# LIFAP1 – CC mi-parcours – Séquence 1

**Contrôle Continu** (Durée totale : 1h)
Lundi 19 octobre 2020

NOM :

CORRIGE et BAREME

PRENOM :

……

Numéro Etudiant :

……

***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 – Algorithmique

1. Ecrire l'algorithme d'un sous-programme `saisie_valeur` qui demande à l'utilisateur une valeur **strictement positive et impaire** et la retourne. On recommencera la saisie tant que la valeur proposée ne respecte pas ces contraintes.

3 points

```
fonction saisie_valeur() : entier
préconditions : aucun
donnée : aucune                                              entête :  1 point
donnée / résultat : aucune
résultat : entier
description : saisit et retourne une valeur strictement positive et impaire
variable locale : val : entier

début
        faire
                afficher (« Donnez un entier paire strictement positif »)
                saisir (val)                                 saisie : 0,5 point
        tant que ((val<=0) ou (val modulo 2 = 0))            boucle : 1 point
        retourner val                                        retour : 0.5 pt
fin
```

2. Ecrire l'algorithme d'un sous-programme `dessine_triangle` qui permet d'afficher le motif ci-contre. La hauteur (une valeur strictement positive et impaire) ainsi que les caractères seront passés en paramètres du sous-programme.
On pourra utiliser `afficher(saut de ligne)` pour passer à la ligne suivante. Dans l'exemple ci-contre on aura : dessine_triangle(7,'@','*')

```
@@@@@@@
*******
@@@@@
*****
@@@
***
@
```

4.5 points

```
procédure dessine_triangle (h : entier, c1 : caractère, c2 : caractère)
préconditions : aucun
donnée : h, c1, c2                                          entête :  1 point
donnée / résultat : aucune
description : affiche le motif
variable locale : i, j : entier

début
        pour i allant de 1 à h par pas de 1 faire
                pour j allant de 1 à h-i+1 par pas de 1 faire    double boucle : 2 points
                        si (i modulo 2 = 1) alors                test : 1 point
                                afficher (c1)
                        sinon afficher (c2)                      affichage : 0.5 point
                        fin si
                fin pour
                afficher (saut de ligne)
        fin pour
fin
```

3. En utilisant les sous-programmes écrits en 1- et 2-, écrire l'algorithme du programme principal qui affiche un triangle avec des caractéristiques choisies par l'utilisateur.

```
Programme principal                                          2.5 points
Variables :
        hauteur : entier
        car_pair, car_impair : caractères
début
        hauteur ← saisie_valeur ()                           appel saisie_valeur : 1 point
        afficher ("Donnez deux caractères")
        saisir (car_pair, car_impair)                        saisie des caractères : 0.5 point
        dessine_triangle (hauteur,car_impaire, car_pair)     appel dessin : 1 point
fin
```

## Partie B – Langage C/C++

1. Ecrire en langage C/C++ une **procédure** `som_prod_proc` permettant à partir d'un entier `n` passé en paramètre de calculer et "retourner" au programme principal la somme des chiffres pairs et le produit des chiffres impairs qui composent ce nombre `n`. Exemple : si n = 14071789 on devra obtenir somme = 12 (=4+0+8) et produit = 441 (=1*7*1*7*9)

```
                                                    5.5 points
void som_prod_proc (int n, int & som, int &prod)    entête : 1 point
{
   int chiffre;
   som = 0 ;
   prod = 1;                                        initialisation : 1 point
   while (n!= 0)                                    boucle :  1 point
   {
     chiffre = n % 10;
     if (chiffre % 2 == 0)                          test : 1 point
          som += chiffre ;                          calculs : 1 point
     else prod *= chiffre ;
     n/=10;                                         itération suivante : 0.5 point
   }
}
```

2. Transformez l'entête et uniquement l'entête de la procédure précédente en fonction C/C++ que vous nommerez `som_prod_fct`. Cette fonction devra être capable de transmettre les mêmes informations que la procédure précédente.

```
int som_prod_fct (int n, int &prod)                    1.5 points
```

3. Ecrire en langage C/C++ le programme principal permettant :
   a. de demander un nombre à l'utilisateur et de le saisir,
   b. d'afficher les deux valeurs obtenues en utilisant la procédure `som_prod_proc`,
   c. puis d'afficher les deux valeurs obtenues en utilisant la procédure `som_prod_fct`.

```
                                                       3 points
int main(void)                                         structure générale : 0.5 point
{
   int s,p, val;                                       déclaration variables : 0.5 point
   cout<<"Donnez la valeur à tester"<<endl;
   cin>>val;
   som_prod_proc(val,s,p);                             appel + affichage proc : 1 point
   cout<<"Calcul avec la procédure => La somme est : "<<s<<" et le produit : "<<p<<endl;
   s = som_prod_fct(val,p);                            appel + affichage fct : 1 point
   cout<<"Calcul avec la fonction =>La somme est : "<<s<<" et le produit : "<<p<<endl;
   return 0;
}
```