ANSWERS TO QUIZZES

CHECK YOUR ANSWERS AND SEE HOW MUCH YOU KNOW ABOUT UNITY…☺

Quiz 1

1. The following statement will print the text **Hello World** in the **Console** window. **TRUE**

```
print(“Hello World”);
```

2. The value of the variable **c** in the following statement will be **3**. **FALSE**

```
int a;
int b;
a = 1;
b =1;
c = a + b;
```

3. The value of the variable **fullName**, in the following code snippet, will be **JohnPaul**. **TRUE**

```
string fName = “John”;
string lName = “Paul”;
string fullName = fName + lName;
```

4. The following code snippet will print **I will not go sailing**. **TRUE**

```
bool windIsStrong;
windIsStrong = true;
if (windIsStrong) print (“I will not go sailing”);
```

5. The following code snippet will print **I will not go sailing**. **FALSE**

```
bool weatherIsSunny;
bool windIsStrong;
bool iWillGoSailing;
weatherIsSunny = true;
windIsStrong = false;
If (weatherIsSunny && !windIsStrong ) print (“I will go sailing”);
If (!weatherIsSunny || windIsStrong ) print (“I will not go sailing”);
```

6. Spot three coding mistakes in the following snippet.

```
int test >> ; missing
int test2;
in test3 = 0;
test 3 = test1 + test2;>> tes1 has not been declared
```

7. Consider the method described in the next code snippet, and select the correct way to call it (i.e., A, B, or C): **C**

a) **displayMessage();**
b) **displayAMessage()**
c) **displayAMessage();**

```
public void displayAMessage()
{
}
```

8. The value of the variable **counter** in the following code snippet will be **3** after the code has been executed. **<u>FALSE</u>**

```
int counter;
counter = 0;
counter = counter + 1;
```

9. The following code will print the message **Hello** every second. **<u>FALSE (every frame).</u>**

```
public void Update()
{
      print (“Hello”);
}
```

10. A local variable can be used from any part of a script. **<u>FALSE.</u>**

Quiz 2

1. A class can have more than two constructors. **TRUE**
2. Different constructors can have the same name. **TRUE**
3. A pubic variable can be accessed from anywhere in your programme. **TRUE**
4. When a new instance of an object is created, the corresponding constructor is called. **TRUE**
5. All classes created with Unity will inherit from the **Monobehaviour** class by default. **TRUE**
6. The name of a C# script, when created, will be the same for the class defined within this file. **TRUE**
7. So that it can be called from anywhere outside the class, a **getter** needs to be declared as public. **TRUE**
8. In C# the default access type for member variables and methods is **internal**. **FALSE (private).**
9. In **camel casing** the first character of each word is capitalized except for the first word. **TRUE**
10. In **Pascal casing** the first character of each word is capitalized. **TRUE**

Quiz 3

1. The method **onControllerColliderHit** is called whenever a collision occurs between the **ThirdPersonController** and anther object that includes a collider. **FALSE**

2. To be able to access a variable from a script through the **Inspector**, this variable has to be declared as **public** in the script. **TRUE**

3. Write the missing line in this code to be able to destroy the object we have collided with.

```
function OnCollisionEnter (Collision collision)
{
        Destroy (collision.collider.gameObject);
}
```

4. There is only one way to create a prefab in Unity, that is through the menu **Create | Prefab**. **FALSE** (you can also drag and drop object to **Project** view)

5. A mesh collider will detect collision more precisely than a capsule collider when applied to a spherical object. **TRUE**

6. Find one error in the following code.

```
void Start ()
{
        score = 0;
        GameObject.Find("message").GetComponent<UIText>().text     ="";
}
```

7. Any object selected in the **Hierarchy** window can be duplicated using the shortcut *CTRL + D*. **TRUE**

8. If the object attached to the next script has a **Rigidbody** component, the following code will access this component and apply a forward force to it. **TRUE**

```
gameObject.GetComponent<Rigidbody>().AddForce(transform.forward * 1000);
```

9. Explosions prefabs need to be imported using the **ParticleSystems** asset, in order to be used in Unity. **TRUE**

10. If the following error message appears "**Cannot implicitly convert type**", what do you need to do with the following code:

```
GameObject    t    =       Instantiate    (projectile,    transform.position,
Quaternion.identity);
```

a) Make sure that the type of the variable to the left of the = sign is the same as the type of the variable on right of the = sign.
b) Cast the variable to the right of the = sign using **(GameObject)**.
c) All of the above.

