Algorithmes et Pensée Computationnelle

Programmation de base

Le but de cette séance est d'aborder des notions de base en programmation. Au terme de cette séance, l'étudiant sera capable de :

- Définir une variable, définir son type et sa valeur.
- Définir une fonction et comprendre son rôle.
- Utiliser des notions d'algèbre booléenne.
- Comprendre la notion d'entrée/sortie.

Les langages qui seront utilisés pour cette séance sont Java et Python. Assurez-vous d'avoir bien installé Intellij. Si vous rencontrez des difficultés, n'hésitez pas à vous référer au guide suivant : tutoriel d'installation des outils et prise en main de l'environnement de travail.

1 Représentation de nombres entiers

Question 1: (**0** 5 minutes) **Entiers non signés** Sur 8 bits convertir $113_{(10)}$ en base binaire.

© Conseil

Faire un tableau comme présenté dans la diapositive 9 du cours de la semaine 3. Essayer de décomposer le nombre en une somme de puissances de 2.

>_ Solution

Méthode 1 : Utiliser la division par 2 comme vu dans la première séance d'exercices

Méthode 2 : Utiliser les puissances de 2 pour décomposer le nombre (vu également dans la première séance d'exercices).

$$113 = 64 + 32 + 16 + 1 = 1 * 2^6 + 1 * 2^5 + 1 * 2^4 + 1 * 2^0$$

2^7	2^{6}	2^{5}	2^4	2^3	2^2	2^1	2^{0}
128	64	32	16	8	4	2	1
0	1	1	1	0	0	0	1

On obtient donc $01110001_{(2)}$

Question 2: (*5 minutes*) **Entiers signés négatifs**

En utilisant le résultat de la question précédente, convertir sur 8 bits -113₍₁₀₎ en base binaire.

Conseil

- Il faut prendre la représentation sur 7 bits d'un nombre entier non signé.
- On rajoute un 8ème bit qui sera le signe.
- Pour cet exercice, reprendre l'expression non signée de la question précédente (question
 - 1 Entiers non signés) et changer le premier bit en conséquence.
- Le premier bit vaut 0 pour un nombre positif et 1 pour un nombre négatif.

>_ Solution

$$113_{(10)} = 01110001_{(2)}$$

En changeant le premier bit à 1 pour faire passer le nombre en nombre négatif on obtient :

$$-113_{(10)} = 11110001_{(2)}$$

Question 3: (5 minutes) Complément à 1

Ecrire le complément à 1 de -113₍₁₀₎.

Qu'elle est la différence entre cette méthode et la précédente?

© Conseil

- En programmation l'opposé d'une variable est **not**(la variable). Ici, le même principe s'applique. L'opposé de 0 en binaire est 1.
- Pour étudier la différence, il faut regarder les différentes manières d'exprimer -0 en binaire (se référer à la diapositive 10 du cours de la semaine 3).

>_ Solution

$$113_{(10)} = 01110001_{(2)}$$

0	1	1	1	0	0	0	1
not							
1	0	0	0	1	1	1	0

avec cette méthode, $-113_{(10)} = 10001110_{(2)}$

La portée de cette méthode ne change pas par rapport à la précédente. Par contre, l'expression de -0 sera différente.

Signé :
$$-0_{(10)} = 10000000_{(2)}$$

Complément à 1 :
$$-0_{(10)} = 11111111_{(2)}$$

Les deux ont le même intervalle : $[-127_{(10)}, +127_{(10)}]$

Question 4: (5 minutes) Complément à 2

Quel est le complément à 2 de -113(10) et quelle est l'utilité de cette représentation?

Conseil

Exemple du cours avec 87₍₁₀₎

$$87_{(10)} = 01010111_{(2)}$$

a	0	1	0	1	0	1	1	1
b	1	0	1	0	1	0	0	0
С	1	0	1	0	1	0	0	1

a : convertir le nombre en binaire

b: inverser tous les bits

c : rajouter 1 au nombre pour obtenir le complément à 2

On obtient donc $-87_{(10)} = 10101001_{(2)}$

>_ Solution

Ici, il suffit donc d'ajouter 1 au complément à 1 de -113₍₁₀₎

On obtient donc 10001111₍₂₎

Concernant la portée, elle est changée, étant donné qu'il n'existe plus qu'une seule représentation possible pour $-0_{(10)}$.

Le nouvel intervalle est donc $[-128_{(10)},+127_{(10)}]$

Question 5: (**1** *10 minutes*) **Floating point**

Voici la représentation en binaire d'un nombre à virgule flottante :

signe	exposant	mantisse
0	10110101	010000010000000000000000000000000000000

Que vaut cette représentation en base 10? Utiliser la représentation des floating point (avec un biais de 127) Arrondir les résultats intermédiaires et la valeur finale au 3ème chiffre significatif après la virgule.

