INF147 : Exercices à réaliser dans la première semaine du cours

Tous ces exercices font partie de la préparation au premier test.

Les premiers exercices traitent des entiers, dans un intervalle, des diviseurs d'un entier ou des chiffres d'un entier (c'est l'apprentissage en écriture de code) ou de traces de processus simples (l'apprentissage en lecture de code) ou du calcul de valeurs réelles spécifiques. Pour ces exercices, vous devez d'abord concevoir le pseudo-code puis après, le code en C.

D'abord, quelques considérations sur la division de 2 variables entières :

- Un entier A divise un entier B si le reste après division de B par A donne 0. L'entier A est alors un diviseur de B et B est un multiple de A.
- Lorsqu'on divise deux entiers, la réponse obtenue est entière. Par exemple 17 / 5 donne 3. La division entière existe et est fondamentale en programmation.
- On obtient le reste de la division de deux entiers avec l'opérateur binaire modulo (l'opérateur %). Ainsi 17 % 5 donne 2.
- Un entier est premier s'il n'a que deux diviseurs, soit 1 et lui-même.
- On obtient le chiffre des unités (chiffre le plus à droite) d'un entier K avec l'expression K %10 (ex. 1234 % 10 = 4).
- On obtient avec l'expression (K / 10). un entier ayant les mêmes chiffres que K moins son chiffre le plus à droite (ex. 1234 / 10 = 123).

#1) Les traces

Tracer un bloc de code c'est donner toutes les valeurs prises par les variables pendant son l'exécution. La valeur actuelle de chaque variable est celle qui apparaît au bas de la colonne portant son nom et vous n'inscrivez une nouvelle valeur au bas d'une colonne que lorsque la variable correspondante est modifiée. Tracez ces 4 blocs de code :

A)

```
{ int  n, p, w;

    n = 5;
    p = 1;
    w = 0;
    while (n > 0){
      p = p * n;
      w = w + p/n;
      n = n - 1;
    }
}
```

n	p	w

B)

```
{ int  a, c;

    a = 537701436;
    c = 0;
    while (a > 0){
      if ((a%10)>5){
        c = c + 1;
      }
      a = a/10;
    }
}
```

C)

```
#define  BORNE  13

{ int  w1, w2, fb;

    w1 = w2 = fb = 1;
    while (w2 < BORNE){
      fb = w1 + w2;
      w1 = 2 * w2;
      w2 = w1 + 1;
    }
}
```

D)

```
{ int n1, n2, p;

    n1 = 210;
    n2 = 15;
    p = 0;
    while (n1 > 0) {
      if ((n1%2) == 1){
        p = p+n2;
      }
      n1 = n1/2;
      n2 = n2*2;
    }
}
```

#2) L'énumération de l'ensemble des candidats à une caractéristique donnée

a) Pour un intervalle de nombres entiers, de la borne inférieure, on incrémente de 1 un compteur jusqu'à ce que le compteur arrive à la borne supérieure. Donc, faites un programme qui demande les deux bornes. Si la borne supérieure est plus grande que la borne inférieure, on fait afficher toutes les valeurs entre ces 2 bornes.

b) On affiche tous les digits d'un entier A à partir de la droite vers la gauche : avec A %10 on obtient le chiffre des unités de A, puis avec A = A/10, on se débarrasse du dernier chiffre observé. On répète tant que A est différent de 0.

c) Trouver et afficher tous les diviseurs d'un entier positif P. Traversez l'intervalle des entiers allant de 1 à P/2 et alors tout entier dont le reste après division avec P donne 0 est un diviseur.

#3) Somme ou produit de valeurs

a) Faire la somme des entiers dans un intervalle. Faites un programme qui demande les deux bornes. Si la borne supérieure est plus grande que la borne inférieure, on affiche la somme de toutes les valeurs entre ces 2 bornes.

b) Dans un intervalle donné, faire la somme des entiers qui sont multiples de 11 OU dont le sinus est supérieur à 0.68467711 (`sin()` fait partie de la librairie <math.h>).

#4) Nombre de diviseurs

a) Compter le nombre de diviseurs d'un entier positif particulier.

b) Faire le produit des diviseurs propres (≠ 1 et ≠ N) d'un entier positif N.

c) Dans un intervalle donné, trouver l'entier ayant un maximum de diviseurs.

#5) Les Entiers premiers

a) Tester si un entier positif N est premier.

b) Compter le nombre de diviseurs premiers d'un entier N.

c) Afficher le prochain nombre premier supérieur à un entier donné.

#6) Les digits (chiffres) d'un nombre

a) Faire la somme des chiffres d'un entier donné : ex. 5067220 donne 22

b) Faire le produit des digits non nuls d'un entier donné : ex. 5067220 donne 840

c) Compter les occurrences d'un digit particulier dans un entier N :
ex. avec N = 5067**22**0 et digit = 2, on obtient 2 occurrences de ce chiffre

d) Donner la valeur du K^{ième} chiffre d'un entier N : ex. avec N = 50**6**7220 et K = 3 → 6

e) Extraire dans un entier K tous les chiffres impairs d'un autre entier N :
ex. avec N = **5**8**1**6**7**22**3**, K sera **5173**

f) Itérer la somme des digits de N jusqu'à l'invariance : N = 156773 → 29 → 11 → 2→ **2**

#7) Les fonctions trigonométriques

On calcul une approximation du sinus avec une série :

a) Approximer le sinus d'un réel donné x en sommant jusqu'à un nombre fixé N de termes de la série de puissances $\sum_{n=0} \frac{(-1)^n x^{2n+1}}{(2n+1)!}$

b) Semblable au précédent, mais cette fois-ci on itère le calcul jusqu'à obtenir un terme dont la valeur absolue sera strictement inférieure à une constante EPSILON = 0.0001