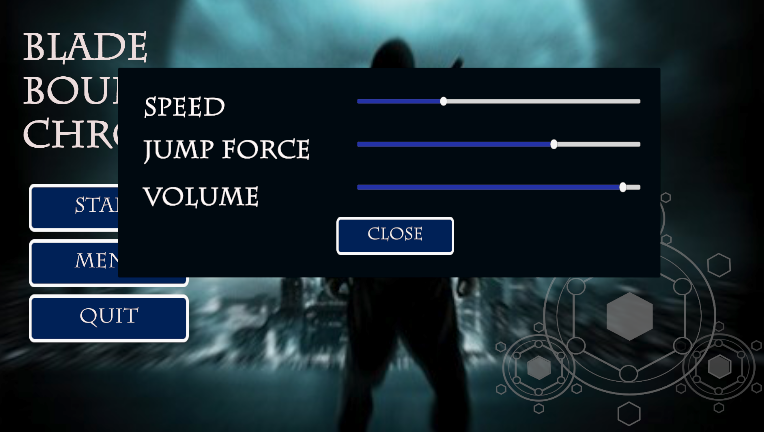
**Group Members:**

**Mr. Aoun Haider (FA21-BSE-133)**

**Talha Shafique (FA21-BSE-152)**

***Screenshots:***

****

****

****

****

****

****

***Scripts:***

**./Player.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

public class Player : MonoBehaviour,IDamageable

{

    public int diamonds = 0;

    new private AudioSource audio;

    private PlayerAnimation \_playerAnim;

    private Rigidbody2D \_rigid;

    [SerializeField]

    private float \_jumpForce = 2.0f;

    private bool \_grounded = false;

    [SerializeField]

    private LayerMask \_groundLayer;

    private bool \_resetJump = false;

    [SerializeField]

    private float \_speed = 3.0f;

    private SpriteRenderer \_playerSprite;

    private SpriteRenderer \_swordArcSprite;

    public int Health{get;set;}

    // Start is called before the first frame update

    void Start()

    {

        if(PlayerPrefs.HasKey("speed"))

          \_speed = PlayerPrefs.GetFloat("speed");

        if(PlayerPrefs.HasKey("jumpForce"))

          \_jumpForce = PlayerPrefs.GetFloat("jumpForce");

        \_rigid = GetComponent<Rigidbody2D>();

        \_playerAnim = GetComponent<PlayerAnimation>();

        \_playerSprite =  GetComponent<SpriteRenderer>();

        \_swordArcSprite = transform.GetChild(1).GetComponent<SpriteRenderer>();

        audio = GetComponent<AudioSource>();

        Health = 4;

    }

    // Update is called once per frame

    void Update()

    {

        if(transform.position.x > 93.0f)

        {

            Vector3 newPos = new Vector3(0.8f,0.88f,transform.position.z);

            transform.position = newPos;

        }

        Movement();

        if(Input.GetMouseButtonDown(0) && \_grounded)

        {

            \_playerAnim.Attack();

            audio.Play();

        }

    }

    public void Movement()

    {

        float horizontalInput = Input.GetAxisRaw("Horizontal");

        \_grounded = IsGrounded();

        if(horizontalInput > 0) //moving right

        {

            Flip(true);

        }

        else if(horizontalInput < 0) //moving left

        {

            Flip(false);

        }

        if(Input.GetKeyDown(KeyCode.Space) && IsGrounded())

        {

            \_rigid.velocity = new Vector2(\_rigid.velocity.x,\_jumpForce);

            StartCoroutine(ResetJumpNeededRoutine());

            \_playerAnim.Jump(true);

        }

        \_rigid.velocity = new Vector2(horizontalInput \* \_speed,\_rigid.velocity.y);

        \_playerAnim.Move(horizontalInput);

    }

    public bool IsGrounded()

    {

        RaycastHit2D hitInfo = Physics2D.Raycast(transform.position,Vector2.down,1.0f,\_groundLayer.value);

        Debug.DrawRay(transform.position,Vector2.down,Color.green);

        if(hitInfo.collider != null)

        {

            // Debug.Log(hitInfo.collider.name);

            if(!\_resetJump)

            {

                \_playerAnim.Jump(false);

                return true;

            }

        }

        return false;

    }

    public void Flip(bool isFacingRight)

    {

        if(isFacingRight){

            \_playerSprite.flipX = false;

            \_swordArcSprite.flipX = false;