Conseil

- Se référer à la diapositive 14 du cours de la semaine 3 pour plus de détails.
- Poser chaque calcul, et ensuite tout fusionner avec la formule.
- L'exposant et la mantisse sont des entiers positifs, donc non signés.

>_ Solution

- Signe du nombre : $(-1)^{signe} = (-1)^0 = 1$
- Exposant en base $10:10110101_{(2)} = 181_{(10)}$
- Pour obtenir l'exposant, il faut encore lui appliquer le biais, il faut soustraire 127 à notre résultat. Ici on aura $2^{(181-127)} = 2^{54}$

Question 6: (5 minutes) Conversion d'un nombre binaire (au format complément à 2) en base 10 Optionnel

Soit le nombre binaire suivant exprimé sur 8 bits au format complément à 2 : $10010011_{(2)}$. Convertissez ce nombre en base 10.

Conseil

- Utilisez le même tableau que dans la question 7.
- On applique le processus inverse à celui utilisé dans la question 7.

>_ Solution

Il faut appliquer le processus de la question 4 (complément à 2), mais de façon inversée. Cela permet d'obtenir la valeur positive en binaire du nombre que l'on cherche. Ensuite, il faut convertir cette valeur en base 10 et puis multiplier par -1.

ſ	a	1	0	0	1	0	0	1	1
Γ	b	1	0	0	1	0	0	1	0
ſ	С	0	1	1	0	1	1	0	1

a : écrire le nombre en binaire

b: soustraire 1

c: inverser tous les bits

On obtient donc $01101101_{(2)} = 109_{(10)}$

Pour finir, on obtient - $109_{(10)}$ après multiplication par -1.

2 Typage

Le but de cette partie des travaux pratiques est de comprendre les notions de typage statique et dynamique, ainsi que leur impact sur l'exécution et la gestion des erreurs.

Question 7: (O *3 minutes*) Les types de variables

Quelles sont les principaux types de variable (donnez en 3), comment les déclareriez vous en Python et en Java?

Conseil

Se référer aux diapositives du cours ou à la documentation officielle de votre langage de programmation préféré.

>_ Solution

Les principaux types de variables sont les suivants : Integer (int), Float, String (str), Boolean (bool).

```
En Python:
```

```
    i = 0
    f = 3.14
    s = "Hello World"
    b = True
    En Java :
    int i = 0;
    float f = 3.14f;
    String s = "Hello World";
```

boolean b = true;

Il existe d'autres types de variable, comme par exemple les double.

Question 8: (5 minutes) **Typage statique et dynamique**

Parmi ces différents programmes, lesquels pourront être exécutés sans problème et lesquels lèveront des erreurs? Dans le cas où ils génèreraient des erreurs, expliquez la raison de ces erreurs.

Java:

```
1
     // Programme 1
 2
     int variable_1 = 0;
     int variable_2 = 3;
     variable_2 = variable_1;
 4
     // Programme 2
 6
 7
     var variable_1 = 0;
 8
     var variable_2 = "Hello";
     variable_2 = variable_1;
10
     // Programme 3
11
     int variable_1 = 0;
12.
     String variable_2 = "Hello";
13
     variable_2 = variable_1;
14
15
    // Programme 4
     var variable_1 = 0;
17
     variable_1 = 3.14;
18
19
20
     // Programme 5
21
     var variable_1 = 0;
     String variable_2 = "Hello";
     String variable_3 = "World";
```

```
variable_2 = variable_3;
25
26
    // Programme 6
    int variable_1 = 0;
    variable_1 = 3.14;
     Python:
    # Programme 7
 1
 2
     variable_1 = 0
 3
     variable_2 = "Hello"
 4
     variable_2 = variable_1
 5
 6
    # Programme 8
     variable_1 = 0
 7
     variable_1 = 3.14
```

Conseil

En Java, le type est statique, ce qui signifie qu'à partir du moment où vous créez une variable, un type va lui être attribué (par vous ou par le code en lui-même par déduction). Ce type ne pourra pas être changé, et si vous tentez de le faire (en lui attribuant une valeur d'un autre type par exemple), une erreur sera levée.

En Python, le typage est dynamique, il est donc tout à fait possible de changer le type d'une variable sans déclencher d'erreur.

