Quiz : Introduction à la Programmation en C# pour Unity

1. Quelle est la syntaxe correcte pour déclarer une variable de type entier en C# ?

a) `int = score 100;`

b) `int score = 100;`

c) `score int = 100;`

d) `int score == 100;`

Réponse correcte : b) `int score = 100;`

---

2. Quel type de données utiliseriez-vous pour stocker une valeur comme `3.14` ?

a) `int`

b) `float`

c) `string`

d) `bool`

Réponse correcte : b) `float`

---

3. Quelle est la sortie suivante pour le code suivant ?

```
int score = 60;

if (score >= 50)
{
    Debug.Log("Passé");
}
else
{
```

```
    Debug.Log("Échoué");
}
```

a) `Échoué`

b) `Passé`

c) `0`

d) Aucune sortie

Réponse correcte : b) `Passé`

---

4. Quelle est la fonction de la boucle `for` en C# ?

a) Elle répète une action un nombre spécifique de fois.

b) Elle répète une action jusqu'à ce qu'une condition soit vraie.

c) Elle répète une action indéfiniment.

d) Elle répète une action pour chaque élément d'un tableau ou d'une liste.

Réponse correcte : a) Elle répète une action un nombre spécifique de fois.

---

5. Que fait le code suivant dans une boucle `for` ?

```
for (int i = 0; i < 3; i++)
{
    Debug.Log(i);
}
```

a) Affiche les valeurs `0, 1, 2`

b) Affiche `3`

c) Affiche les valeurs `0, 1, 2, 3`

d) La boucle ne s'exécute pas

Réponse correcte : a) Affiche les valeurs `0, 1, 2`

---

6. Quel est le type de données utilisé pour représenter des valeurs vraies ou fausses en C# ?

a) `string`

b) `int`

c) `bool`

d) `float`

Réponse correcte : c) `bool`

---

7. Dans quel cas utiliseriez-vous une boucle `while` plutôt qu'une boucle `for` ?

a) Lorsque vous voulez exécuter une action un nombre fixe de fois.

b) Lorsque vous voulez exécuter une action tant qu'une condition reste vraie, sans savoir à l'avance combien de fois cela se produira.

c) Lorsque vous voulez itérer sur les éléments d'une liste.

d) Lorsque vous avez besoin de connaître l'indice de chaque élément.

Réponse correcte : b) Lorsque vous voulez exécuter une action tant qu'une condition reste vraie, sans savoir à l'avance combien de fois cela se produira.

---

8. Qu'affiche le code suivant dans la console Unity ?

```
string[] fruits = { "Pomme", "Banane", "Orange" };
foreach (string fruit in fruits)
{
    Debug.Log(fruit);
}
```

a) `Pomme, Banane, Orange`

b) `Pomme`

c) `Orange, Banane, Pomme`

d) `Aucun fruit`

Réponse correcte : a) `Pomme, Banane, Orange`

---

9. Comment incrémenter la valeur de `score` de 1 dans un script C# ?

a) `score = score + 1;`

b) `score++;`

c) `score + 1 = score;`

d) Les deux réponses a) et b)

Réponse correcte : d) Les deux réponses a) et b)

---

10. Si le score est égal à 100, quel message sera affiché avec ce code ?

```
int score = 100;

if (score >= 50)
{
    Debug.Log("Gagné !");
}
else
{
    Debug.Log("Perdu !");
}
```

a) `Perdu !`

b) `Gagné !`

c) `Aucun message`

d) `Erreur de syntaxe`

Réponse correcte : b) `Gagné !`

---

11. Que fait la déclaration `Vector3 playerPosition = new Vector3(0, 1, 0);` en C# dans Unity ?

a) Crée un objet de type `Vector3` avec des coordonnées `(x=0, y=0, z=0)`.

b) Crée un objet de type `Vector3` pour définir la position du joueur dans l'espace 2D.

c) Crée un objet `Vector3` pour définir la position du joueur dans l'espace 3D avec `x=0, y=1, z=0`.

d) Crée un objet `Vector3` sans valeurs initiales.

Réponse correcte : c) Crée un objet `Vector3` pour définir la position du joueur dans l'espace 3D avec `x=0, y=1, z=0`.

---

12. Quel est le rôle de l’instruction `else if` en C# ?

a) Elle permet d'exécuter un autre bloc de code si la première condition est fausse.

b) Elle permet de tester plusieurs conditions et d'exécuter un bloc de code lorsque l'une d'entre elles est vraie.

c) Elle termine le programme si la condition est vraie.

d) Elle est équivalente à `else`.

Réponse correcte : b) Elle permet de tester plusieurs conditions et d'exécuter un bloc de code lorsque l'une d'entre elles est vraie.

Conclusion

Ce quiz couvre les concepts de base de la programmation en C# pour Unity. En répondant à ces questions, vous avez pu tester votre compréhension des **variables**, des **types de données**, des **conditions**, des **boucles**, et de l’utilisation des **types spécifiques de Unity** comme `Vector3`. Ces concepts sont essentiels pour créer des comportements interactifs dans vos jeux Unity.