Ollscoil na hÉireann, Gaillimh *GX____*
National University of Ireland, Galway

Semester I Examinations 2018/ 2019

Exam Code(s)	3BCT
Exam(s)	B.Sc. in Computer Science & Information Technology
Module Code(s)	CT3111
Module(s)	Next Generation Technologies
Paper No.	1
Repeat Paper	
External Examiner(s)	Dr. Jacob Howe
Internal Examiner(s)	*Dr. Sam Redfern

Instructions: Answer any three questions.
All questions carry equal marks.
Note that the final page of this exam paper lists useful classes from the Unity3D SDK.

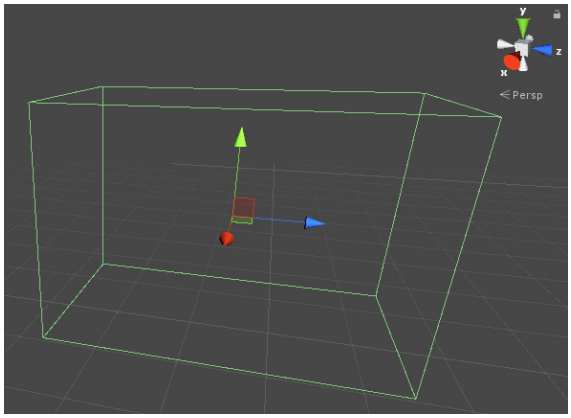
Duration	2 hours
No. of Pages	5
Department(s)	Information Technology
Course Co-ordinator(s)	Dr. Des Chambers

Requirements:
MCQ
Handout
Statistical Tables
Graph Paper
Log Graph Paper
Other Material

Q.1.

(i) Explain how Unity's MonoBehaviour class provides tight integration with the Game Loop. Refer to appropriate methods of the MonoBehaviour class in your answer. [6]

(ii) What is a Coroutine in Unity, and how do Coroutines integrate with the Game Loop? [4]



(iii) The Game Object depicted has a Box Collider component, whose 'isTrigger' property is true. A script on the game object contains a reference to the Box Collider and to a prefab of a ball.

```
public BoxCollider bc;
public GameObject ball;

public IEnumerator SpawnBallsInBox(){

}
```

Write code for the SpawnBallsInBox() coroutine, so that it continually instantiates balls, at a rate of one ball every two seconds. The balls should be initialised to a random position somewhere inside the Box Collider. (Hint: use the 'bounds' property of the Box Collider, which has 'min' and 'max' properties, each of which are of type Vector3). [10]

Q.2.

(i) Making appropriate use of local and global co-ordinates, write Unity3D/C# code to perform the following transformations. You may assume that references to the runtime Game Objects are provided:

- rotate a gameobject 5 degrees around its own z axis [3]
- move a gameobject 6 units upwards **per second**, in the world's co-ordinate system [4]
- move a gameobject 7 units directly towards another gameobject [4]

(ii) Write code for the following method, which considers the supplied list of objects and returns the one which is closest to the specified 3D point: [9]

```
public static GameObject GetClosestObject(List<GameObject> objects, Vector3 pos) {

}
```

Q.3.

(i) In 3D games development, what does the term **'raycast'** mean, as supported by various static methods of the Unity3D SDK's Physics class? [5]

(ii) Assuming you are writing a game in which computer-controlled enemies are chasing a player-controlled character, complete the code below, which is part of the EnemyController.cs script (comments marked 'to do' indicate where your code is needed)

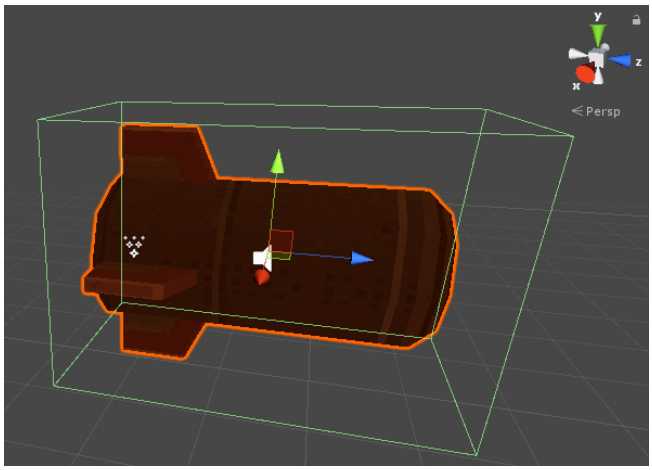
- uses raycasting to determine whether an enemy can see its target [4]
- (if it can see the target) whether it is facing towards it [5]
- (if facing towards it) accelerates forward. [3]
- (if not facing its target), a random rotation should be applied [3]