            \_swordArcSprite.flipY = false;

            Vector3 newPos = \_swordArcSprite.transform.localPosition;

            newPos.x = 1.01f;

            \_swordArcSprite.transform.localPosition = newPos;

        }

        else{

            \_playerSprite.flipX = true;

            \_swordArcSprite.flipX = true;

            \_swordArcSprite.flipY = true;

            Vector3 newPos = \_swordArcSprite.transform.localPosition;

            newPos.x = -1.01f;

            \_swordArcSprite.transform.localPosition = newPos;

        }

    }

    IEnumerator ResetJumpNeededRoutine()

    {

        \_resetJump = true;

        yield return new WaitForSeconds(1.0f);

        \_resetJump = false;

    }

    public void Damage()

    {

        if(Health < 1) return;

        // Debug.Log("Player::Damage()");

        Health--;

        UIManager.Instance.UpdateLives(Health);

        if(Health<1)

        {

            \_playerAnim.Death();

            SceneManager.LoadScene(2);

        }

    }

    public void AddGems(int ammount)

    {

        diamonds += ammount;

        UIManager.Instance.UpdateGemCount(diamonds);

    }

}

**Hurdle.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Hurdle : MonoBehaviour

{

    private void OnCollisionEnter2D(Collision2D col)

    {

        if(col.gameObject.CompareTag("Player"))

        {

            Player player = col.gameObject.GetComponent<Player>();

            if(player != null)

            {

                player.Damage();

            }

        }

    }

}

**Shop.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.SceneManagement;

public class Shop : MonoBehaviour

{

    public GameObject shopPanel;

    public GameObject warningPanel;

    public int currentSelectedItem;

    public int currentSelectedItemCost;

    private Player player;

    private void OnTriggerEnter2D(Collider2D other)

    {

        if(other.tag == "Player")

        {

            player = other.GetComponent<Player>();

            if(player != null)

            {

                UIManager.Instance.OpenShop(player.diamonds);

            }

            shopPanel.SetActive(true);

        }

    }

    private void OnTriggerExit2D(Collider2D other)

    {

        if(other.tag == "Player")

        {

            shopPanel.SetActive(false);

        }

    }

    public void SelectItem(int item)

    {

        switch(item)

        {

            case 0:

             UIManager.Instance.UpdateShotSelection(83);

             currentSelectedItem = 0;

             currentSelectedItemCost = 200;

             break;

            case 1:

             UIManager.Instance.UpdateShotSelection(-10);

             currentSelectedItem = 1;

             currentSelectedItemCost = 400;

             break;

            case 2:

             UIManager.Instance.UpdateShotSelection(-103);

             currentSelectedItem = 2;

             currentSelectedItemCost = 100;

             break;

        }

    }

    public void BuyItem()

    {

        if(player.diamonds >= currentSelectedItemCost)

        {

            if(currentSelectedItem == 2)

            {

                Game\_Manager.Instance.HasKeyToCastle = true;

                SceneManager.LoadScene(3);

            }

            // player.diamonds -= currentSelectedItemCost;

            // Debug.Log("Purchased: "+currentSelectedItem);

            // Debug.Log("Remaining gems: "+player.diamonds);

        }

        else{

            // Debug.Log("You have no enough gems");

            warningPanel.SetActive(true);

            shopPanel.SetActive(false);

        }

    }

    public void CloseWarningPanel()

    {

        warningPanel.SetActive(false);

    }

}

**MainMenu.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

using UnityEngine.SceneManagement;

public class MainMenu : MonoBehaviour

{

    public Slider speedSlider, volumeSlider, jumpForceSlider;

    public GameObject menuPanel;

    public void StartButton()

    {

        AudioListener.volume = 1;

        SceneManager.LoadScene(1);

    }

    public void QuitButton()

    {

        Application.Quit();

    }

    public void ShowMenu()

    {

        menuPanel.SetActive(true);

    }

    public void CloseMenu()

    {

        menuPanel.SetActive(false);

        AudioListener.volume = volumeSlider.value;

        //Create prefs to be accessible in other scripts

        PlayerPrefs.SetFloat("speed",speedSlider.value\*10);

        PlayerPrefs.SetFloat("jumpForce",jumpForceSlider.value\*10);

        // Debug.Log(speedSlider.value+","+volumeSlider.value+","+jumpForceSlider.value);