A Erreurs fréquentes

Au cas où vous exécuteriez les programmes 2, 4 et 5 et rencontriez l'erreur suivante : Exception in thread "main" java.lang.UnsupportedClassVersionError: ... has been compiled by a more recent version of the Java Runtime (class file version 54.0), this version of the Java Runtime only recognizes class file versions up to 52.0, suivez les étapes ci-dessous :

- Mettez à jour votre JDK en téléchargeant la version adéquate via le lien suivant : https: //www.oracle.com/java/technologies/javase-jdk15-downloads. html.
- 2. Une fois votre JDK installé, redémarrez IntelliJ
- 3. Dans la barre de menus, cliquez sur **Run** > **Edit Configurations**.
- 4. Dans la fenêtre qui apparaitra (capture ci-dessous), sélectionnez la version du JRE la plus récente.

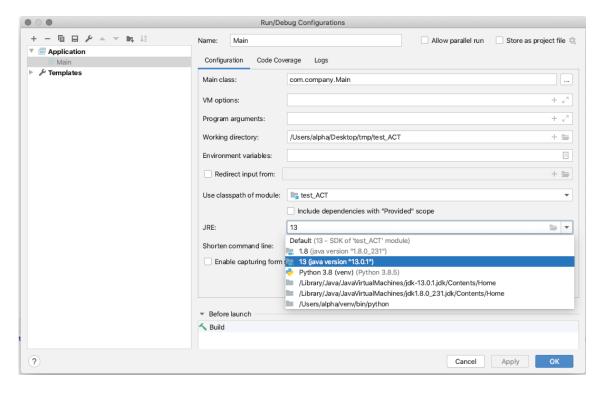
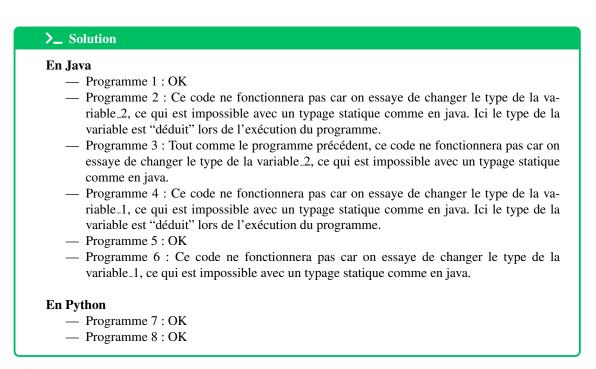


FIGURE 1 – Fenêtre de configuration de l'environnement de développement Java.



Question 9: (O *5 minutes*) **Détection d'erreurs**

Voici deux programmes, l'un en Java et l'autre en Python. Chaque programme comporte une fonction nommée raise_error() levant une exception de type TypeError. Dans les deux cas, cette fonction ne sera pas appelée lors de l'exécution (la condition est remplie d'office). Pourtant, l'un de ces deux codes lèvera une erreur, et l'autre sera exécuté sans problème. Quel code lèvera l'erreur et lequel sera exécuté? Expliquez pourquoi.

En Java:

```
public class Main {
 2
 3
       static void raise_error(){
 4
         int variable_1 = 0;
 5
          variable_1 = "Bonjour";
 6
 7
 8
       public static void main(String[] args) {
 9
         if (true) {
            System.out.print("OK");
10
11
12
         else {
13
            raise_error();
14
15
16
     En Python:
 1
     def raise_error():
 2
       print("3" + 5)
 3
 5
       print("OK")
 6
     else:
       raise_error()
```

Conseil

Pensez à vérifier le type des variables. Faites attention à la différence principale entre le typage des variables en Java et le typage en Python.

>_ Solution

Le premier code va lever une erreur de type **Type error**, même si la fonction n'est jamais appelée. En Java, le typage est statique, et de ce fait, toutes les erreurs seront détectées avant exécution du programme.

Le deuxième code en revanche ne va pas lever d'erreur. En python, le typage est dynamique et de ce fait, l'erreur ne sera détectée qu'au moment ou le programme sera exécuté. Si on change le True en False, la fonction provoquant l'erreur va être appelée, et dans ce cas ci, l'erreur sera mise en évidence.

3 Bases en programmation

Le but de cette section est d'écrire vos premières lignes de code. Les notions abordées concerneront les variables, les fonctions, et les interactions avec l'utilisateur (input/output). Vous pouvez les écrire en Java ou en Python.

Question 10: (5 minutes) Output (Java ou Python)

Créez une variable *nom* (str) contenant votre nom, et une autre *prenom* (str) contenant votre prénom puis affichez : "Bonjour, *prenom nom*".

Conseil

Utilisez la fonction print() de Python et System.out.println() de Java.

