

Partie II – Les fonctions

- Plan :
- Introduction : programmes et fonctions
- Ecriture de fonctions avec Python
 - fonction sans paramètre
 - fonction avec paramètres
 - paramètres formels / paramètres effectifs
- Définition d'une « vraie » fonction
- Test d'une fonction
- Typage des paramètres

Introduction

- deux points de vue pour définir un programme :
- une suite d'instructions
- une fonction (mathématique)
- 1) programmation impérative
- 2) programmation fonctionnelle
- Python : premier point de vue
- comment écrire un programme?

Introduction

- exemple :

Calcul et affichage de la surface d'un triangle isocèle de base 11 cm et de hauteur 5 cm

```
b=11
h=5
surface=b*h/2
print ("la surface est de", surface, "cm2")
```

Introduction

- programmation impérative : programme = suite d'instructions exécutées séquentiellement (l'une après l'autre)
- suite potentiellement très longue
- décomposition du problème en sous problèmes plus simples (éventuellement décomposés aussi)
- plus facile à écrire, maintenir
- réutilisables pour d'autres problèmes

Fonctions avec Python

- spécification : **fondamental** de l'écrire
- contrat entre utilisateur et concepteur
- cinq informations :
 - le nom de la fonction
 - les types des données en entrée (paramètres)
 - les types des résultats
 - les noms des paramètres
 - ce qu'elle fait

Fonctions avec Python

- types :
 - vide : ne retourne rien
 - booléen : vrai ou faux
 - naturel : entier ≥ 0
 - entier : entier relatif
 - réel : flottant
 - nombre : réel U entier
 - chaîne : chaîne de caractères
 - indifférent : n'importe quel type

Fonctions avec Python

- définition de la fonction :
instruction composée avec le mot-clé def :
entête :
<indentation> instruction 1
...
<indentation> instruction n
dans le cas d'une fonction :
def nomfonction (liste paramètres) :
 (entête de la fonction)
 instructions de la fonction
 (corps de la fonction)

Ifsisc – Univ Rennes

31

Fonctions avec Python

- **tout programme doit être accompagné de commentaires**
- avant la fonction (spécification : ce qu'elle fait, ce qu'elle reçoit en entrée et donne en sortie)
- la description doit être donnée en commentaire sur la première ligne
- imposé en TP

Ifsisc – Univ Rennes

32

Fonctions avec Python

- nom de la fonction suivi de (), exemple :
#declaration d'une variable globale
b=12
h=225
#fonction aireTriangle11_5 : vide -> vide
def aireTriangle11_5 ():
 print("affiche surface triangle de base 11, hauteur 5")
 h=11
 b=5
 surface=h*b/2.
 print("la surface est de", surface,"cm2")

Ifsisc – Univ Rennes

33

Fonctions avec Python

- ```
##programme principal
aireTriangle11_5()

#les variables globales b et h ne sont pas modifiées

print ("h = ", h, "et b =", b)

▪ résultat de l'exécution du programme :
affiche surface triangle de base 11, hauteur 5
la surface est de 27.5 cm2
h = 225 et b = 12
```

Ifsisc – Univ Rennes

34

## Fonctions avec Python

- variable locale / variable globale
- dans la fonction, 3 variables : b, h et surface
- toute variable définie dans une fonction n'a d'existence que dans la fonction et est inexistante en dehors : ce sont des **variables locales**
- variable globale de même nom, seule la variable locale est modifiée (cas de b et h dans notre exemple)
- utilisation de variables globales rend **mise au point difficile**

Ifsisc – Univ Rennes

35

## Fonctions avec Python

- fonctions avec paramètre(s)
- pourquoi que la surface d'un triangle de 11 par 5?
- b et h en paramètres formels, calcul de l'aire de triangle de n'importe quelle base / hauteur (dont on donnera la base et la hauteur en paramètres effectifs)
- exemple de :
  - spécification
  - mise en œuvre
  - utilisation

