# LIFAP1 – CC mi-parcours – Séquence 5

**Contrôle Continu** (Durée totale : 1h)

Jeudi 27 octobre 2022

NOM

……

PRENOM

……

Numéro Etudiant

……

Groupe

……

***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.*

## Partie A – Langage C/C++ ( / 14 pts)

**Conjecture de Goldbach** : Tout nombre pair strictement supérieur à 2 est la somme de deux nombres premiers.
Rappel : Un nombre premier n'a pas d'autres diviseurs que 1 et lui-même. Par exemple : 2, 3, 5, 7, 11, 13, ... sont des nombres premiers.

Exemples
24 est la somme de 11 et 13, qui sont des nombres premiers.
16 est la somme de 11 et 5, qui sont des nombres premiers.

1- Ecrire en C/C++ une fonction `saisie_paire` qui renvoie un entier `n` pair strictement supérieur à 2 choisi par l'utilisateur. La saisie devra être recommencée tant que la valeur proposée n'est pas paire et supérieure à 2.

```
int saisie_paire ()
{
   int val;
   do
   {
      cout<<"donnez une valeur strictement positive"<<endl;
      cin>>val;
   } while (val%2 != 0 || val<=0);
   return val;
}
```

2- Ecrire en C/C++ une fonction booléenne `est_premier` qui retourne vrai si le nombre `n` passé en paramètre est **premier**, faux sinon.

```
bool est_premier (int n)
{
   int i;
   for (i=2; i<n-1;i++)
   {
      if (n%i == 0)
         return false;
   }
   return true;
}
```

3- Ecrire en C/C++ un sous-programme `goldbach` qui trouve et "retourne" les deux entiers `n1` et `n2` premiers tels que `n1+n2 = n`.

```
void  conjecture_goldbach (int n, int &n1, int &n2)
{
  int i,j;
  for (i=2; i<n;i++)
  {
     if (est_premier(i))
        for (j=2; j<n ; j++)
        {
           if (est_premier(j))
           {
              if ((i+j)==n)
              {
                 n1=i;
                 n2=j;
              }
           }
        }
  }
}
```

4- Ecrire en C/C++ un sous-programme `tab_goldbach` qui remplit un tableau 2D de taille `2*MAX` avec les décompositions en nombres premiers des `n` premiers nombres pairs à partir de 4.
Exemple pour `n = 12`.

| | 4 = | 6= | 8= | 10= | 12= | 14= | 16= | 18= | 20= | 22= | 24= | 26= |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 2 | 3 | 5 | 7 | 7 | 11 | 13 | 13 | 17 | 19 | 19 | 23 |
| + | 2 | 3 | 3 | 3 | 5 | 3 | 3 | 5 | 3 | 3 | 5 | 3 |

```
void tab_glodbach (int T[2][MAX], int n)
{
   int i, n1, n2, ind=0;
   for (i=0;i<n;i++)
   {
      conjecture_goldbach(2*i+4,n1,n2);
      T[0][ind] = n1;
      T[1][ind] = n2;
      ind++;
   }
}
```

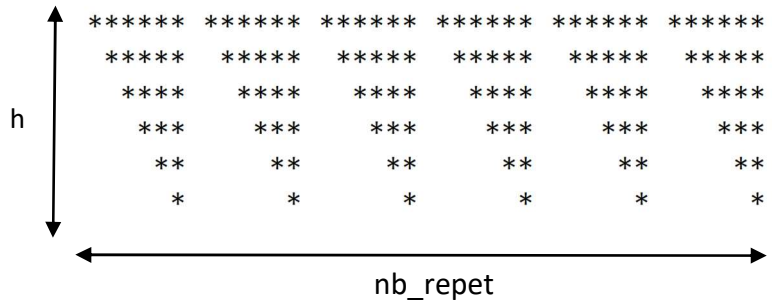
5- On dispose d'une procédure `affiche_tab (int T[2][MAX], int tailleT)` qui permet d'afficher le tableau `T` jusqu'au rang `tailleT`. Ecrire en C/C++ le programme principal qui permet, en utilisant les sous-programmes précédents,

a. de saisir un entier pair strictement supérieur à 2,
b. de remplir un tableau avec les `n` premières décompositions en 2 nombres premiers,
c. et d'afficher le résultat obtenu.

```
int main (void)
{
   int T[2][MAX]= {0};
   int v = saisie_paire();
   tab_glodbach(T, v);
   affiche_tab(T,v);
   return 0;
}
```

# Partie B – Algorithmique ( / 6 pts)

On souhaite écrire un programme permettant d'affiche le motif suivant.




Ecrire l'algorithme d'un sous-programme `dessin` permettant d'afficher ce motif. La hauteur `h` et le nombre de répétitions du motif `nb_repet` sont passés en paramètres. Dans l'exemple `h` = 7 et `nb_repet` = 6.

```
Procedure dessin (h : entier, rep : entier)
Préconditions : h et rep > 0
Données : h, rep
Données / résultat : aucune
Description affiche le motif souhaité
Variables locales : i, j, l : entiers
Début
        Pour i allant de 1 à h par pas de 1 faire
                Pour l allant de 1 à rep par pas de 1 faire
                        Pour j allant de 1 à i par pas de 1 faire
                                Afficher ("_")
                        Fin pour
                        Pour j allant de 1 à h-i par pas de 1 faire
                                Afficher ("*")
                        Fin pour
                Fin pour
                Afficher saut de ligne
        Fin pour
Fin
```