Python: 1 prenom = "John" 2 nom = "Doe" 3 print("Bonjour," + prenom + "" + nom) Java: 1 String prenom = "John"; 2 String nom = "Doe"; 3 System.out.println("Bonjour," + prenom + "" + nom);

Question 11: (Input (Java ou Python)

En vous référant à l'exercice précédent (**Output** (**Java ou Python**)), demandez à l'utilisateur d'entrer son nom et son prénom via la fonction input() au lieu d'initialiser vous-même les variables.

Conseil

Utilisez la fonction input() en Python, la classe Scanner() en Java (n'oubliez pas d'ajouter import java.util.Scanner;) tout au début de votre code.

```
>_ Solution
   Python:
   prenom = input("Quel est votre prénom ?")
   nom = input("Quel est votre nom ?")
   print("Bonjour, " + prenom + " " + nom)
    Java:
   Scanner my_scanner = new Scanner(System.in);
3
   System.out.println("Entrez votre Prenom : ");
4
   String prenom = my_scanner.nextLine();
   System.out.println("Entrez votre Nom : ");
6
7
   String nom = my_scanner.nextLine();
   System.out.println("Bonjour," + prenom + ""+ nom);
```

Question 12: (*5 minutes*) **Format d'impression (Python uniquement)**

Créez et assignez des valeurs à 2 variables prenom (str) et age (int), puis affichez : "Je m'appelle prenom et

j'ai age ans". Gérez le format de l'impression via l'opérateur +, puis en utilisant la fonction format().

Conseil

N'hésitez pas à consulter ce lien pour plus de détails concernant l'utilisation de la fonction format() :

https://docs.python.org/fr/3.5/library/stdtypes.html#str.format

>_ Solution

- 1 prenom = input("Quel est votre prénom ?")
- 2 age = input("Quel est votre age ?")
- 3 print("Bonjour, je m'appelle" + prenom + " et j'ai" + age + " ans.")
- 4 print("Bonjour, je m'appelle {0} et j'ai {1} ans.".format(prenom,age))

Question 13: (**Q** *3 minutes*) **Type** (**Python uniquement**)

Déclarez deux variables nom (String) et age (int), puis affichez le type de chacune de ces deux variables.

Conseil

Vous pouvez contrôler le type de vos variables via la fonction type().

>_ Solution

- 1 **nom = "John"**
- $2 \quad \text{age} = 23$
- 3 print(type(nom))
- 4 print(type(age))

Question 14: (5 minutes) Conversion des variables (Type casting) (Java ou Python)

Il est possible de convertir une variable d'un certain type vers un autre type. Il est par exemple possible de changer un int en float ou un float en int. Déclarez une variable *nombre_entier* de type int, puis une autre variable *nombre_decimal* de type float. Affichez *nombre_entier* en le convertissant en float et *nombre_decimal* en le convertissant en int.

Conseil

Utilisez la fonction int(float) et float(int) en Python / Utilisez (int) float et (float) int en Java.

>_ Solution

Python:

- $1 \quad nombre_entier = 0$
- $2 \quad nombre_decimal = 3.14$
- 3 print(float(nombre_entier))
- 4 print(int(nombre_decimal))

Java:

- 1 int nombre_entier = 0;
- 2 **float** nombre_decimal = 3.14f:
- 3 System.out.println((float) nombre_entier);
- 4 System.out.println((int) nombre_decimal);

Question 15: (5 minutes) Conversion des variables (Type casting) (Java ou Python) Optionnel Qu'afficheront les programmes suivants?

Python:

- 1 nombre_entier = 3
- 2 nombre_decimal = float(nombre_entier)
- 3 print(nombre_entier)
- 4 print(nombre_decimal)

Java:

- 1 float nombre_decimal = 3.14f;
- 2 int nombre_entier = (int) nombre_decimal;
- 3 System.out.println(nombre_entier);
- 4 System.out.println(nombre_decimal);



Conseil

Attention, ces fonctions ne changent pas le type des variables, elles ne font que les convertir.

>_ Solution

Python:

3

3.0

Java:

3

3.14

Question 16: (3 minutes) Calculs (multiplication) (Java ou Python)

Créez 2 variables *facteur_1* (= 11) et *facteur_2* (= 3). Multipliez la première variable par la deuxième et stockez le résultat dans une nouvelle variable *produit*. Vous pouvez afficher les différentes variables pour voir leurs valeurs. Vous pouvez répéter l'exercice avec l'addition et la soustraction.



Conseil

L'opérateur de multiplication est le *, celui d'addition est le + et celui de soustraction est le -.