    }

}

**UIManager.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

using UnityEngine.UI;

using UnityEngine.SceneManagement;

//SINGLETON CLASS

public class UIManager : MonoBehaviour

{

    private static UIManager \_instance;

    public GameObject playButton,pauseButton,musicOnButton,musicOffButton;

    public static UIManager Instance

    {

        get{

            if(\_instance == null)

            {

                Debug.Log("UI Manager is null!");

            }

            return \_instance;

        }

    }

    public Text playerGemCountText;

    public Image selectionImg;

    public Text gemCountText;

    public Image[] healthBars;

    public void OpenShop(int gemCount)

    {

        playerGemCountText.text = "" + gemCount + "G";

    }

    private void Awake()

    {

        \_instance = this;

    }

    public void UpdateShotSelection(int yPos)

    {

        selectionImg.rectTransform.anchoredPosition = new Vector2(selectionImg.rectTransform.anchoredPosition.x,yPos);

    }

    public void UpdateGemCount(int count)

    {

        gemCountText.text = "" + count;

    }

    public void UpdateLives(int livesRemaining)

    {

        for(int i=0;i<=livesRemaining;i++)

        {

            if(i==livesRemaining)

            {

                healthBars[i].enabled = false;

            }

        }

    }

    public void ResumeGame()

    {

        Time.timeScale = 0;

        pauseButton.SetActive(false);

        playButton.SetActive(true);

    }

    public void PauseGame()

    {

        Time.timeScale = 1;

        playButton.SetActive(false);

        pauseButton.SetActive(true);

    }

    public void ReloadGame()

    {

        SceneManager.LoadScene(0);

    }

    public void MusicOn()

    {

        AudioListener.volume = 0;

        musicOffButton.SetActive(true);

        musicOnButton.SetActive(false);

    }

    public void MusicOff()

    {

        AudioListener.volume = 1;

        musicOffButton.SetActive(false);

        musicOnButton.SetActive(true);

    }

}

**MossGiant.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class MossGiant : Enemy,IDamageable

{

    public int Health{get;set;}

    public override void Init()

    {

        base.Init();

        Health = base.health;

    }

    public override void Movement()

    {

        base.Movement();

    }

    public void Damage()

    {

        if(isDead) return;

        // Debug.Log("Moss Giant damage invoked!");

        \_anim.SetTrigger("hit");

        Health--;

        isHit = true;

        \_anim.SetBool("isCombat",true);

        if(Health < 0)

        {

            // Destroy(this.gameObject);

            \_anim.SetTrigger("Death");

            isDead = true;

            GameObject diamond = Instantiate(diamondPrefab,transform.position, Quaternion.identity);

            diamond.GetComponent<Diamond>().gems = base.gems;

        }

    }

}

**Enemy.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public abstract class Enemy : MonoBehaviour

{

    public GameObject diamondPrefab;

    [SerializeField]

    protected int health;

    [SerializeField]

    protected int speed;

    [SerializeField]

    protected int gems;

    protected bool isHit = false;

    [SerializeField]

    protected Transform pointA,pointB;

    protected Vector3 \_currentTarget;

    protected Animator \_anim;

    protected SpriteRenderer sprite;

    protected Player player;

    protected bool isDead = false;

    public virtual void Init()

    {

        \_anim = GetComponentInChildren<Animator>();

        sprite = GetComponentInChildren<SpriteRenderer>();

        player = GameObject.FindGameObjectWithTag("Player").GetComponent<Player>();

    }

    // Start is called before the first frame update

    private void Start()

    {

        Init();

    }

    // Update is called once per frame

    public virtual void Update()

    {

        //if idle animation is going on, do nothing

        if(\_anim.GetCurrentAnimatorStateInfo(0).IsName("idle") && !\_anim.GetBool("isCombat"))

        {

            return;

        }

        if(!isDead)

          Movement();

    }

    public virtual void Movement()

    {

        if(\_currentTarget == pointA.position)

        {

            sprite.flipX = true;

        }

        else{

            sprite.flipX = false;

        }

        if(transform.position == pointA.position)

        {

            \_currentTarget = pointB.position;

            \_anim.SetTrigger("idle");

        }

        else if(transform.position == pointB.position)

        {

            \_currentTarget = pointA.position;

            \_anim.SetTrigger("idle");

        }

        if(!isHit)

          transform.position = Vector3.MoveTowards(transform.position,\_currentTarget,speed\*Time.deltaTime);