You can assume that chaseTarget (which is the game object being chased) and rigid (which is a reference to the Rigidbody of the enemy itself) have been initialised in the Inspector.

```
public GameObject chaseTarget;
public Rigidbody rigid;

void FixedUpdate() {

   // to do: perform a raycast to determine whether chaseTarget can be seen
   bool canSeeTarget = true;

   if (canSeeTarget) {
       // to do: use vector dot product to see if we're (almost) facing towards
the target
       bool facingTowardsTarget = true;

       if (facingTowardsTarget) {
           // to do: accelerate forwards

       }
       else {
           // to do: rotate randomly

       }
   }

}
```

Q.4.

The 'missile' game object depicted has a script attached, HomingMissile.cs. It also has a Rigibody component and a Box Collider. When the missile is launched, the LaunchMissile method is called, which initialises the velocity and target object which the missile will chase.



(i) Write code for the FixedUpdate() method (below) which will turn the missile slightly towards its target each frame. Hint: use the static method Vector3.RotateTowards(). Note that your code needs to change the direction that the missile is facing, and also modify its velocity to the new direction [10]

(ii) Write appropriate code, to add to the HomingMissile script below, to destroy the missile when it collides with something. [[5]

(iii) Your code should also destroy the object that was hit, but only if it has a Rigidbody. [5]

```
private GameObject homingTarget = null;

public void LaunchMissile(Vector3 direction, GameObject target) {
    GetComponent<Rigidbody>().velocity = direction * 20f;
    homingTarget = target;
}

void FixedUpdate() {
    if (homingTarget!=null) {
        // to do: turn towards target

    }
}
```

Q.5.

Write technical notes on the following: [5 x 4]

(i) How you would display (and update) a score on the screen while a game is being played, using the Unity GUI system
(ii) Entity Component Systems
(iii) The Object Pool pattern – why it's useful and how it operates
(iv) Screen space, viewport space and world space in Unity
(v) Coarse-to-fine algorithms

Some Useful Unity3D SDK Classes

GameObject: static methods

Instantiate()	Destroy()	DestroyImmediate()	Find()

GameObject: methods

AddComponent()	SendMessage()	GetComponent()	SetActive()

GameObject: data members

activeInHierarchy	transform	tag

MonoBehaviour: methods

Start()	OnDestroy()	Awake()	Update()
FixedUpdate()	LateUpdate()	OnDisable()	OnEnabled()
OnBecameInvisible()	OnBecameVisible()	OnCollisionEnter()	OnCollisionExit()
OnCollisionStay()	OnTriggerEnter()	OnTriggerExit()	OnTriggerStay()
SendMessage()	BroadcastMessage()	SendMessageUpwards()	GetComponent()
GetComponentInChildren()	GetComponentInParent()	GetComponents()	GetComponentsInChildren()
GetComponentsInParent()	GetInstanceID()	Invoke()	StartCoroutine()

MonoBehaviour: data members

enabled	gameObject	transform	name

Transform: methods

Rotate()	Translate()	TransformPoint()	InverseTransformPoint()
LookAt()	RotateAround()	SetParent()	TransformVector()
InverseTransformVector()	TransformDirection()	InverseTransformDirection()	

Transform: data members

position	localPosition	rotation	localRotation
lossyScale	localScale	parent	right
up	forward	gameObject	

RigidBody: methods

AddForce()	AddForceRelative()	AddForceAtPosition()	AddTorque()
AddRelativeTorque()	MovePosition()	MoveRotation()	

RigidBody: data members

drag	angularDrag	mass	velocity
angularVelocity	centerOfMass		

Camera: methods

ScreenToWorldPoint()	WorldToScreenPoint()	ScreenToViewportPoint()
ViewportToScreenPoint()	WorldToViewportPoint()	ViewportToWorldPoint()
ViewportPointToRay()	ScreenPointToRay()	

Physics: static methods

Raycast()	SphereCast()	OverlapBox()	BoxCast()