>_ Solution

Python:

- 1 facteur_1 = 11
- 2 facteur 2 = 3
- 3 produit = facteur_1*facteur_2
- 4 **print**(facteur_1)
- 5 **print(facteur_2)**
- 6 **print(produit)**

Java:

- 1 int facteur_1 = 11;
- 2 int facteur 2 = 3;
- 3 int produit = facteur_1*facteur_2;
- $4 \quad \textbf{System.out.println(facteur_1);} \\$
- 5 System.out.println(facteur_2);
- 6 System.out.println(produit);

Question 17: (10 minutes) Calculs (division) Optionnel (Java ou Python)

Créez 2 variables *nb_bonbons* avec pour valeur 11 et *nb_personnes* avec pour valeur 3. Divisez la première variable par la deuxième et stockez le résultat dans une nouvelle variable *bonbons_personnes*. Pour finir, calculez le nombre de bonbons restants via l'opérateur % (modulo) et stockez le résultat dans une nouvelle variable *reste*. Vous pouvez afficher les différentes variables pour voir leurs valeurs.

Conseil

Attention, en Python il existe 2 opérateurs de division, / effectue une division classique, tandis que // effectue une division entière. En Java, si vous travaillez uniquement avec des int, / effectuera une division entière tandis que si vous travaillez avec au moins un float, / effectuera une division classique. Vous pouvez aussi formater le type du résultat lorsque vous créez une variable.

>_ Solution Python: 1 #1 2 $nb_bonbons = 11$ 3 $nb_personnes = 3$ bonbons_personnes = nb_bonbons // nb_personnes reste = nb_bonbons % nb_personnes print (nb_bonbons) 7 print(nb_personnes) 8 print(bonbons_personnes) print(reste) 10 11 $nb_{-}bonbons = 11$ 12 13 $nb_personnes = 3$ 14 bonbons_personnes = nb_bonbons / nb_personnes 15 print (nb bonbons) print(nb_personnes) print(bonbons_personnes) Java: // 1 2 int nb_bonbons = 11; 3 int nb_personnes = 3; int bonbons_personnes = nb_bonbons / nb_personnes; int reste = nb_bonbons % nb_personnes; System.out.println(nb_bonbons); 6 7 System.out.println(nb_personnes); 8 System.out.println(bonbons_personnes); 9 System.out.println(reste); 10 11 // 2 12 float nb_bonbons = 11; int nb_personnes = 3; float bonbons_personnes = nb_bonbons / nb_personnes; 14 15 System.out.println(nb_bonbons); 16 System.out.println(nb_personnes); System.out.println(bonbons_personnes);

Question 18: (5 minutes) Calculs (incrémentation / décrémentation) (Java ou Python)
Gardez vos variables de l'exercice précédent (Calculs (division) (Java ou Python)), augmentez la valeur de nb_bonbons de 1, et diminuez celle de nb_personnes de 1.

Conseil

Vous pouvez utiliser les opérateurs += et -= en Python, et les opérateurs ++ et - en Java.

```
>_ Solution
    Python:
    nb_bonbons = 11
2
    nb_personnes = 3
3
    nb_bonbons += 1
4
    nb_personnes -= 1
    bonbons_personnes = nb_bonbons // nb_personnes
    reste = nb_bonbons % nb_personnes
6
    print (nb_bonbons)
    print(nb_personnes)
    print(bonbons_personnes)
10
    print(reste)
    Java:
    int nb_bonbons = 11;
    int nb_personnes = 3;
2
3
    nb_bonbons ++;
    nb_personnes --;
5
    int bonbons_personnes = nb_bonbons / nb_personnes;
6
    int reste = nb_bonbons % nb_personnes;
    System.out.println(nb_bonbons);
8
    System.out.println(nb_personnes);
    System.out.println(bonbons_personnes);
    System.out.println(reste);
```

Question 19: (5 minutes) Manipulation des chaînes de caractères (indexation) (Java ou Python) Créez une variable mon_mot de type chaîne de caractères avec pour valeur "Hard But Cool!!". Créez ensuite une variable premiere contenant la première lettre de mon_mot en utilisant l'indexation. Créez ensuite une variable dernière contenant la dernière lettre de mon_mot en utilisant l'indexation. Affichez les résultats et voyez ce que vous obtenez.

Conseil

Pour Python, utilisez [], et pour Java, utilisez la fonction substring() ainsi que la fonction length() qui permet d'obtenir la taille d'un élément.

>_ Solution Python: mon_mot = "Hard But Cool !!" $premiere = mon_mot[0]$ 3 $derniere = mon_mot[-1]$ print(premiere) 4 print(derniere) Java: String mon_mot = "Hard But Cool !!"; String premiere = mon_mot.substring(0,1); 2 3 String derniere = $mon_mot.substring(mon_mot.length()-1)$; 4 System.out.println(premiere); System.out.println(derniere);

Question 20: (5 minutes) Manipulation des chaînes de caractères (indexation 2) (Java ou Python) Gardez votre variable, mon_mot et créez une variable lettre_5 contenant la cinquième lettre de mon_mot en utilisant l'indexation. Créez ensuite une variable lettre_9_13 contenant les lettres 9, 10, 11, 12, 13 de mon_mot. Afficher les résultats et voyez ce que vous obtenez.