Quiz 4

1. A new prefab can be created by dragging and dropping an object to the **Project** window. **TRUE**

2. The following code will empty the text for the component named **userMessageUI**. **FALSE** (See missing code in bold)

```
GameObject.Find("userMessageUI").GetComponent.<UI.Text>().text ="";
```

3. To be able to instantiate a prefab, the following code could be used: **TRUE**

```
Instantiate (prefab, transform.position, Quaternion.identity);
```

4. Find one error in the following code (**"=" should read "=="**).

```
void OnControllerColliderHit (ControllerColliderHit hit)
{
      if (hit.collider.tag = "pick_me") print ("Collided with a box");
}
```

5. Any prefab can be duplicated using the shortcut *CTRL + F*. **FALSE (use CTRL + D instead)**

6. If the object **myObject** does not have a Rigidbody component, and the following code is used, an error message will be displayed in the **Console** window. **TRUE**

```
myObject.GetComponent<Rigidbody>().AddForce (transform.forward*100);
```

7. What does this error message most likely mean **"; missing"**.

   a) You have forgotten to declare a variable.
   b) One of the statements in your code is missing a semi-colon.
   c) The method that you have called does not exist.

8. There is only one way to add an **Audio Source** component to an object, and this is using the button **Add Component** button in the **Inspector** window for this object. **FALSE (also through the top menu)**

9. If the method **manageCollision** is defined as follows…

```
public void manageCollision()
{
      print ("Collision detected");

}
```

... it can be called from outside its containing class. **TRUE**

10. The following code will create an array and then access its first element. **FALSE (first elements starts at index 0 not 1)**

```
int [] myArray = new int [4];
int newVar = myArray [1];
```

<u>Quiz 5</u>

1. It is possible to duplicate an Animator Controller using the keys *CTRL + D*. **<u>TRUE</u>**
2. The following code will change the value of a Boolean parameter for an animation. **<u>FALSE .</u>**

```
SetParameter(“canSeePlayer”, true).
```

3. The variable type **AnimatorInfo** can be used to provide information about a specific animation. **<u>FALSE (the correct type is AnimatorStateInfo)</u>**
4. By default, within an Animation Controller, a Boolean parameter is set to false. **<u>TRUE</u>**
5. If the **Animator** linked to this script is in the state **FOLLOW_PLAYER**, the following code will display the message **We are in the FOLLOW_PLAYER Mode**. **<u>TRUE</u>**

```
private Animator anim;
private AnimatorStateInfo info;
anim = GetComponent<Animator>();
info = anim.GetCurrentAnimatorStateInfo(0);
if  (info.IsName("FOLLOW_PLAYER"))  print  (“We  are  in  the  FOLLOW_PLAYER
Mode”);
```

6. When using finite state machines, it is possible to have two states active at the same time. **<u>FALSE</u>**
7. In Mecanim, for a transition to occur between two states, a condition always needs to be defined. **<u>FALSE (can also occur after an animation us complete).</u>**
8. The following code will stop the movement of a **NavMeshAgemt**. **<u>TRUE</u>**

```
GetComponent<NavMeshAgent>().Stop();
```

9. The following code, provided that the script is linked to an object that includes a **NavMeshAgent**, will display the distance between the NPC and its destination. **<u>TRUE</u>**

```
print (GetComponent<NavMeshAgent>().remainingDistance);
```

10. Complete the following code so that we can test if the **Animator** is transitioning and that the next state is **GoBacktoStart**.

```
if(anim.IsInTransition(0)                                              &&
anim.GetNextAnimatorStateInfo(0).IsName("GoBackToStart"))
```