Algorithmes et Pensée Computationnelle

Fonctions, gestion de la mémoire et des exceptions

Le but de cette séance est d'approfondir vos connaissances en programmation. Au terme de cette séance, l'étudiant sera capable de :

- Utiliser des librairies contenant des fonctions prédéfinies,
- définir une fonction et l'utiliser dans un programme,
- connaître quelle est la portée d'une variable,
- comprendre comment fonctionne une pile d'exécution (call stack),
- gérer des exceptions.

1 Variables et Fonctions

Question 1: (*5 minutes*) **Les fonctions (fonctions basiques) (Java ou Python))**
Définissez une fonction nommée **ping()** qui, lorsqu'elle est appelée, affiche "pong".

Conseil

Référez vous aux diapositives du cours pour la création et l'appel des fonctions.

Solution

Python :

```python
def ping() :
    print("pong")

ping()
```

Java :

```java
public class Main {
    static void Ping(){
        System.out.println("Pong");
    }
    public static void main(String[] args) {
        Ping();
    }
}
```

Question 2: (*5 minutes*) **Les Fonctions (Fonction multiplication) (Java ou Python)**
Définissez une fonction nommée **multiplicateur()** qui prend deux arguments *multiple1* et *multiple2*, les multiplie et retourne le résultat. Stockez le résultat de **multiplicateur(2,3)** dans une variable *resultat* et affichez la.

Conseil

- Référez vous aux diapositives du cours pour la création et l'appel des fonctions.
- Pour retourner une valeur au lieu de l'afficher, utilisez le mot-clé **return** (pour Python et Java).

Solution

Python :

```python
def multiplicateur(multiple1, multiple2):
    return multiple1 * multiple2


if __name__ == "__main__":
    resultat = multiplicateur(2, 3)
```

Java :

```java
public class Main {
    public static int multiplicateur(int multiple1, int multiple2){
        return multiple1*multiple2;
    }

    public static void main(String[] args) {
        int resultat = multiplicateur(1, 2);
    }
}
```

Question 3: (*5 minutes*) **Les Fonctions (Fonctions Aire et Périmètre) (Java ou Python)**
Définissez deux fonctions nommées **aire()** et **perimètre()** qui prennent un argument (**rayon**) et renvoient respectivement l'aire et le périmètre d'un cercle. Stockez les résultats dans des variables **aire** et **perimetre** et affichez le contenu de ces variables.

Conseil

- Référez vous au cours pour la création et l'appel des fonctions.
- Pour retourner une valeur au lieu de l'imprimer, utilisez le mot clé `return` (pour Python et Java).
- Pour rappel, le périmètre d'un cercle s'obtient en utilisant la formule $P = 2 * \pi * r$ et l'aire s'obtient en utilisant la formule $A = r^2 * \pi$.

Solution

Python :

```python
import math

def aire(rayon):
  return (rayon**2)*math.pi

def perimetre(rayon):
  return 2*math.pi*rayon

if __name__ == '__main__':
    rayon = 10
    aire = aire(rayon)
    perimetre = perimetre(rayon)
    print("L'aire d'un cercle de rayon {} est égale à {}".format(rayon, aire))
    print("Le périmètre d'un cercle de rayon {} est égal à {}".format(rayon, perimetre))
```

Java :

```java
public class Main {

    static double aire(int rayon){
        return Math.pow(rayon, 2)*Math.PI;
    }

    static double perimetre(int rayon){
        return 2*Math.PI*rayon;
    }

    public static void main(String[] args) {
        int rayon = 5;
        double aire = aire(rayon);
        double perimetre = perimetre(rayon);
        System.out.println("L'aire d'un cercle de rayon "+rayon+" est égale à "+aire);
        System.out.println("Le périmètre d'un cercle de rayon "+rayon+" est égal à "+perimetre);
    }
}
```

Question 4: (*5 minutes*) **Portée des variables** (Python)
Qu'affiche le programme suivant ?

```python
x = 2

def fonction():
    x = 3

    def fonction2():
        global x
        print("x=" + str(x))

    fonction2()
    print("x=" + str(x))

print(fonction())
```

Conseil

- Le mot-clé `global` permet d'accéder aux variables globales (définies à l'extérieur de la fonction).
- Soyez attentif au type des éléments retournés par chacune des fonctions.

Solution

x=2
x=3
None

2 Gestion des exceptions

Question 5: (*5 minutes*) Qu'affiche le programme suivant ? **Types d'erreurs (Python)**

```
value1 = "Algorithms"
value2 = 4

try:
    size = len(value1)
    result = size/value2
    print(f"Le résultat de la division est: {result}")

except Exception as error:
    print("On ne peut pas effectuer l'opération")

try:
    result = value1/2
    print(f"Le résultat de la division est: {result}")

except TypeError as error:
    print("On ne peut pas diviser une chaîne de caractères")
```

Conseil

Utilisée sur une chaîne de caractères, la fonction `len()` renvoie le nombre de caractères de la chaîne.

Solution

Le résultat de la division est: 2.5
On ne peut pas diviser une chaîne de caractères.

Question 6: (*5 minutes*) Qu'affiche le programme suivant ? **Types d'erreurs (Python)**

```
value1 = 4
value2 = ""

try:
    count = len(value2)
    result= value1/count
except ZeroDivisionError as error:
    print("Nous ne pouvons pas diviser un nombre par 0")
```

Conseil

La chaîne de caractères vide est représentée par ""

Solution

Nous ne pouvons pas diviser un nombre par 0.

Question 7: (*5 minutes*) Qu'affiche le programme suivant ? **Types d'erreurs (Python)**

```
value1 = "Algorithms"
value2 = 4


try:
    decimal = float(value2)
except ValueError as error:
    print("Nous ne pouvons pas convertir un entier en décimal")

finally:
    try:
        value2 = int(value1)
    except ValueError as error:
        print("Nous ne pouvons pas convertir une chaîne de caractères en nombre")
```

Conseil

— La fonction `float()` permet de convertir une variable en nombre à décimal.
— La fonction `int()` permet de convertir une variable en nombre entier.

Solution

Nous ne pouvons pas convertir une chaîne de caractères.

Question 8: (*5 minutes*) **Gestion d'erreurs (Java)** Le programme suivant est incorrect. Que devez-vous modifier pour qu'il fonctionne correctement ?

```
public class Question8 {

    public static void division(int a, int b) throws ArithmeticException{
        if(b==0){
            throw new ArithmeticException();
        }else{
            float result = a/b;
            System.out.println("Le résultat de la division de " + a + "/" + b + "= " + result);
        }
}
    public static void main(String args[]){
        int value1 = 2;
        int value2 = 4;
        try {
            division(value1,value2);
            value2 = 0;
            division(value1,value2);
        }catch(IndexOutOfBoundsException err){
            System.out.println("Nous ne pouvons pas effectuer une division par 0");
        }
    }
}
```

Conseil

Référez-vous à la diapositive 26 du cours de cette semaine.

Solution

À l'intérieur de la fonction **division**, lorsque b est égal à 0, le programme lève une exception de type **ArithmeticException**. Mais dans le `main`, l'erreur qui est interceptée est **IndexOutOfBoundsException**. Pour corriger cette erreur, une des solutions est de remplacer **IndexOutOfBoundsException** par **ArithmeticException**.