        //check for distance b/w player and enemy

        float distance = Vector3.Distance(transform.localPosition,player.transform.localPosition);

        if(distance > 5.0f)

        {

            isHit = false;

            \_anim.SetBool("isCombat",false);

        }

        Vector3 direction = player.transform.localPosition - transform.localPosition;

        // Debug.Log("Side: " + direction.x);

        if(direction.x > 0 && \_anim.GetBool("isCombat")) //player is on enemys' right side

        {

            sprite.flipX = false;

        }

        else if(direction.x < 0 && \_anim.GetBool("isCombat"))

        {

            sprite.flipX = true;

        }

    }

}

**Acid\_Effect.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Acid\_Effect : MonoBehaviour

{

    // Start is called before the first frame update

    void Start()

    {

        Destroy(this.gameObject,5.0f); //destroy me after 5 seconds

    }

    // Update is called once per frame

    void Update()

    {

        transform.Translate(Vector3.right \* 3 \* Time.deltaTime);

    }

    public void OnTriggerEnter2D(Collider2D other)

    {

        if(other.tag == "Player")

        {

            IDamageable hit = other.GetComponent<IDamageable>();

            if(hit != null)

            {

                hit.Damage();

                Destroy(this.gameObject);

            }

        }

    }

}

**Spider.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Spider : Enemy,IDamageable

{

    public GameObject acidEffectPrefab;

    public int Health{ get; set; }

    public override void Init()

    {

        base.Init();

        Health = base.health;

    }

    public void Damage()

    {

        if(isDead) return;

        // Debug.Log("Spider::Damage()");

        Health--;

        if(Health < 1)

        {

            // Destroy(this.gameObject);

            \_anim.SetTrigger("Death");

            isDead = true;

            GameObject diamond = Instantiate(diamondPrefab,transform.position, Quaternion.identity);

            diamond.GetComponent<Diamond>().gems = base.gems;

        }

    }

    public override void Movement()

    {

        // base.Movement();

        //stand still there!!

    }

    public void Attack()

    {

        Instantiate(acidEffectPrefab,transform.position,Quaternion.identity);

    }

}

**Diamond.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Diamond : MonoBehaviour

{

    public int gems = 1;

    public void OnTriggerEnter2D(Collider2D other)

    {

        if(other.tag == "Player")

        {

            Player player = other.GetComponent<Player>();

            if(player != null)

            {

                player.AddGems(gems);

                Destroy(this.gameObject);

            }

        }

    }

}

**SpiderAnimationEffect.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class SpiderAnimationEffect : MonoBehaviour

{

    private Spider \_spider;

    private void Start()

    {

        \_spider = transform.parent.GetComponent<Spider>();

    }

    public void Fire()

    {

        \_spider.Attack();

    }

}

**Skeleton.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Skeleton : Enemy,IDamageable

{

    public int Health{get;set;}

    public override void Init()

    {

        base.Init();

        Health = base.health;

    }

    public override void Movement()

    {

        base.Movement();

    }

    public void Damage()

    {

        if(isDead) return;

        // Debug.Log("Skeleton damage invoked!");

        \_anim.SetTrigger("hit");

        Health--;

        isHit = true;

        \_anim.SetBool("isCombat",true);

        if(Health < 0)

        {

            // Destroy(this.gameObject);

            \_anim.SetTrigger("Death");

            isDead = true;

            GameObject diamond = Instantiate(diamondPrefab,transform.position, Quaternion.identity);

            diamond.GetComponent<Diamond>().gems = base.gems;

        }

    }

}

**IDamageable.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public interface IDamageable

{

    int Health {get;set;}

    void Damage();

}

**PlayerAnimation.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public interface IDamageable

{

    int Health {get;set;}

    void Damage();

}

**Attack.cs**

using System.Collections;

using System.Collections.Generic;

using UnityEngine;

public class Attack : MonoBehaviour

{

    private bool \_canDamage = true;

   private void OnTriggerEnter2D(Collider2D col)

   {

    //   Debug.Log("Hit: "+col.name);

      IDamageable hit = col.GetComponent<IDamageable>();

      if(hit != null)

      {

        if(\_canDamage)

        {

            hit.Damage();

            StartCoroutine(ResetDamageCoroutine());

        }

      }

   }

   IEnumerator ResetDamageCoroutine()

   {

      yield return new WaitForSeconds(0.5f);

      \_canDamage = true;

   }

}