TD-TP 1
Algorithmique du traitement des données

Raphaël Fournier-S'niehotta, fournier@cnam.fr

Octobre 2016

1 Algorithmique

1.1 Méthodologie de conception

Pour écrire un programme il faut suivre les étapes suivantes :

1. Déterminer les informations nécessaires pour résoudre le problème. Ce sont les données ou encore les entrées du programme
2. Déterminer ce que l'on veut calculer. Ce sont les résultats ou sorties du programme
3. Être capable soi même de calculer les sorties à partir d'exemples de données. Ceci est une étape de compréhension du problème.
4. Décrire comment on calcule les sorties par rapport aux entrées.
5. Écrire le programme dans un langage de programmation Pour ce qui est de la dernière étape, nous apprendrons dans les prochains cours comment la réaliser. Pour cette première séance d'exercice, nous allons nous arrêter à l'étape 4. Prenons d'abord un exemple.

Exemple

On considère le problème suivant : "tester si trois nombres entiers sont triés dans l'ordre croissant." Les réponses aux quatre premières étapes de conception sont :

1. entrées : 3 entiers que nous appellerons (a), (b) et (c)
2. Sorties : la réponse au problème est de nature booléenne : soit la propriété testée est vraie (la valeur calculée par le programme sera alors true) soit elle est fausse (valeur false). En résumé, la sortie du programme sera un booléen que nous appellerons (rep)
3. Quelques exemples :

a	b	c	rep	explication
6	10	45	true	$6 \leq 10 \leq 45$
6	7	4	false	$6 \leq 7 > 4$
6	3	2	false	$6 > 3$

4. Si $a < b$ et $b < c$ alors rep vaut VRAI (true) sinon rep vaut FAUX (false)

1.2 Application

Faire les étapes 1, 2 et 3 et 4 pour chacuns des problèmes suivants :

1. Calculer la note finale d'une unité d'enseignement étant données la note du partiel et celle de l'examen et sachant que :
   - une note inférieure à 7 à l'examen est éliminatoire
   - si la note d'examen est supérieure à 7, la note finale est la moyenne des deux notes si elle avantage l'étudiant. Sinon, la note finale est la note d'examen.
2. Tester si un entier appartient à l'intervalle donné par deux nombres entiers.
3. Calculer puis afficher le plus grand parmi trois nombres entiers.
4. Tester si trois nombres entiers sont triés dans l'ordre croissant ou décroissant.
5. Tester si une année est bissextile. On sait qu'une année divisible par 4 est bissextile sauf si elle est divisible par 100, cependant les années divisibles par 400 sont également bissextiles.

2 Terminologie

Soit le morceau de code suivant : (x * 3) + 1

- quel nom donne-t-on à ce morceau de code tout entier ?
- quels sont les opérateurs ?
- quels sont les opérandes de * ?
- quels sont les opérandes de + ?
- quel genre de chose est x dans ce morceau de code ?
- quel(s) type(s) a ou peut éventuellement avoir x ?
- quel est le type de 3 et 1 ?
- quel est le type de (x * 3) + 1 ?
- quelle est la valeur de (x * 3) + 1 ?

3 Traces d'exécution

Dans cet exercice, nous allons donner des programmes dont il faut simuler sur papier l'exécution pas à pas. Il faut suivre l'évolution de la mémoire et de l'affichage à chaque ligne de code. Nous allons d'abord vous proposer un exemple pour illustrer ce que nous attendons comme réponse.

```
print("Somme en euros? ")
euros = input()
dollars = euros *1.10
print("La somme en dollars :"+ dollars)
```

	euros	dollars	clavier	ecran
1	X	X		Somme en euros?
2	10	X	10	
3	10	11		
4	10	11		La somme en dollars : 11

Le tableau suivant donne la valeur de chaque variable, des entrées du clavier de l'affichage à l'écran pour chaque ligne de code. Avant la ligne 3, les variables n'existent pas. On notera cela avec un X dans la colonne correspondante. On notera par un ? le contenu des variables déclarées et non initialisées.

La première colonne comporte un numéro de ligne et les autres colonnes comportent les valeurs de variable après exécution de cette ligne de code et les effets se produisant pendant cette exécution. La première ligne du tableau apparaît pour donner l'état initial, avant prise en compte de la première ligne de code. Les entrées au clavier sont arbitraires, au choix de l'utilisateur du programme.

3.1 Somme 1

```
x = 3
y = 4
z = x+y
print("la somme de"+ x + "et de"+ y + "est "+z)
```

3.2 Somme 2

```
x = 3
y = 4
x = x+y
print("la somme de"+ x + "et de"+ y + "est "+x)
```

3.3 Somme 3

```
x = 3
y = 4
print("la somme de"+ x + "et de")
x = x+y
print(y + "est "+x)
```

4 Opérateurs booléens

Qu'affiche ce programme :

```
print(True and False)
a = True
b = True
c = True
print(b and c)
print(not a)
print((not a) or (b and c))
```

Et qu'affiche celui-ci ?

```
a = "x"
b = 2
print("Valeur de 5 > 3: ")
print(5>3)
print("Valeur de a == 'b': ")
print(a == 'b')
print("Valeur de (b>=0) && (b<=100): ")
print((b>=0) && (b<=100))
```

5 Travaux pratiques

Ouvrir un fichier texte et saisir les programmes des deux exercices précédents. Essayer de saisir des valeurs différentes à chaque exécution pour la conversion euros/dollars, observer les résultats. Exécuter plusieurs fois les programmes sur les opérateurs booléens, observer les résultats.