Informatique tronc commun : Introduction

H. Menet, menet.hugo@gmail.com

Sept 2023

Lycée Saint Louis

Documents de cours, dont cette présentation, disponible sur :

Prendre du papier et un crayon : pour prendre des notes, pour le TD de rappel, comme brouillon pour le TP ...

Informatique en prépa

Présentation

Concours :

- Ecrits sur papier : 2h à Centrale, Mines et X, 3h CCINP et E3A
- Oraux : Epreuve de maths et physique avec machine à Centrale, utile pour les TIPE

Présentation

Concours :

- Ecrits sur papier : 2h à Centrale, Mines et X, 3h CCINP et E3A
- Oraux : Epreuve de maths et physique avec machine à Centrale, utile pour les TIPE

Emploi du temps :

- 2h TP 1 semaine sur 2, par groupes de 16, en salle machine
- 2h de cours toutes les deux semaines au S2

Présentation

Concours :

- Ecrits sur papier : 2h à Centrale, Mines et X, 3h CCINP et E3A
- Oraux : Epreuve de maths et physique avec machine à Centrale, utile pour les TIPE

Emploi du temps :

- 2h TP 1 semaine sur 2, par groupes de 16, en salle machine
- 2h de cours toutes les deux semaines au S2

Evaluation :

- Quelques DM sur papier, avec des parties machines.
- DS de 2h toutes les 3 semaines le samedi matin (avec la SI)

Programme

Un programme centré sur l'informatique :

Au S1 : Prendre en main **Python3**

Applications :

- Recherche dans un tableau
- Tri
- Traitement d'image

Principes :

- Complexité
- Dichotomie
- Récursif
- Glouton

Au S2 :

- Bonnes manières de coder et d'analyser un algorithme
- Représentation des nombres
- Graphes (piles, files, Dijkstra, A*)

Programme

En spé :

- Base de donnée en SQL
- Programmation dynamique
- Machine Learning (supervisé KNN, non supervisé K Means)
- Etude des jeux (minmax)

Esprit du programme

- Emphase sur l'algorithmie, et sur les bonnes pratiques de développement, avec un première aperçu de notions importantes de l'informatique
- "Nouveau" programme, avec pour le moment une seule session de concours en 2023, mais l'informatique, sous d'autres formes, existe depuis longtemps en prépa.
- MPSI : **Option info** au S2 : Théorie des langages (fondement de l'informatique), logique, un peu plus d'algorithmique. Un autre langage très intéressant et très différent de Python : OCaml

Programme officiel

Jeter un œil au programme officiel du S1, et à l'annexe A, détaillant les éléments du langage Python à connaître en fin de semestre.

TPs du S1

1. Prise en main 1
2. Prise en main 2
3. Prise en main 3
4. Algorithme dichotomique et preuve de programme
5. Modules et lecture de fichier
6. Récursivité
7. Tris
8. Approche glouton
9. Traitement d'image

Prise en main

A la fin de la prise en main, il faudra savoir être indépendant en Python.

Etre au point sur l'ensemble de l'annexe A (à l'exception de la section Divers).

Notion de complexité. Manipuler une liste.

Programmation et Python

Variable et type

Definition
Une variable est un espace mémoire abstrait auquel on associe un nom, et qui contient une valeur.

Le **type** d'une variable va décrire la forme que va prendre cet espace mémoire abstrait et les façons dont il peut être lu (donc sa valeur) et modifié.

Typage dynamique

En Python, le typage est **dynamique**. Pendant l'exécution du programme, le type d'une variable est inféré à partir de la valeur donnée à l'initialisation de la variable.

```
1  x=3
```

Types et structures de données

Quels types connaissez-vous ?

