LIFAP1 – Partie B – Langage C/C++

Contrôle Continu Terminal (Durée totale : 2h)
Lundi 14 décembre 2020

NOM :

……

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 programmes vues en cours et en TP.*

Restons en forme !!!

On souhaite faire quelques statistiques avec les informations fournies par les bracelets connectés des membres d'une famille.

1. Définir en C/C++ 2 constantes `CHMAX` (qui sera utilisée pour les chaines de caractères) et `MAX_MONTRES` ayant pour valeurs respectives 64 et 8.

```
const int CHMAX = 64;
const int MAX_MONTRES = 8;
```

2. Une `montre` est définie par son `proprietaire`, sa `marque` et un tableau `info` (3 lignes, 7 colonnes) contenant pour chaque jour de la semaine le nombre d'étages montés, le nombre de pas effectués et la fréquence cardiaque moyenne (uniquement des valeurs entières). En vous aidant de l'exemple ci-contre, définir en C/C++ la structure `montre`.

propriétaire : Alfred
marque : speed
info

12	15	6	7	11	12	24
7554	12342	5675	8745	9965	14387	17786
72	56	89	120	67	64	87

```
struct montre
{
    char marque[CHMAX],a_qui[CHMAX];
    int tab[3][7];
};
```

3. Écrire en langage C/C++ une **fonction** `remplir_montre` permettant de remplir une structure `montre` avec des informations choisies par l'utilisateur. Attention, la saisie de chaque information contenue dans le tableau `info` devra être recommencée tant que la valeur n'est pas strictement positive.

```
struct montre remplir_montre()
{
    struct montre m;
    int i,j;
    cout<<"Marque de la montre : "<<endl;
    cin>>m.marque;
    cout<<"a qui ?"<<endl;
    cin>>m.a_qui;
    for(i=0;i<3;i++)
    {
        for (j=0;j<7;j++)
        {
            cin>>m.tab[i][j] ;
        }
    }
    return m;
```

4. Ecrire en C/C++ une procédure `affiche_montre` permettant d'afficher toutes les caractéristiques d'une `montre` passée en paramètre.

```
void afficher_montre(struct montre m)
{

    int i,j;
    cout<<"Marque de la montre : "<<endl;
    cout<<m.marque;
    cout<<"a qui ?"<<endl;
    cout<<m.a_qui;
    for(i=0;i<3;i++)
    {
        for (j=0;j<7;j++)
        {
            cout<<m.tab[i][j] <<" ";
        }
        cout<<endl;
    }
}
```

Dans la famille, chacun des membres possède sa propre montre connectée et voudrait comparer des données avec celles des autres personnes du foyer.

5. Définir en C/C++ la structure `famille` contenant le nombre de `montres` que la famille possède `nbmontres`, ainsi qu'un tableau de `montre` `tab_montres` ayant au maximum `MAX_MONTRES` cases.

```
struct famille
{
    int nb;
    struct montre tab[MAX_MONTRE];
};
```

6. Écrire en langage C/C++ une **procédure** `ajoute_une_montre` permettant d'ajouter une `montre` à la `famille` s'il reste encore de la place. On fera appel à la fonction écrite en 3.

```
void ajoute_1_montre (struct famille &f)
{
    if (f.nb<MAX_MONTRE)
    {
        f.tab[f.nb] = remplir_montre();
        f.nb++;
    }
    else cout<<"tableau plein"<<endl;
}
```

7. Écrire en langage C/C++ une procédure `moyennes` permettant, **à partir d'un tableau 1D de 7 entiers** `tab` de calculer et "retourner" `moy_tot` la moyenne de toutes les valeurs du tableau et `moy_part` la moyenne partielle obtenue en enlevant les deux valeurs extrêmes (minimum et maximum) du calcul.

```
void moyennes (int T[7], float &moy, float &moy_partielle)
{
    int mini, maxi;
    int i;
    mini = T[0];
    maxi = T[0];
    moy = T[0];
    for (i=1;i<7;i++)
    {
        if (mini<T[i]) mini = T[i];
        if (maxi>T[i]) maxi = T[i];
        moy +=T[i];
    }
    moy_partielle = (moy - mini - maxi) /5.0;
    moy = moy / 7.0 ;
}
```

8. Écrire en langage C/C++ une procédure `calcul_moyenne_montre` permettant à partir d'une `montre` passée en paramètre de remplir et "retourner" un tableau `Tab_Moy` de taille 3*2 contenant les moyennes totales et partielles pour chacune des données présentes dans le tableau d'informations de la `montre`.

	Moyenne Totale	Moyenne Partielle
étages	12,43	11,4
pas	10922	10598,6
fréquence	79,29	75,8

```
void calcul_moyennes_montre (montre m, float Tmoy[3][2])
{
    int i;
    for (i=0;i<3;i++)
    {
        moyennes(m.tab[i],Tmoy[i][0], Tmoy[i][1]);
    }
}
```

9. Écrire en langage C/C++ un sous-programme `affiche_tab_moy` permettant d'afficher le contenu du tableau obtenu à la question précédente.

```
void affiche_Tab_moyenne_montre (float Tmoy[3][2] )
{
    int i,j;
    for (i=0;i<3;i++)
    {
        for (j=0;j<2;j++)
            {
                cout<<Tmoy[i][j]<<" ";
            }
            cout<<endl;
    }
}
```

10. En utilisant les sous-programmes des questions 4, 8 et 9, écrire en langage C/C++ le sous-programme `affiche_perf` permettant d'afficher les informations d'une `montre` ainsi que le tableau de moyennes `tab_moy` d'un membre de la famille dont le `nom` est passé en paramètre.

```
void  affiche_perf (famille f, char nom[CHMAX])
{
    int i;
    float res[3][2];
    for (i=0;i<f.nb ; i++)
    {
        if (strcmp (nom, f.tab[i].a_qui)==0)
        {
            afficher_montre(f.tab[i]);
            calcul_moyennes_montre(f.tab[i],res);
            affiche_Tab_moyenne_montre(res);
        }
    }
}
```

11. Écrire en langage C/C++ le programme principal qui permet
    a. de saisir autant de `montre` que l'utilisateur le voudra pour la `famille` (on lui demandera à chaque saisie s'il souhaite ou non continuer),
    b. et d'afficher les performances d'un membre dont le `nom` sera choisi par l'utilisateur.

```
int main (void)
 {
     struct famille f;
     char qui[CHMAX];
     f.nb = 0;
     char rep ;
     do
     {
         ajoute_1_montre(f);
         cout<<"encore ? "<<endl;
         cin>>rep;
     } while (rep=='o');
    cout<<"vous voulez afficher les infos pour qui ?"<<endl;
    cin>>qui;
    affiche_perf(f,qui);
     return 0;
 }
```