NOTIONS FONDAMENTALES – 2 - TESTS
RESUME DE COURS – EXERCICES

Version sept 2018

4. ++ Test

Troisième exemple

```
Programme racineCarré
    Lire (x)
    si x  < 0 alors
        afficher(« Pas de racine »)
    sinon
        racine = racineCarrée(x)
        afficher (racine)
    finsi
Fin
```

si – sinon – finsi, Evaluation de la condition, Bloc d'instructions, Fonction racine

Le sinon est facultatif :

```
    si condition alors
        bloc1
    finsi
```

sinonsi

```
    si condition alors
        bloc1
    sinonsi
        bloc2
    sinonsi
        bloc3
    …
    sinon
        bloc4
    finsi
```

Le sinon est facultatif.

Switch

```
switch variable vaut
cas 1 : expression ou liste d'expressions
        bloc1
cas 2 : expression ou liste d'expressions
        bloc2
…
défaut
        blocDefaut
finSwitch
```

Le « switch» fait une comparaison d'égalité entre la variable et l'expression ou une liste d'expressions.

Le « défaut » est facultatif.

Expression booléenne, arbre dichotomique

Simulation

	x	x < 0	racineCarrée(x)	racine	écran
Lire (x)	4				
x < 0		faux			
racineCarrée(x)			2		
racine = racineCarrée(x)				2	
Afficher (racine)					2

Python de base

```
from math import * # importation des functions mathématiques
x=float(input('entrez un reel : '))
if x<0 :
    print("pas de racine")
else :
    racine = sqrt(x)
    print ("la racine de ",x," vaut ", racine)
    print ("la racine de ",x," vaut ", round(racine,2)) # round arrondit à 2 chiffres
    print ("la racine de %g vaut %.2f" % (x, racine)) # %g : adapté, %.2 : 2 après ,
```

EXERCICES – SERIE 2 – TESTS

Méthode de base d'analyse algorithmique

La méthode de base pour écrire un algorithme suit les 4 étapes suivantes :

1. Comprendre le problème : bien lire le sujet et bien comprendre ce qu'il y a à faire.
2. Lister ce dont on a besoin pour résoudre le problème (les données) et ce qu'on va produire (les résultats) : préciser les Entrées et les Sorties.
3. Trouver un principe de résolution : se donner les grandes lignes, en français, de la méthode de résolution. Eventuellement, se donner des procédures ou des fonctions (des actions générales).
4. Ecrire l'algorithme en détail.

Bases

Exercice 1 : valeur absolue

Écrire un programme qui calcule la valeur absolue de la différence de deux nombres.

Exercice 2 : pair ou pas

Écrire un programme qui dit si un entier est pair ou pas.

Exercice 3 : max de 2 nombres

Écrire un programme qui calcule le plus grand de deux nombres.

Exercice 4 : max de trois nombres

Écrire un programme qui calcule le plus grand de trois nombres.

Exercice 5 : inverse d'un nombre

Écrire un programme qui calcule l'inverse d'un nombre. Il faut traiter les cas particuliers.

Exercice 6 : inverser deux nombres

Écrire un programme qui inverse les valeurs de deux variables

Exercice 8 : photocopieuse

Ecrire une fonction qui permet de calculer le prix d'un certain nombre de photocopies en appliquant le tarif suivant : les 10 premières coûtent 0,10 E pièce, les 20 suivantes 0,08 E pièce et toutes les autres 0,07 E pièce.

Exercices avancés

Exercice 1 : trier 3 nombres

Écrire un programme qui permet de redéfinir 3 variables, X, Y, Z de façon à obtenir : X < Y < Z

Exercice 2 : année bissextile

Ecrire un programme qui affiche le nombre de jours d'un mois d'une année donnée. On rappelle que les années bissextiles sont celles qui sont divisibles par 4. Toutefois, les années divisibles par 100 ne sont pas bissextiles. Mais les années divisibles par 400 sont quand même bissextiles. Par exemple : 2008 est bissextile, 2010 n'est pas bissextile, 2000 est bissextile, 1900 n'est pas bissextile.

Branchés maths

Exercice 1 : équation 1er degré

Écrire un programme qui résout dans IR une équation du premier degré en traitant tous les cas possibles (si a=0, etc.)

Exercice 2 : équation 2ème degré

Écrire un programme qui résout dans IR une équation du second degré en traitant tous les cas possibles (si a=0, etc.)