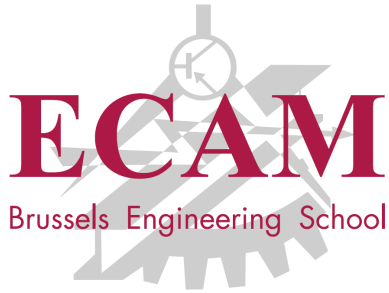

IC1T Programmation

Cours 2
Contrôle de flux

Sébastien Combéfis, Quentin Lurkin

Rappels

- Calculs en Python
  - Mode interactif de Python
  - Expressions et valeurs
  - Opérateurs arithmétiques
  - Priorité des opérations
  - Fonctions mathématiques
  - Variables

Objectifs

- Scripts Python
  - Exécuter un script
  - fonction `print()`
  - paramètres nommé d'une fonction
  - fonction `input()`
- Contrôle de flux
  - opérateurs de comparaison, booléen et opérateurs logiques
  - instuctions blocs
  - `if elif else`
  - boucle `while`

Scripts

Scripts Python

- Script = fichier texte standard

  généralement sauvé avec l'extension `.py`

- Python est un langage interprété

  L'interprétation a lieu à chaque exécution.

- Un script est réutilisable

  Les données ne devraient pas être hardcodées.

Afficher des résultats

- Un script n'affiche rien si on ne le demande pas.

  pas comme le mode interactif

- La fonction `print` permet d'afficher une ou plusieurs valeurs

```
1  print("Hello World !!")      # affiche 'Hello World !!'
2  print(3, "x", 5, "=", 3*5)  # affiche '3 x 5 = 15'
```

Paramètres nommés d'une fonction

- La fonction `print()` utilise une chaîne de séparation entre chaque valeur.

  par défaut c'est l'espace `" "`

- La fonction `print()` utilise une chaîne de fin à chaque appel.

  par défaut c'est le passage à la ligne `"\n"`

- Les chaînes par défaut peuvent être modifiées .

  grâce aux paramètres nommés `sep` et `end`

```
1  print(24, "08", 1982, sep="/")      # affiche '24/08/1982'
2  print("super", end=" ")
3  print("cool")                       # affiche 'super cool'
```

Scripts interactifs

- Un script est réutilisable

  Les données ne devraient pas être hardcodées.

- La fonction `input()` permet à l'utilisateur d'entrer une chaîne de caractères.

  Cette fonction renvoie toujours une chaîne de caractères

- Les fonctions `int()`, `float()`, `complex()` et `str()` permettent de convertir le type d'une valeur.

```
1 x = int(input("Entrez un nombre"))
2 y = int(input("Entrez un autre nombre"))
3 print(x, "+", y, "=", x+y)
```

Contrôle de flux

Contrôle de flux

- modifier le flux d'exécution du programme

  Sur base de conditions.

```
1  answer = input("As-tu faim? ")
2  if answer == "oui":
3      print("Va manger un hamburger !")
4  else:
5      print("Va étudier !")
```

Instructions blocs

- Instruction qui contient un groupe d'instructions
- Elle commence par un entête

  l'entête se termine par un :
- Les instructions contenues dans le bloc sont indentées.

  toutes de le même façon

```
1  entête:
2      instruction 1
3      instruction 2
4      instruction 3
5      ...
```

Opérateur de comparaison

- Comparaison de deux valeurs
  - Égalité (==) et différence (!=)
  - Strictement plus grand/petit (>, <)
  - Plus grand/petit ou égal (>=, <=)
- Comparaison limitée aux types compatibles

  Sans quoi il faut procéder à des conversions

```
1  a = 12 == 3 * 4          # a vaut True
2  b = "Eat" > "Drink"      # b vaut True
3  c = a != b               # c vaut False
```

Opérateur logique

- Combinaison d'expressions booléennes
  - NON logique (`not`) inverse une valeur
  - ET logique (`and`) impose les deux expressions à `True`
  - OU logique (`or`) nécessite une seule expression à `True`

a	b	not a	a and b	a or b
False	False	True	False	False
False	True		False	True
True	False	True	False	True
True	True		True	True

```
1  a = 8 > 2 and 12 <= 4                    # a vaut False
2  b = 5 != 5 or 'PY' == 'P' + 'Y'          # b vaut True
```

Instruction `if`

- Exécution d'un bloc de code si une condition est vérifiée

  Dans tous les cas, le programme continue après l'instruction `if`

```
1  x = -5
2  if x <= 0:
3      print("x est négatif !")
4      print("sa valeur absolue vaut ", -x)
5
6  print("Fin du programme")
```

```
x est négatif !
sa valeur absolue vaut 5
Fin du programme
```

Instruction `if-else`

- Exécution alternative si la condition n'est pas vérifiée

  Dans tous les cas, le programme continue après le `if-else`

```
1  grade = 9.5
2  if grade >= 10:
3      print("vous avez réussi")
4  else:
5      print("vous avez raté")
6
7  print("Fin du programme")
```

```
vous avez raté
Fin du programme
```

Instruction `if-elif-else`

- Définition de plusieurs alternatives disjointes

  Dans tous les cas, le programme continue après le `if-elif-else`

```
1  temp = 126
2  if temp < 100:
3      print("tout va bien")
4  elif 100 <= temp <= 130:
5      print("attention")
6  else:
7      print("danger")
8
9  print("Fin du programme")
```

```
attention
Fin du programme
```

Exemple

```
1  from math import sqrt
2
3  # Coefficients du trinôme
4  a = 1
5  b = -4
6  c = 2e2
7
8  # Calcul du discriminant
9  delta = b ** 2 - 4 * a * c
10
11 # Calcul des racines
12 if delta > 0:
13     x1 = (-b + sqrt(delta)) / (2 * a)
14     x2 = (-b - sqrt(delta)) / (2 * a)
15     print("Les solutions sont", x1, "et", x2)
16 elif delta == 0:
17     x = -b / (2 * a)
18     print("La solution double est", x)
19 else:
20     print("il n'y a pas de solution réelle")
```

Instruction `while`

- Répète un bloc de code tant qu'une condition est vérifiée

Dans tous les cas, le programme continue après `while`

```
1 n = 1
2 while n <= 5:
3     print(n)
4     n = n + 1
```

```
1
2
3
4
5
```

Exemple

```
1  x = int(input("Entrez un nombre: "))
2  i = 1
3  while i <=10:
4      print(i, "x", x, "=", i*x)
5      i += 1
```

```
Entrez un nombre: 7
1 x 7 = 7
2 x 7 = 14
3 x 7 = 21
4 x 7 = 28
5 x 7 = 35
6 x 7 = 42
7 x 7 = 49
8 x 7 = 56
9 x 7 = 63
10 x 7 = 70
```

Exemple

```
1  x = int(input("Entrez un nombre: "))
2  sqrt = 1
3  while abs(sqrt - x/sqrt) > 1e-6:
4      sqrt = (sqrt + x/sqrt) / 2
5  print("La racine carrée de", x, "est", sqrt)
```

```
Entrez un nombre: 2
La racine carrée de 2 est 1.4142135623746899
```

Attention aux boucles infinies

- Il faut que la condition de la boucle puisse devenir fausse

  sinon elle ne s'arrête jamais

```
1       n = 1
2       while n <= 5:
3           print(n)        # boucle infinie !!
```

```
1       n = 1
2       while n > 0:
3           print(n)
4           n += 1          # boucle infinie !!
```

Crédits

- https ://www.flickr.com/photos/jakerust/16659686228/
- https ://www.flickr.com/photos/pfranche/18779176121/