1 La programmation impérative

1.1 Le problème : Résolution des équations du second degré

Pour continuer à pratiquer les énoncés conditionnels :

(a) Faire un programme (dans le langage de votre choix C ou Python) de résolution des équations du second degré de la forme

$$a * x^2 + b * x + c = 0$$

avec $a, b, c \in \Re$.

(b) Réaliser une présentation *élégante* des solutions (indiquer le nombre de racines réelles distinctes, nombre de solutions complexes,...).

Pour autant vous n'utiliserez pas la notion de complexe disponible en C et en Python.

(c) Proposer des jeux de valeurs permettant de tester les différents cas de figure.

(d) Tester votre programme avec le jeu de coefficients suivants : $a = 0.25$, $b = 0.1$ et $c = 0.01$

Vous aurez remarqué que c'est un cas de figure où le discriminant est "théoriquement" nul.

1.2 Démarche de mise en "informatique"

Ce problème se résoud en calculant un discriminant (fonction de a, b, c) qui permet de choisir la forme des solutions :

- ➢ $delta = b^2 - 4 * a * c$
- ➢ ensuite selon la valeur de $delta$:

si $delta > 0$
alors $x1 = \frac{-b-\sqrt{2*a}}{2*a}$ et $x2 = \frac{-b+\sqrt{2*a}}{2*a}$
si $delta == 0$ alors ...

1.3 La voie impérative

① Les variables qui vont me permettre de décrire le problème seront :

$$a, b, c, delta$$

Je n'ai pas prévu de $x1$ ou $x2$ dans la mesure ou je me contente d'afficher les résultats.

② On va déduire les instructions du traitement que l'on sait devoir effectuer mathématiquement.

1.4 Le programme C

```c
/*
  Fichier : seconddeg.c

  Ebauche d'une solution au pb de la resolution d'une equation du
  second deg

  04/09/2017 -    G.MENEZ
 */
#include <stdio.h>   // pour printf et scanf
#include <math.h> // pour sqrt
#include <stdlib.h> // pour EXITSUCCESS

int main (void){

  double a, b, c ;               /*Coefficients reels*/
  double d = 0.0;              /*Delta*/

  /* Saisie au clavier */
  printf("Entrez une valeur de a : ");
  scanf("%lf", &a);
  printf("Entrez une valeur de b : ");
  scanf("%lf", &b);
  printf("Entrez une valeur de c : ");
  scanf("%lf", &c);

  d = pow(b,2)-(4*a*c);     /*Delta*/
  printf("d = %f\n", d);

  /* Des cas, des racines ... */
  if (d == 0)
    printf("Une racine dans R : %f\n", -b/(2*a));
  else if (d > 0){
    printf("Deux racines dans R :\n");
    printf("x1 = %f\n", (-b+sqrt(d))/(2*a));
    printf("x2 = %f\n", (-b-sqrt(d))/(2*a));
  }
  /* etc */

  return EXIT_SUCCESS;
}
```

1.5 Sa compilation et son exécution

La commande :

`gcc seconddeg.c -lm`

produit, si le code est correct, un fichier `a.out` qui est un commande externe du Shell, donc "lançable" :

```
menez@vtr ~/EnseignementsCurrent/Cours_Prog/Tps/PetitsPgm01/Seconddeg
$ gcc seconddeg.c -lm

menez@vtr ~/EnseignementsCurrent/Cours_Prog/Tps/PetitsPgm01/Seconddeg
$ file a.out
a.out: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, for GNU/Linux 2.6.32,
39249f2de9e43adb4, not stripped

menez@vtr ~/EnseignementsCurrent/Cours_Prog/Tps/PetitsPgm01/Seconddeg
$ ./a.out
Entrez une valeur de a : 2
Entrez une valeur de b : 4
Entrez une valeur de c : 1
d = 8.000000
Deux racines dans R :
x1 = -0.292893
x2 = -1.707107
```

1.6 Quelques remarques

① Cette solution n'est pas achevée !

On voit bien comment les instructions viennent modifier les variables qui ont été introduites. On parle d'effets de bord.

② Cette solution n'est pas unique :

- ➢ Des variables pour $x1$ et $x2$?
- ➢ Une fonction ou une macro pour calculer *delta* ?
- ➢ ...