Types et structures de données

Quels types connaissez-vous ? On va revoir aujourd'hui :

- int
- float
- str
- bool

On reverra ensuite :

- list
- dict

Instructions

- Condition : `If, else, elif`
- Boucle bornée : `For`
- Boucle non bornée : `While`

En Python, l'**indentation** permet de définir où commencent et s'arrêtent les instructions.

```
1  a=-1
2  if a>0:
3       a=2
4  a=a+1
```

```
1  a=-1
2  if a>0:
3       a=2
4       a=a+1
```

Tableau de suivi des variables

Tableau de suivi de variable et exécution ligne à ligne :

Code Python :

```
1  x=3
2  y=x+2
3  x=y
4  print(y)
```

Tableau de suivi des variables :

Ligne	x	y
1		
2		
3		
4		

Sortie :

Tableau de suivi des variables

Tableau de suivi de variable et exécution ligne à ligne :

Code Python :

```
1  x=3
2  y=x+2
3  x=y
4  print(y)
```

Tableau de suivi des variables :

Ligne	x	y
1	3	
2	3	5
3	5	5
4	5	5

Sortie : Affichage "5"

Exercice

Etant donné une variable x, et une y, écrire un script pour que la valeur contenue dans la variable x soit dans la variable y, et celle de la variable y dans x.

Exercice

Donner le type des variables :

```
1  x=3
2  s="cha"
3  nombre="4"
4  entier=4.43
5  d=1.0
6  L=True
7  y=x+d
8  n=s+nombre
```

Exercice

Donner le type des variables :

```
1  x=3
2  s="cha"
3  nombre="4"
4  entier=4.43
5  d=1.0
6  L=True
7  y=x+d
8  n=s+nombre
```

Respecter les conventions de nommage !

Operations

int : + , -, *, //, %, **

float : +,-,*,/,**

// est la division entière, au sens de la division euclidienne.
/ est la division flottante, commence par transformer en flottant, puis divise.
a % b correspond à a modulo b, qui renvoit le reste dans la division euclidienne de a par b.

Exercice

Qu'affiche les lignes suivantes ?

```
1  5//2
2  5/2
3  8%2
4  5%3
5  12%17
```

Fonctions

Fonctions

```
1  def signe(x):
2      if x > 0:
3          print("Strictement positif")
4      else:
5          print("Negatif")
```

Que se passe-t-il quand on execute ce code ?

Fonctions

```
1  def signe(x):
2      if x > 0:
3          print("Strictement positif")
4      else:
5          print("Negatif")
```

Que se passe-t-il quand on execute ce code ? Rien

Fonctions

```
1  def signe(x):
2      if x > 0:
3          print("Strictement positif")
4      else:
5          print("Negatif")
```

Que se passe-t-il quand on execute ce code ? Rien Et celui-ci ?

```
1  signe(-5)
```

Sortie :

Fonctions

```
1  def signe(x):
2      if x > 0:
3          print("Strictement positif")
4      else:
5          print("Negatif")
```

Que se passe-t-il quand on execute ce code ? Rien Et celui-ci ?

```
1  signe(-5)
```

Sortie : "Negatif"

Fonctions

```
1  def signe(x):
2      if x > 0:
3          print("Strictement positif")
4      else:
5          print("Negatif")
```

Que se passe-t-il quand on execute ce code ? Rien Et celui-ci ?

```
1  signe(-5)
```

Sortie : "Negatif"

Modifier cette fonction pour également séparer les cas "strictement négatif" et "nul".

Print et return

Print ne renvoie rien, return si.

Return arrête l'execution de la fonction.

Portée lexicale

Variable locale et globale

```
1  x=3
2  def f(x):
3      return x**2
4  def g(y):
5      return x+y
6  print(f(5))
7  print(g(5))
```

Instruction et opérations

Opérations

`bool` :

`or, and, not`

Opérations

`bool` :

`or, and, not`

a	b	a and b	a or b
True	True	True	True
True	False	False	True
False	True	False	True
False	False	False	False

a	not a
True	False
False	True

Tables de vérité du and, or et not.

Exercice

Donner le résultat de chaque ligne.

```
1 True and True
2 True and False
3 False or True
4 not(True)
5 not(True and (False or True))
```

Evaluation de booléens

L'évaluation est dite paresseuse.

```
1 if True or 1000**100 >0 :
2     print("Bien")
```

On ne calcule pas 1000^{100}.

Comparaison

Les comparaisons renvoie des booléens !

`int` :

```
1  ==, !=, >, <, >=, <=
```

`float` :

```
1          >, <, >=, <=
```

Comparaison

Les comparaisons renvoie des booléens !

`int` :

```
1  ==, !=, >, <, >=, <=
```

`float` :

```
1          >, <, >=, <=
```

Jamais de test d'égalité entre variables de type float !

