# Devoir Long – Corrigé

## Enoncé :

En binôme, écrire un programme en langage C qui résout un des problèmes ci-dessous. Le programme doit être réalisé pendant la séance de TD et le code produit doit être rendu au chargé de TD à la fin de la séance par email ou par clé USB. Les noms des étudiants composant le binôme doivent être inscrits dans le code, sous la forme d’un commentaire.

## Solutions :

### Problème 1 – Résolution d’équations de second degré

#### Enoncé

Etant donné une équation du second degré *ax2 + bx + x = 0*, avec *a*, *b* et *c* fournis par l’utilisateur, trouver les solutions pour cette équation.

Rappel :

Etant donné l’équation *ax2 + bx + x = 0*, le discriminant (delta) est défini comme suit :  
Δ = *b*2 − 4*ac*.

Alors l'équation a deux solutions réelles distinctes *x*1 et *x*2 :

x_1 = \frac{-b + \sqrt{\Delta}}{2a} et x_2 = \frac{-b - \sqrt{\Delta}}{2a}

#### Analyse

Il faut lire trois nombres réels a, b et c, et calculer en fonction de ces trois nombres la valeur de delta. Si delta ≥ 0, on calcule x1 et x2. La bibliothèque *math.h* est nécessaire pour le calcule de x1 et x2 (fonction *sqrt*), et donc, il faut utiliser l’argument « *-lm* » pour compiler le code source (option vu lors du TD n°4).

#### Code source

|  |
| --- |
| #include <stdio.h>  #include <math.h>  /\* DST 1 : prob 1 - eq 2nd degre \*/  int main () {  float a,b,c;  float delta;  float x1, x2;  printf ("Entrez a : ");  scanf ("%f", &a);  printf ("Entrez b : ");  scanf ("%f", &b);  printf ("Entrez c : ");  scanf ("%f", &c);  printf("Solutions pour l'equation: %f x^2 + %f x + %f = 0 : \n",a,b,c);  delta = b\*b - 4\*a\*c;  if (delta<0) {  printf ("Pas de solution reel: delta = %f \n",delta);  }  else {  x1 = (b\*(-1) + sqrt(delta))/(2\*a);  x2 = (b\*(-1) - sqrt(delta))/(2\*a);  printf ("x1 = %f \t x2 = %f \n", x1, x2);  }  } |

### Problème 2 – 100% Algèbre

#### Enoncé

Avec deux entiers positifs a et b, on a effectué les opérations suivantes :

1. L'addition des deux nombres,
2. La soustraction (du plus grand on a retranché le plus petit),
3. La multiplication des deux nombres,
4. La division du plus grand par le plus petit.

La somme de ces quatre résultats a été trouvée égale à 16807.

Ecrire un programme en C capable de trouver les deux nombres initiaux a et b (afficher toutes les solutions possibles).

#### Analyse

On sait que *(a + b) + (a-b) + (a\*b) + (a/b) = 16807* pour deux nombre entier positifs (a, b > 0) avec a > b. A partir de ce constat, on peut déduire que :

* 2a + ab + a/b = 16807

Donc, si on considère *b = 1* (minimal) :

* 2a + a2 + 1 = 16807
* 2a (1 + a) = 16806
* a (1 + a) = 8403

Ainsi, on déduit que *a>0* et *a<8404*.

Il faut donc tester les nombres positifs entre 2 et 8404 dont l’équation 2a + ab + a/b résulte en 16807. Pour cela, deux boucles imbriquées et un teste sont nécessaires.

#### Code source

|  |
| --- |
| #include <stdio.h>  /\* DST : prob 2 100% Algebre  \* pour deux n° entiers a et b, avec a>b et b>0  \* (a + b) + (a-b) + (a\*b) + (a/b) = 16807  \* 2a + ab + a/b = 16807  \* si b = 1  \* 2a + a^2 + 1 = 16807  \* 2a (1 + a) = 16806  \* a (1 + a) = 8403  \* donc a>0 et a<8404  \*/  int main () {  unsigned long a,b, c;  for ( a=2; a<8404 ; a++ ) {  for (b=1; b<a; b++) {  //c = 2a + ab + a/b  c = 2\*a + a\*b + a/b;  if (c == 16807) {  printf ("Trouve! a=%ld et b=%ld\n", a,b);  }  }  }  } |

### Problème 3 – Modulo

#### Enoncé

Si l'on divise le nombre entier X par: 2, 3, 4, 5, 6, 7, 8, 9,10, 11, 12, le reste est toujours égal à 1. Une autre indication pour vos machines X est inférieur à 30000, et ce n'est pas 1.

Ecrire un programme en C capable de découvrir quel est ce nombre.

#### Analyse

On cherche un n° entier *x>0* et *x<30000*, avec *x≠1*, dont le reste de la division entière (modulo) est toujours 1 pour les nombres 2 à 12 :

* *x%2 = x%3 = x%4 = x%5 = x%6 = x%7 = x%8 = x%9 = x%10 = x%11 = x%12 = 1*

Le programme consiste donc à deux boucles imbriqués qui testent la valeur de l’opération modulo pour chaque nombre de 2 à 30000.

#### Code source

|  |
| --- |
| #include <stdio.h>  /\* DST 1: Prob 3 - Modulo  \* etant donne un n° entier x>0  \* x%2 = x%3 = x%4 = x%5 = x%6 = x%7 =  \* x%8 = x%9 = x%10 = x%11 = x%12 = 1  \* x < 30000 et x<>1  \*/  int main () {  unsigned long x = 2;  int j;  int module;  for (x=2; x<30000; x++) {  module = 1;  j = 2;  while (module == 1 && j<=12) {  module = x%j;  j++;  }  if (module == 1) {  printf ("Trouve! X = %ld\n", x);  }  }  short int i=1;  while(i++) for ( j=2; j<=12 && !(i%j-1); ++j,((j==13)?printf("%d\n",i):0));  } |