Conseil

Attention, ici les espaces comptent comme des lettres!

Pour Python, utilisez [:], et pour Java, utilisez la fonction substring().

>_ Solution Python: 1 mon_mot = "Hard But Cool!!" 2 lettre_5 = mon_mot[4]

4 print(lettre_5)
5 print(lettre_10_13)

Java:

3

- 1 String mon_mot = "Hard But Cool !!";
- 2 String lettre_5 = mon_mot.substring(4,5);
- 3 String lettre_10_13 = mon_mot.substring(9,13);
- 4 System.out.println(lettre_5);
- 5 System.out.println(lettre_10_13);

 $lettre_10_13 = mon_mot[9:13]$

Question 21: (10 minutes) Manipulation des chaînes de caractères (Java ou Python) Optionnel Il est possible d'obtenir la longueur d'une chaîne de caractère (ou d'une liste ou d'un dictionnaire) en utilisant la fonction len(). Gardez votre variable mon_mot et créez une nouvelle variable nommée ln_mon_mot contenant le nombre de caractère de la variable mon_mot, puis une nouvelle variable moitie contenant la première moitié de la variable mon_mot (utilisez la variable que vous venez de créer). Affichez le résultat.

Conseil

La fonction présentée dans l'énoncé de la question n'est valable que pour python. L'équivalent pour Java est la fonction length().

>_ Solution

Python:

- 1 mon_mot = "Hard But Cool !!"
- $2 \quad \mathbf{ln}_{-}\mathbf{mon}_{-}\mathbf{mot} = \mathbf{len}(\mathbf{mon}_{-}\mathbf{mot})$
- 3 moitie_mon_mot = mon_mot[:ln_mon_mot//2]
- 4 print(ln_mon_mot)
- 5 **print**(moitie_mon_mot)

Java:

- 1 String mon_mot = "Hard But Cool !!";
- 2 int ln_mon_mot = mon_mot.length();
- 3 String moitie = mon_mot.substring(0,ln_mon_mot/2);
- 4 System.out.println(ln_mon_mot);
- 5 System.out.println(moitie);

Question 22: (5 minutes) Les Fonctions (Fonctions basiques) (Java ou Python))

Définissez une fonction nommée ping() qui, lorsqu'elle est appelée, affiche "pong". Appelez la plusieurs fois et observez le résultat.

Conseil

- Référez vous aux diapositives du cours pour la création et l'appel des fonctions.
- Vous pourriez utiliser une boucle for pour effectuer plusieurs appels à la fonction ping().

>_ Solution Python: def ping(): print("pong") 3 4 ping() ping() Java: public class Main { 2 static void Ping(){ 3 System.out.println("Pong"); 4 5 public static void main(String[] args) { 6 Ping(); 7 Ping(); 8 9

Question 23: (5 minutes) Les Fonctions (Fonction multiplication) (Java ou Python)

Définissez une fonction nommée multiplicateur() qui prend deux arguments multiple_1 et multiple_2, les multiplie et retourne le résultat. Stockez le résultat de multiplicateur(2,3) dans une variable resultat et affichez la.

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).

>_ Solution

```
Python:
     def multiplicateur(a,b):
2
       return(a*b)
3
     print(multiplicateur(2,3))
5
     print(multiplicateur(4,5))
     Java:
     public class Main {
3
       static int Multiplicateur(int a, int b){
4
         return a*b;
5
6
7
       public static void main(String[] args) {
         System.out.println(Multiplicateur(2,3));
8
9
10
   }
```

4 Opérateurs et conditions Booléennes (Python uniquement)

Le principe d'une valeur booléenne est qu'elle ne puisse contenir que 2 valeurs possibles, soit True, soit False. Il est possible de les définir en leur associant une de ces valeurs d'emblée ou de les obtenir en effectuant une comparaison. Pour ce faire, il faut utiliser des opérateurs booléens. Voici les plus utilisés : == (est égal), ! = (n'est pas égal), < (est strictement plus petit), <= (est plus petit ou égal), > (est strictement plus grand), >= (est plus grand ou égal). Si la condition est satisfaite, on obtiendra True, si elle ne l'est pas, on obtiendra False. L'utilisation de l'opérateur not inversera le résultat.

Dans les exercices suivants, vous devrez anticiper la valeur que la console va vous donner (résultat du(des) print(s)).

