TD1. Premiers programmes

Bases de la Programmation Impérative – Chapitre 1 : bases
version du 25 décembre 2023

L'objectif de ce premier TD est d'être capable d'écrire **et de comprendre** les premiers programmes Python que nous réaliserons en TP.

Attention : comme nous l'avons vu dans le premier cours, Python est un langage de haut niveau. Par conséquent, bien que les programmes que nous allons voir dans ce premier TD soient petits et simples, ils cachent beaucoup de choses. L'objectif ici n'est pas de rentrer dans toutes ces choses cachées, mais de comprendre le nécessaire pour aborder les premiers TP.

Exercice 1 : portée des variables

On considère le programme suivant :

```
1  #!/usr/bin/env python3
2
3  une_var_glob = 7
4
5  def f(un_param):
6      une_var_loc = 4
7      une_autre_var_loc = un_param + une_var_loc
8      print(une_autre_var_loc)
9
10 f(une_var_glob)
11 print(une_var_glob)
```

Question 1

Comme nous l'avons fait en cours (diapositive 21 ici), exécutez le programme une instruction Python à la fois en reportant, sur papier, dans un tableau, le contenu des variables accessibles lors de l'exécution de ladite instruction Python. Ce tableau indiquera pour chaque variable : son nom, son type, sa valeur et sa portée. Notez également ce qui s'affiche sur la sortie standard.

Exercice 2 : portée des variables avec même nom

On considère le programme suivant :

```
1  #!/usr/bin/env python3
2
3  a = 7
4
5  def f(a):
6      b = 8
7      a = a + b
8      print(a)
9
10 a = 11
11 print(a)
12 f(a)
13 print(a)
```

Question 1

> Même question que dans l'exercice précédent.

Exercice 3 : typage dynamique

On considère le programme suivant :

```python
#!/usr/bin/env python3

a = "1"
b = "3"
print(a + b)

a = 1
b = 3
print(a + b)

a = "1"
b = 3
print(a + b)

t1 = (1, 3)
print(t1[0] + t1[1])

t2 = (1, "3")
print(t2[0] + t2[1])
```

Question 1

> Même question que dans les exercices précédents.

Exercice 4 : et dans un autre langage, ça donne quoi ?

On considère le programme suivant :

```ruby
#!/usr/bin/env ruby

a = 2010

def f(p)
  puts "bienvenue "
  return p + 11
end

puts "à l'Ensimag "

a = f(a)
puts "en " + String(a)
puts "(signé Yoda)"
```

Question 1

> Même question que dans les exercices précédents.

Exercice 5 : exceptions

Question 1

Qu'affiche le programme Java ci-dessous sachant que le point d'entrée d'un programme Java est nécessairement la fonction `public static void main(String[] args)` ?

```
1
2       public static int f(int i) {
3           System.out.println("I am f, and I am going to divide " +
4                              String.valueOf(i) +
5                              " by (i - 42) and return the result");
6           int i_minus_42 = i - 42;
7           return i / i_minus_42;
8       }
9
10      public static void g(int i) {
11          System.out.println("I am g, and because I am lazy, I am just going " +
12                             "to callf(i) and print what it returned to me");
13          int res_f = f(i);
14          System.out.println("f(" + i + ") = " + res_f);
15      }
16
17      public static void main(String[] args) {
18          g(84);
19          g(42);
20      }
```

Question 2

Qu'affiche le programme Python ci-dessous ?

```
1   #!/usr/bin/env python3
2
3
4   def f(i):
5       print(
6           "I am f, and I am going to divide "
7           + str(i)
8           + " by (i - 42) and return the result"
9       )
10      i_minus_42 = i - 42
11      if i_minus_42 == 0:
12          raise ArithmeticError("Une division par 0, mais " "pour qui vous prenez vous
                ?")
13      return i / i_minus_42
14
15
16  def g(i):
17      print(
18          "I am g, and because I am lazy, "
19          "I am just going to callf(i) and print what it returned to me"
20      )
21      try:
22          res_f = f(i)
23          print("f(" + str(i) + ") = " + str(res_f))
24      except ArithmeticError:
25          print("Oulala, f a fait des bétises")
```

```
26
27
28  def main():
29      g(84)
30      g(42)
31
32
33  main()
```

Exercice 6 : mes propres types (pour aller plus loin)

On considère le programme suivant :

```
1   #!/usr/bin/env python3
2   """Illustration des namedtuple"""
3
4   import collections
5
6   bob = ("Robert", 23)
7   bill = ("William", 47)
8
9   # affiche le prénom de bob et l'age de bill (beurk)
10  print(bob[0])
11  print(bill[1])
12
13  bill[1] = 43
14
15  Personne = collections.namedtuple("Personne", "prenom, age")
16
17  bob_nt = Personne("Robert", 23)
18  bill_nt = Personne("William", 47)
19
20  # affiche le prénom de bob et l'age de bill (plus lisible !)
21  print(bob_nt.prenom)
22  print(bill_nt.age)
23
24  bill_nt.age = 43
```

Question 1

> Même question que dans les exercices précédents.

Exercice 7 : et encore dans un autre langage, ça donne quoi ? (pour aller plus loin)

Question 1

> Réécrire le programme de l'exercice précédent dans le langage impératif de votre choix (autre que Python et Ruby, en C par exemple).

Question 2

> Identifier les différences fondamentales avec Python.