LIF1

Contrôle Continu Terminal (Durée : 2h)
Mardi 7 janvier 2014

Recommandations : *Les documents, calculatrice, téléphone portable sont interdits.*
La qualité de l'écriture et de la présentation sera prise en compte dans la note finale.
Vous veillerez à respecter les notations et les règles d'écriture des algorithmes vues en cours et en TD.
Le barème est donné à titre indicatif.

Partie A : Question de cours (3 pts)

Soit le programme C suivant :

```
#include <iostream>
using namespace std;

void mystere(int a, int b, int &p)
{  p = 0;
  while(b>=1)
   {
          if(b%2==0)
          {a = 2*a; b = b/2;}
   else
    {   p = p+a;
        b = b-1;
    }
   }
 cout << b << endl;
}

int main(void)
{
  int x,y,z;
  x=3; y = 5;
  mystere(x,y,z);
  cout << z <<endl;
  return 0;
}
```

1. Identifiez et nommez
   a. les paramètres formels.
   b. les paramètres effectifs.
   c. les paramètres passés en donnée.
   d. les paramètres en donnée / résultat.
2. Quel sera l'affichage produit ?
3. Que fait la fonction mystere ?

Partie B : Algorithmique (9 pts)

1- Soient motif et repetition deux chaines de caractères de longueur maximum TailleMax.
Ecrire l'algorithme d'une procédure qui remplit la chaine repetition en répétant la chaine motif autant de fois que nécessaires pour que repetition soit totalement rempli.

Exemple :

motif	B	O	N	J	O	U	R	\0										

repetition	B	O	N	J	O	U	R	B	O	N	J	O	U	R	B	O	N	J	O	\0

2- Soit la suite de Syracuse, définie comme suit :
$u_{n+1} = 1 + 3u_n$ si u_n impair ;
$u_{n+1} = u_n / 2$ si u_n pair.

Quel que soit le nombre entier u_0 strictement positif, cette suite finit par engendrer le nombre 1.

a. Écrire l'algorithme d'un sous-programme permettant de saisir une valeur comprise entre 1 et 20 (dimension maximale du tableau déclaré) ; on s'assurera que la valeur saisie appartient bien à l'intervalle voulu.
b. Écrire l'algorithme d'une fonction `syracuse` qui remplit un tableau avec les termes de la suite de `syracuse` précédemment définie à partir d'une valeur `lambda`, donnée par l'utilisateur, et jusqu'à l'obtention de la valeur 1. Cette fonction devra retourner le nombre d'itérations au bout duquel la suite engendre la valeur 1.
   Exemple : Pour `lambda`= 11. Il faudra 14 itérations pour engendrer 1 et le tableau final sera :

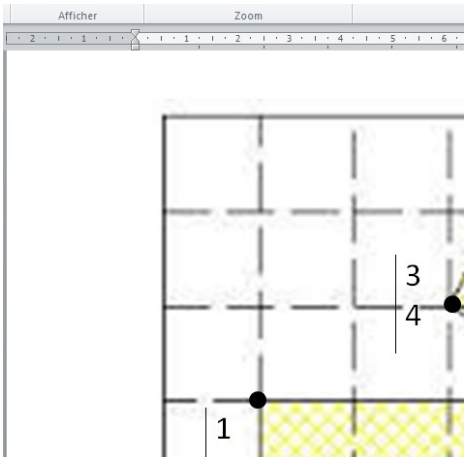
11	34	17	52	26	13	40	20	10	5	16	8	4	2	1					

c. Écrire l'algorithme d'une procédure `affiche_syracuse` permettant l'affichage du tableau obtenu ; attention, on n'affichera que les valeurs significatives du tableau (i.e. celles pour lesquelles le tableau a été rempli).
d. Écrire l'algorithme du programme principal permettant de saisir une valeur `lambda`, d'appeler la fonction `syracuse` et d'afficher le nombre d'itérations nécessaires ainsi que le tableau obtenu.

Partie C : Langage C/C++ (8 pts)

On définit une structure ***point*** permettant de stocker les coordonnées 2D (x et y deux réels) d'un point. On utilise ensuite cette structure pour définir une structure ***quadrilatère*** (constituée de 4 points du plan notés A, B, C, D).

1- Définir en C/C++ les deux structures *point* et *quadrilatere* décrites précédemment.
2- Remplissage des structures
   a. Ecrire en langage C/C++ un sous-programme permettant de remplir la structure *point*.
   b. Ecrire en langage C/C++ un sous-programme permettant de remplir la structure *quadrilatere*. Ce sous-programme devra impérativement faire appel au sous-programme écrit en 2.a.
3- On souhaite écrire un sous-programme permettant de vérifier qu'un quadrilatère passé en paramètre forme un carré. On utilisera la définition suivante du carré : un carré est un quadrilatère ayant ses 4 cotés égaux et ses diagonales de même longueur.
   a. Ecrire en langage C/C++ une fonction permettant à partir des coordonnées de deux points de calculer et retourner la longueur du segment défini. On pourra utiliser les fonctions C/C++ sqrt et pow.
   b. Ecrire en langage C/C++ une fonction booléenne permettant de tester si un quadrilatère représente ou non un carré. On utilisera pour cela la définition précédente ainsi que la fonction écrite en 3.a.



4- Ecrire en C/C++ une procédure permettant de calculer et de "renvoyer" la surface et le périmètre d'un quadrilatère ; attention on ne demande pas d'afficher ces deux valeurs. Si le quadrilatère n'est pas un carré, on ne calculera pas sa surface ni son périmètre, on se contentera de leur affecter la valeur -1.
5- Ecrire en C/ C++ le programme principal permettant de saisir les coordonnées des 4 points constituants un quadrilatère, puis, en utilisant la procédure écrite en 4 d'afficher sa surface et son périmètre.