LIFAP1 – CC mi-parcours – Séquence 3

Contrôle Continu (Durée totale : 1h)
Jeudi 14 novembre 2024

Recommandations : *Les documents, calculatrice, téléphone portable sont interdits. La qualité de l'écriture et de la présentation seront prises 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. Un soin tout particulier devra être apporté à l'écriture des entêtes des différents sous-programmes.*

NOM

.....

PRENOM

.....

Numéro Etudiant

.....

Groupe TD

.....

Vous allez réaliser un programme complet permettant d'afficher tous les nombres premiers inférieurs à une valeur déterminée.

Plusieurs sous-programmes vous seront demandés pour y parvenir. **Vous devrez les écrire soit en langage algorithmique soit en langage C/C++ en vous référant aux consignes qui sont données.**

1- Définir en **C/C++** une constante `MAX` ayant pour valeur `100`.

```
const int MAX =100 ; // ou #define MAX 100
```

2- Ecrire l'**algorithme** d'une **fonction** `SaisieValeur` qui demande à l'utilisateur et retourne une valeur entière comprise entre 1 et 9 inclus. La saisie sera recommencée tant que la valeur n'est pas dans l'intervalle [1 ; 9].

```
Fonction SaisieValeur () : entier
Précondition : aucune
Donnée : aucune
Donnée / résultat : aucun
Résultat : entier
Description : saisit et retourne un entier entre 1 et 9
Variable locale : val : entier
Début
        Faire
            Afficher( "Donnez une valeur entre 1 et 9 ")
            Saisir(val)
        Tant que (val<1 ou val>9)
        Retourne val
Fin
```

3- Ecrire en **C/C++** un sous-programme `RemplitTab` qui remplit un tableau 2D `TabNombres` de taille `MAX*MAX` jusqu'au rang `taille-1` en mettant toutes les valeurs de 1 à `taille*taille` comme dans l'exemple ci-contre où `taille = 5`.

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

TabNombres

```
void remplit_tab (int T[MAX][MAX], int n)
{
   int i,j;
   for (i=0;i<n;i++)
   {
      for (j=0;j<n;j++)
      {
         T[i][j] = i*n+j + 1;
      }
   }

}
```

4- Ecrire l'**algorithme** d'une procédure `AfficheTab` qui affiche un tableau 2D de taille `MAX*MAX` jusqu'au rang `taille-1`.

```
procédure affiche_tab(T tableau[MAX][MAX] d'entiers, Taille entier)
Précondition : 0<Taille<MAX
Donnée : Taille
Donnée / résultat : T
Description : affiche le tableau T
Variable locale : i,j : entiers
Début
   Pour i allant de 0 à Taille par pas de 1 faire
      Pour j allant de 0 à Taille par pas de 1 faire
            Afficher(T[i][j])
      Fin pour
      Afficher (saut de ligne)
    Fin pour
Fin
```

5- Ecrire en **C/C++** un sous-programme `MultiplesDiviseurs` qui compte et "retourne" le nombre de multiples `NbMult` inférieurs ou égaux à une valeur `Val` et le nombre de diviseurs `NbDiv` d'un entier `n` passé en paramètre.

```
void diviseurs_multiples (int n, int val, int &div, int &mult)
{
   int i;
   div = 0;
   mult = 0 ;
   for (i=1;i<=val;i++)
   {
      if (n%i==0) div++;
      else if (i%n== 0) mult++;
   }
}
```

6- En utilisant le sous-programme `MultiplesDiviseurs` de la question précédente, écrire l'**algorithme** d'un sous-programme `RemplitDiviseurs` qui construit le tableau des diviseurs `TabDiv` contenant pour chaque case le nombre de diviseurs comme illustré à droite.

1	2	2	3	2
4	2	4	3	4
2	6	2	4	4
5	2	6	2	6
4	4	2	8	3

TabDiv

Par exemple `9 (TabNombres[1][3]) a 3 diviseurs (TabDiv[1][3])`.

```
procédure remplit_nb_diviseurs (T tableau[MAX][MAX] d'entiers , n entier , TDIV tableau[MAX][MAX] )
d'entiers
Précondition : 0<Taille<MAX
Donnée : n
Donnée / résultat : T, TDIV
Description : remplit TDIV avec les diviseurs des valeurs de T
Variable locale : i,j,mult : entiers
Début
   Pour i allant de 0 à n-1 par pas de 1 faire
      Pour j allant de 0 à n-1 par pas de 1 faire
         diviseurs_multiples(T[i][j],n*n,TDIV[i][j],mult);

      Fin pour
   Fin pour
Fin
```

7- On remarquera qu'un nombre premier a exactement 2 diviseurs (1 et lui-même). Ecrire en **C/C++** le programme principal qui demande une valeur entière comprise entre 1 et 9, construit et affiche le tableau initial `TabNombres`, construit et affiche le tableau de diviseurs `TabDiv` et affiche tous les nombres premiers compris entre `1` et `taille²`. On utilisera les sous-programmes écrits dans les questions précédentes.

```
int  main (void)
{
   int Tab[MAX][MAX], TabDiv[MAX][MAX];
   int n=  saisie_valeur();
   remplit_tab(Tab,n);
   affiche_tab(Tab,n);
   remplit_nb_diviseurs(Tab,n,TabDiv);
   affiche_tab(TabDiv,n);
   int i,j;
   for (i=0;i<n;i++)
   {
      for (j=0;j<n;j++)
      {
         if (TDIV[i][j]==2)
            cout<<T[i][j]<<" est premier "<<endl;
      }
   }
   return 0;
}
```