On peut cependant tester la distance entre deux flottants, en utilisant un seuil, un epsilon, en vérifiant que ce seuil est plus grand que la précision de la machine.

Comparaison

Les comparaisons renvoie des booléens !

`int` :

```
1  ==, !=, >, <, >=, <=
```

`float` :

```
1           >, <, >=, <=
```

Jamais de test d'égalité entre variables de type float !

On peut cependant tester la distance entre deux flottants, en utilisant un seuil, un epsilon, en vérifiant que ce seuil est plus grand que la précision de la machine.

```
1  1+10**45 - 10**45 == 1
2  1.0+10**45 - 10**45 == 1
```

Exercice

Ecrire une fonction qui prend en entrée un entier et renvoie True s'il est pair, False sinon.

Exercice

Ecrire une fonction qui prend en entrée un entier et renvoie True s'il est pair, False sinon.

```
1  def pair(n):
2      return n%2==0
```

Exercice

Quels sont les types des variables définis ci-dessous ?

```
1  x = 5%2 == 0
2  y = 5/2
3  z = 5//2
4  t=1.0+5
```

Exercice

On définit une suite (u_n) pour n entier naturel par $u_0 = 5$ et $u_{n+1} = 4u_n + 3$.
Ecrire un script pour calculer u_1, puis u_2, puis u_{10}.

For

Boucle bornée, boucle for, boucle pour

On connaît le nombre de répétition avant de commencer.

On repète *n* fois l'instruction, avec une variable i qui va de 0 à n-1, en changeant à chaque répétition.

For

Boucle bornée, boucle for, boucle pour

On connaît le nombre de répétition avant de commencer.

On repète *n* fois l'instruction, avec une variable i qui va de 0 à n-1, en changeant à chaque répétition.

```
1  for i in range(5):
2      print(i)
3  0
4  1
5  2
6  3
7  4
```

Range

```
1 range(n) : de 0 a n-1 : n passages !
2
3 range(start, stop, pas)
```

Range

```
1  range(n) : de 0 a n-1 : n passages !
2
3  range(start, stop, pas)
```

Exemple :

```
1  for i in range(2, 4, -1):
2      print(i)
3  for i in range(1, 6, 2):
4      print(i)
```

La mémoire

Dresser le tableau de suivi des variables pour le programme suivant.

```
1  u=3
2  v=4
3  for i in range(4):
4      u=u+2
5  u=v-u
6  print(u)
```

Exercice

Ecrire une fonction qui calcule u_n étant donné n, avec (u_n) défini pour n entier naturel par $u_0 = 5$ et $u_{n+1} = 3u_n + 2 + n$.

Exercice

On considère la suite $(u_n)_{n\in\mathbb{N}}$ définit par $u_0 = 10$ et $u_{n+1} = u_n + 5$ pour $n > 0$.
Quelle est la limite de cette suite ? Ecrire un script qui affiche le premier rang n pour lequel $u_n > 100$.

While

Boucle non bornée, boucle while, boucle tant que

On ne connaît pas à l'avance le nombre de répétitions nécessaires.

On repète les instructions tant que la condition en entrée n'est pas vérifié.

While

Boucle non bornée, boucle while, boucle tant que

On ne connaît pas à l'avance le nombre de répétitions nécessaires.

On repète les instructions tant que la condition en entrée n'est pas vérifié.

```
1  a=30
2  div=4
3  q=0
4  while u > 0 :
5      a=a-4
6      q=q+1
7  print(q)
```

Dresser un tableau de suivi des variables.

Que fait cette suite d'instructions ?

On aura très souvent besoin d'un compteur (ici "q").

Exercice

1. Ecrire une fonction qui prend en entrée un premier terme u0, et une raison r positive, et renvoie le rang du premier terme supérieur à 100 de la suite arithmétique de premier terme u0, et de raison r.
2. Modifier la fonction pour que le seuil soit donné en argument.
3. La raison n'est plus supposée positive, modifier la fonction pour renvoyer -1 si la suite ne dépasse jamais le seuil, et le rang de dépassement sinon.

TP

Ouvrez vos sessions

1 personne par poste

Allumez vos ordinateurs et lancez vos sessions.

Pyzo

Ouvrir Pyzo : console et fichier.

Suivre l'énoncé du TP1.