Ifsisc – Univ Rennes

36

## Fonctions avec Python

```
déclaration d'une variable globale
b=12
h=225

aireTriangle : nombre x nombre -> vide
b, h ->
def aireTriangle (b,h):
 print("Affiche la surface d'un triangle de base b et de hauteur h")
 surface=h*b/2.
 print ("La surface est de ",surface, "cm2")
```

Ifsisc – Univ Rennes

37

## Fonctions avec Python

```
Programme principal
print ("Pour un triangle de base 5 cm et de hauteur 11 cm")
aireTriangle(5,11)

print() #affichage d'une ligne vide

print ("Pour un triangle de base 7 cm et de hauteur 2 dm")
aireTriangle(7,20)
les variables globales b et h ne changent pas

print ("h est égal à ", h, "et b à ", b)
```

Ifsisc – Univ Rennes

38

## Fonctions avec Python

- dans la définition de aireTriangle, b et h sont des paramètres formels
- dans l'appel aireTriangle (6,3), 6 et 3 sont des paramètres effectifs

Ifsisc – Univ Rennes

39

## Vraie fonction avec Python

- après compilation la fonction est utilisable comme n'importe quelle fonction du logiciel (par ex. print())
- mais ce n'est pas une "vraie" fonction
- elle ne donne pas de résultat
- c'est une procédure
- ne peut pas être utilisée par une autre fonction pour un calcul plus complexe
- pour rendre un résultat, il faut utiliser le mot clé **return**

Ifsisc – Univ Rennes

40

## Vraie fonction avec Python

- utilisation de la fonction pour réalisation d'une autre fonction
- aire d'un polygone régulier (n cotés de taille c)
- propriété : l'aire d'un polygone de n côtés de taille c = aire n triangles isocèles de base c et de hauteur  $c/(2 \times \tan(\pi/n))$

Ifsisc – Univ Rennes

41

## Vraie fonction avec Python

```
surfaceTriangle : nombre x nombre -> réel
b, h -> surface triangle base b et hauteur h
def surfaceTriangle (b,h):
 "Rend la surface du triangle de base b et de hauteur h"
 aire = b*h/2.
 return aire
Programme principal, calcul pour b=6 et h =8
resultat = surfaceTriangle(6,8)
print ("La surface pour b = 6 et h= 8 est : ",surfaceTriangle(6,8))
print ("La surface pour b = 6 et h= 8 est : ",resultat)
#calcul pour b=6 et h =11
print ("surface pour b=6 et h=11 : ",surfaceTriangle (6,11))
```

Ifsisc – Univ Rennes

42

## Vraie fonction Python

- utiliser cette fonction pour en réaliser une autre
- calculer l'aire d'un polygone régulier (n côtés de taille c)
- aire = n triangles isocèles (base c, hauteur  $c/(2 \times \tan(\pi/n))$ )

## Vraie fonction Python

```
import math
#surfacePolygone : naturel x nombre -> reel
n , c -> surface d'un polygone à n cotés
(n>2) chaque coté faisant c cm
def surfacePolygone (n,c):
 print("surface d'un polygone à n cotés (n>2) de c cm")
 return n*surfaceTriangle(c,c/ (2*math.tan(math.pi/n)))
surfaceTriangle : nombre x nombre -> réel
b, h -> surface triangle base b hauteur h
def surfaceTriangle (b,h):
 "rend surface triangle base b hauteur h"
 return b*h/2.
```

## fonctions avec Python

### #Tests

```
print ("Avec n=3, c=4 :", SurfacePolygone(3,4))

print ("Avec n=6, c=7 :", surfacePolygone(6,7))

print ("Avec n=2, c=4 :", surfacePolygone(2,4))

print ("Avec n=%d, c=%d : %f"%(3,4,surfacePolygone(3,4)))

print ("Avec n=2, c=4 : %f" %surfacePolygone(2,4))
```

## Tests

- primordial dans la programmation
- le code produit correspond à la spécification
- couverture des cas possibles
- stratégie avant programmation

## Typage paramètres

- typage paramètre est celui effectif à l'appel
- si paramètre incorrect, résultat non garanti
- spécification = contrat d'utilisation