Conseil

Utilisez les tables de vérité présentées à la diapositive 23 du cours.

Question 24: (5 *minutes*) Qu'affichera le programme suivant?

```
1 a = 3
2 b = 2
3 c = 6
4 d = c >= b
5 print(a==b)
6 print(a*b==c)
7 print(d)
8 print(a <= b)
9 print(not d)
```

>_ Solution

False True True

False

False

Question 25: (*5 minutes*) Qu'affichera le programme suivant?

```
1  a = 3
2  b = 2
3  c = 6
4  d = c >= b
5  print(a==b or d)
6  print(a==b and d)
7  print(a==b or a==c or False or d)
8  print(a < c and not (b == 2 and not d))</pre>
```

>_ Solution

True False

True

True

Question 26: (**O** *5 minutes*) Qu'affichera le programme suivant?

```
    a = True
    b = False
    c = True
    if a or (b and not c):
    print("output 1")
```

```
6 if b!= c:
7 print("output 2")
8 if a or (not b != (c and a)):
9 print("output 3")
10 if not a or (b==c and not b):
11 print("output 4")
```

>_ Solution

output 1 output 2 output 3

5 Conditions

Le but de cette section est de vous entraîner à la lecture de code, la compréhension des opérateurs booléens et au "case switching" à travers le branchement conditionnel.

Question 27: (*10 minutes*) **Branchement conditionnel en Java** Qu'affiche le code suivant?

```
1
      int numero_mois = 7;
 2
 3
      switch(numero_mois) {
 4
 5
        case 1:
 6
              System.out.println("Janvier");
 7
              break;
 8
        case 2:
 9
              System.out.println("Février");
10
11
        case 3:
              System.out.println("Mars");
12
13
              break;
14
        case 4:
15
              System.out.println("Avril");
16
              break;
        case 5:
17
18
              System.out.println("Mai");
19
              break:
20
        case 6:
21
              System.out.println("Juin");
22
              break;
23
        case 7:
24
              System.out.println("Juillet");
25
              numero_mois = 9:
26
27
              System.out.println("Aout");
28
        case 9:
29
              if (numero\_mois == 8){
30
               System.out.println("Septembre");
31
               break;
32
              else{
33
               System.out.println("Décembre");
34
35
               numero_mois = 13;
36
               break:
37
38
        case 10:
39
              System.out.println("Octobre");
40
41
        case 11:
              System.out.println("Novembre");
42
43
              break;
44
        case 12:
45
              System.out.println("Décembre");
46
              break:
47
        default:
48
              System.out.println("Ce n'est pas un mois.");
     }
49
```

Conseil

- break indique que l'on sort de l'accolade. Les cas suivants ne seront pas traités.
- L'absence de break indique que l'on va rentrer dans tous les cas suivants, jusqu'à enfin atteindre un break.
- Lorsque l'on pose case n où n est un nombre cela est équivalent au test n == numero_mois. Ce test est aussi valable si on cherche à comparer des chaînes de caractères (par exemple si numero_mois = "Juin", à ce moment là n sera aussi une chaîne de caractères).

>_ Solution

Juillet

Aout

Décembre

Explications:

- Comme la case 7 ne contient pas de break et modifie numero_mois, la lecture du code va continuer.
- On rentre dans le case 9, qui contient un break. Le numero_mois sera aussi modifié mais cela ne sera pas important car on sort de l'accolade et les cas suivants ne seront pas traités.

Question 28: (5 minutes) Conditions en Python

Qu'affiche le code suivant?

```
teacher = "Garbinato"
    assistant = "Diallo"
2
3
    lesson = "ACT"
    if ((teacher == "Diallo") or not(teacher == "Garbinato")) and (lesson == "ACT"):
       print("Alpha Diallo et Benoît Garbinato")
     elif ((teacher == "Michelet") and (lesson == "ACT")) or (teacher == assistant):
7
       print("Gaëtan Michelet")
    elif (teacher == "Yasser Haddad" or assistant == "Diallo") and (lesson == "ACT" and teacher == "Olivier"):
8
9
       print("Yasser Haddad")
10
    elif ((teacher == "Garbinato" or lesson == "INF") and assistant == "Diallo") or (lesson == "ACT" and teacher == "Diallo");
       print(teacher + ": Professeur du cours Algorithmique et pensée computationelle.")
11
12
       print("Benoît")
13
```

Conseil

- Il faut vérifier la condition de chaque cas de façon linéaire.
- Une fois une condition vérifiée, toutes celles d'après ne sont pas traitées.

>_ Solution

Garbinato: Professeur du cours Algorithmique et pensée computationelle.

