

1 Manipulation de tableau

Question 1 Soit les instructions suivantes :

```
1 int [][] a = new int[4][5];
```

Quelle est le type de cette expression `a[0]` ?

- (a) `int [][]`
 - (b) `int []`
 - (c) `int`
-

Question 2 Soit les instructions suivantes :

```
1 int [][] a = new int[4][];
```

Quelle est la valeur de `a[0][0]` ?

- (a) `null`
 - (b) **Erreur de compilation**
 - (c) `0`
 - (d) `NullPointerException`
-

Question 3 Soit les instructions suivantes :

```
1 int [][] a = new int[4][4];  
2 a[0][1] = 9;  
3 int [][] b = a;
```

Quelle est la valeur de `b[0][1]` ?

- (a) `IndexOutOfBoundsException`
 - (b) `9`
 - (c) `0`
-

Question 4 Soit le code suivant :

```
1 int [][] a = new int[4][4];  
2 a[0][1] = 9;  
3 int [][] b = a;  
4 b[1][0] = 9;
```

Quelle est la valeur de `a[1][0]` ?

- (a) `IndexOutOfBoundsException`
 - (b) `9`
 - (c) `0`
-

Question 5 Soit les deux tableaux suivants :

```
1 int [] a = new int[0];  
2 int [] b = new int[0];
```

En quoi est évalué l'expression suivante : `a == b`

- (a) `true`
 - (b) `false`
-

Question 6 Soit les instructions suivantes :

```
1 int [] a = new int[0];  
2 int [][] b = {a};  
3 int [][] c = {a};
```

En quoi est évalué l'expression suivante : `b[0] == c[0]`

- (a) `true`
 - (b) `false`
-

2 Manipulation des entiers

Question 1 Soit les instructions suivantes :

```
1 int a = 1;  
2 int b = 2;  
3 int c = a + b;
```

Quelle est la valeur stockée dans `c`

- (a) `0`
 - (b) `3`
 - (c) **Erreur de compilation**
-

Question 2 Soit les instructions suivantes :

```
1 int a = 1;  
2 int b = 2;  
3 byte c = a + b;
```

Quelle est la valeur stockée dans `c`

- (a) `0`
 - (b) `3`
 - (c) **Erreur de compilation**
-

Question 3 Soit la variable suivante :

```
1 byte a = 15;
```

Quelle est le type de l'expression suivante : `a >> 3`

- (a) `int`
 - (b) `byte`
 - (c) `long`
-

Question 4 Soit la variable suivante :

```
1 int a = -1;
```

L'expression `a >> 3` s'évalue à :

- (a) -1
- (b) 0
- (c) **Erreur de compilation**
- (d) **Une autre valeur, calculer la**

Question 5 Soit la variable suivante :

```
1 int a = 1;
```

L'expression `a >> 3` s'évalue à :

- (a) 1
- (b) 0
- (c) **Erreur de compilation**
- (d) **Une autre valeur, calculer la**

Question 6 Soit la variable suivante :

```
1 byte a = (byte) 0b11_11_11_11;
```

L'expression `(byte)(a >> 3)` s'évalue à :

- (a) 0b11_11_11_11
- (b) 0b00_01_11_11
- (c) 0b00_00_00_00

Question 7 Convertissez les valeurs suivantes en hexadécimal. Quelle est la taille de l'emplacement de chaque variable en mémoire ?

```
1 int a = 0;
2 byte b = -1;
3 short c = 128;
4 long d = 65535;
```

Question 8 Convertissez les valeurs suivantes en binaire.

```
1 int a = 0;
2 byte b = -1;
3 short c = 128;
4 long d = 65535;
```

Question 9 Soit le code suivant :

```
1 int a = 0xFF_00_FF_00;
2 int b = 0x00_FF_00_FF;
```

Quelle est la valeur de l'expression : `a & b` ?

- (a) 0x00_00_00_00
- (b) 0xFF_00_FF_00
- (c) 0xFF_FF_FF_FF
- (d) 0x00_FF_00_FF

Question 10 Soit le code suivant :

```
1 int a = 0xFF_00_FF_00;
2 int b = 0x00_FF_00_FF;
```

Quelle est la valeur de l'expression : `a | b` ?

- (a) 0x00_00_00_00
- (b) 0xFF_00_FF_00
- (c) 0xFF_FF_FF_FF
- (d) 0x00_FF_00_FF

3 Solutions

- 1.1 : b
- 1.2 : d
- 1.3 : b
- 1.4 : b
- 1.5 : b
- 1.6 : a
- 2.1 : b
- 2.2 : c
- 2.3 : a
- 2.4 : a
- 2.5 : b
- 2.6 : a
- 2.7 :
 - a = 0x00_00_00_00 (32-bit)
 - b = 0xFF (8-bit)
 - c = 0x00_80 (16-bit)
 - d = 0x00_00_00_00_00_00_FF_FF (64-bit)
- 2.8 :
 - a = 0b00000000_00000000_00000000_00000000
 - b = 0b11111111
 - c = 0b00000000_10000000
 - d = 0b0..._00000000_11111111_11111111
- 2.9 : a
- 2.10 : c