Correction Examen du module Programmation 2

Exercice 1

1) Écrire une fonction qui retourne l'indice de la première occurrence du plus petit élément et l'indice de la première occurrence du plus grand élément d'un tableau de réel en utilisant **le formalisme pointeur pour parcourir le tableau**.

```
void ind_MIN_MAX(int  *tab,int n, int  *indice_min,int *indice_max){
   int *p;
   indice_min=0,indice_max=0;
  for(p=tab; p<tab+n;p++){
     if(*(tab+*indice_min)>*p) *indice_min=p-tab;
     if(*(tab+*indice_max)<*p) *indice_max=p-tab;}
    // printf("minndice=%d, maxindice=%d",minindice,maxindice);
  }
```

2) Définir la fonction **int nombre_espaces(char *s)** qui renvoie le nombre de caractères espace ' ' présents dans la chaîne s

```
int nombre_espaces_iter(char *str){

        char *temp=str;
        int cpt=0;

        while(strchr(temp,' ')){
                cpt++;
                temp=strchr(temp,' ')+1;
        }

        return cpt;
}
```

Ecrire la fonction **char **les_mots(char *s)** qui extrait de la chaîne s les mots qui la constituent et les place dans un tableau de chaînes de caractères. **Indication** : utiliser les fonctions prédéfinis **strchr**, **strcpy** et **strncpy** de **<string.h>**.

```
char **les_mots(char *s){

        char **tab, *str=s;
        int n=nombre_espaces_rec(s)+1;

        tab=(char**)malloc((n+1)*sizeof(char*));

        n=0; s=strchr(str,' ');

        while(s){
                tab[n]=(char*)malloc(s-str+1);
                strncpy(tab[n],str,s-str);
                tab[n][s-str]='\0';
                s++;
                str=s;
                s=strchr(str,' ');
                n++;
        }
        tab[n]=(char*)malloc(strlen(str)+1);
        strcpy(tab[n],str);

        tab[n+1]=NULL;

        return tab;
}
```

3) On représente un polynôme par un tableau. Les indices du tableau représentent les puissances et les éléments du tableau représentent les facteurs du polynôme. Ecrire une fonction **float* produit_poly(float *P, int n, float *Q, int m),** qui calcul le produit de deux polynômes et renvoi le polynôme résultat.

```
float* produit_poly(float *P ,int n, float *Q, int m)

{
float *R = (float *)malloc((n+m) *siezof(float));
int I,J;

for (I=0;I<n+1;I++)
{
for (J=0;J<M+1;J++)
{
R[I+J]+=P[I]* Q[J];
}
}
return(R);
}
```

4) Dire ce que calcule la fonction f selon les différentes valeurs des arguments x et y :

```
int f(int x, int y){
        if(x==0) return y ;
        else return( f(x-1,x+y)) ;
}
```

Si x=0 : y affirmation incluse dans la suivante

(Si x ≥ 0) si x > 0 : $\sum_{i=1}^{i=x} i + y$

Si x < 0 ne termine jamais

Exercise 2

1) Créer un nouveau type **DateNaissance** qui inclut le jour/ le moi/ l'année.

```
typedef struct {
   int j,m,a;
}DateNaissance;
```

2) Créer un nouveau type **Participant** qui inclut son nom (chaîne de caractères : tableau de 20 caractères), son prénom (chaîne de caractères : tableau de 30 caractères), date de naissance ainsi que toutes les autres informations nécessaires à son inscription selon les critères définis ci-dessus. On privilégiera une structure contenant un nombre minimal de champs.

```
typedef struct {
   char nom[20];
   char prenom[30];
   int dejeuner;// 0=non 1=oui
   int diner;// 0=non 1=oui
   int hotel;// 1=pas d'hotel 2=2etoiles 3=3etoiles
   int seul;// 0=non 1=oui
   DateNaissance d;
}Participant;
```

3) Créer un nouveau type **TabPart** qui est un tableau de capacité maximale définie par une constante max donnée (max =100) de type Participant.

```
Participant TabPart[100];
```

4) Ecrire une fonction **Nb2Etoiles** qui, pour un argument de type **TabPart** donné, affiche le nom, le prénom et la date de naissance des personnes qui ont choisi de réserver un hôtel 2 étoiles.

```
void Nb2Etoiles(TabPart tab){
   int i;
   for(i=0;i <100;i++)
    if(tab[i].hotel==2)
    printf("nom: %s, prenom:%s, date de naissance:
%d/%d/%d",tab[i].nom,tab[i].prenom,tab[i].d.j,tab[i].d.m,tab[i].d.a);
}
```

5) Ecrire une fonction **NbDej_Din** qui, pour un participant donné, retourne le nombre de déjeuner et le nombre de diner à prévoir.

```
int NbDej(TabPart tab,int *nbdej,in *nbdi){
   nbdej=nbdi=0;
int i;
   for(i=0; i<100;i++){
    if(tab[i].dejeuner==1) nbdej++;
    if(tab[i].diner==1) nbdi++;}
}
```

6) Ecrire une fonction **Montant** qui calcule, pour un Participant donné en argument, le montant de sa facture.

```
int Montant(Participant p){
   int total=0;
   if(p.dejeuner==1)total+=150;
   if(p.diner==1)total+=350;
   if(p.hotel==2)total+=750;
   if(p.hotel==3)total+=100;
   return total;
}
```

7) Ecrire une fonction **Max_min** qui pour un argument de type **TabPart** donné, retourne les informations du participant ayant le montant de la facture maximale et les informations du participant ayant le montant de la facture minimale.

```
Void Max_min( TabPart tab, Participant *pmax, Participant *pmin){
 int i;
 pmax=tab;
 pmin=tab;
    for(i=0;i <100;i++){
     if(Montant (tab[i]) >Montant(* pamax)) pmax=tab+i;
     if(Montant (tab[i]) < Montant(* pmin) )pmin=tab+i;
}
}
```