Ci-dessous, le résultat des propositions booléennes des if et elif :

- 1. (False or False) and True = False
- 2. (False and True) or False = False
- 3. (False or True) and (True and False) = True and False = False
- 4. ((True or False) and True) or (True and False) = True

6 Exercices pour aller plus loin, facultatifs mais recommandés (Solutions en Python uniquement)

Ces deux exercices consistent à créer de petits jeux bien connus, et relativement rapides à implémenter.

Question 29: (20 minutes) Le juste prix Optionnel

Dans le programme suivant, nous vous donnons un nombre aléatoire entre 0 et 30 dans la variable *number*, écrivez un programme qui demande à l'utilisateur de deviner le nombre tiré au sort. L'utilisateur a 5 chances pour le trouver. S'il se trompe, donnez-lui un indice (le nombre qu'il a écrit est-il plus grand ou plus petit que celui qu'il cherche?). Vous pouvez vous amuser à modifier le nombre de chances ou le nombre de possibilités (par exemple 10 chances pour trouver un nombre entre 0 et 100).

- # Programme écrit en Python
- 2 from random import randint
- 3 number = randint(0,30)
- 5 #Votre code

```
>_ Solution
    # Programme écrit en Python
    from random import randint
 2.
 3
     number = randint(0, 30)
     x = int(input("Choisissez un nombre: "))
    if x == number:
 5
 6
     print("Yeeah!")
 7
     elif x < number:
 8
     print("Trop bas!")
 9
10
     print("Trop haut!")
11
12
     x = int(input("Choisissez un nombre: "))
    if x == number:
13
     print("Yeeah!")
15
    elif x < number:
16
     print("Trop bas!")
17
18
     print("Trop haut!")
19
20
    x = int(input("Choisissez un nombre: "))
21
    if x == number:
22
     print("Yeeah!")
23
    elif x < number:
24
     print("Trop bas!")
25
26
     print("Trop haut!")
27
28
     x = int(input("Choisissez un nombre: "))
29
    if x == number:
     print("Yeeah!")
30
31
    elif x < number:
32
     print("Trop bas!")
33
34
     print("Trop haut!")
35
36
    x = int(input("Choisissez un nombre: "))
37
    if x == number:
38
     print("Yeeah!")
    elif x < number:
40
     print("Trop bas!")
41
42
     print("Trop haut!")
     Le problème avec cette solution est le suivant : Si le joueur trouve la réponse, le jeu va continuer,
     une façon plus propre et correcte de coder ce jeu est d'utiliser une boucle (prochain chapitre).
    from random import randint
    number = randint(0, 30)
 3
    for i in range(5):
 4
       x = int(input("Choississez un nombre: "))
       if x==number:
 5
         print("Yeah!")
 6
         break
 8
       elif x<number:</pre>
         print("Trop petit!")
10
         print("Trop grand!")
11
     Ici le code est plus concis et permet de s'arrêter lorsque le joueur aura trouvé la bonne réponse.
```

Question 30: (20 minutes) Pierre, Feuille, Ciseaux Optionnel

Demandez à l'utilisateur d'entrer soit pierre, soit feuille, soit ciseaux. L'ordinateur choisira son coup au hasard (s'il choisi 1 ce sera pierre, si c'est 2 ce sera feuille et si c'est 3 ce sera ciseaux). Les règles sont les règles classiques, une manche gagnante.

```
    # Programme écrit en Python
    from random import randint
    number = randint(1,3)
    #Votre code
```

```
>_ Solution
     # Programme écrit en Python
    from random import randint
 2
 3
     number = randint(1,3)
 4
     if number == 1 :
 5
       ordi = "pierre"
 6
     elif number == 2 :
 7
       ordi = "feuille"
 8
     else:
 9
       ordi = "ciseaux"
10
     player = input("Choississez un signe (pierre, feuille, ciseaux) : ")
12
     print("ordi a choisi " + ordi)
13
14
15
     if player != "pierre" and player != "feuille" and player != "ciseaux" :
16
       print("symbole invalide")
17
18
       if ordi == "pierre":
19
         if player == "pierre":
           print("égalité")
20
21
         elif player == "feuille" :
22
           print("gagné")
23
         else:
24
           print("perdu")
25
       elif ordi == "feuille" :
26
         if player == "pierre" :
27
           print("perdu")
28
         elif player == "feuille" :
29
           print("égalité")
30
31
           print("gagné")
       else:
32
33
         if player == "pierre" :
34
            print("gagné")
         elif player == "feuille" :
35
36
           print("perdu")
37
         else:
38
            print("égalité")
     Vous pouvez également utiliser une boucle pour augmenter le nombre de manches.
```