Série N° 2. Algorithmique et Programmation II **Module : I132 (Section 1)**

Exercice 1.

La fonction exponentielle est définie comme suit :

$$\text{Exp(x)} = 1 + \frac{x}{1!} + \frac{x^2}{2!} + ... + \frac{x^n}{n!} + ...$$

Ecrire un programme permettant de calculer, pour une valeur x donnée du type float et n de type entier, une approximation de exp (x) obtenue avec les n premiers termes de la séquence.

Exercice 2.

Donnez un programme qui demande à l'utilisateur un tableau d'entiers de taille n et qui le modifie de telle sorte que tous les entiers pairs se retrouvent avant les entiers impairs.

Exemple :

Au départ : **T = 7 4 7 8 4 6 3 9 6**

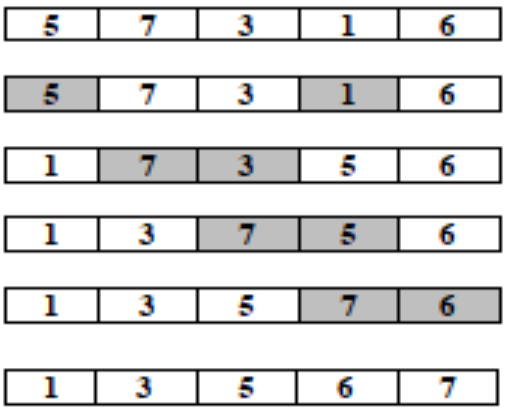
Après modification : **T = 4 8 4 6 6 7 3 9 7**

Exercice 3. *Tri par Sélection*

Le principe est de sélectionner l'élément le plus petit du tableau, c'est à dire de trouver l'entier p tel que ∀ i, T[i] ≥ T[p]. Une fois cet emplacement trouvé, on échange les éléments T[0] et T[p]. Puis on recommence ces opérations sur le reste du tableau (pour les éléments compris entre les indices 1 et n-1). On recherche alors le plus petit élément de cette nouvelle suite de nombre et on l'échange avec T[1]. Et ainsi de suite . . . jusqu'au moment où l'on a placé tous les éléments du tableau.

Donnez un programme de tri par sélection du plus petit élément.

Exemple: Soit le tableau suivant, les cases grisées doivent être permutées.

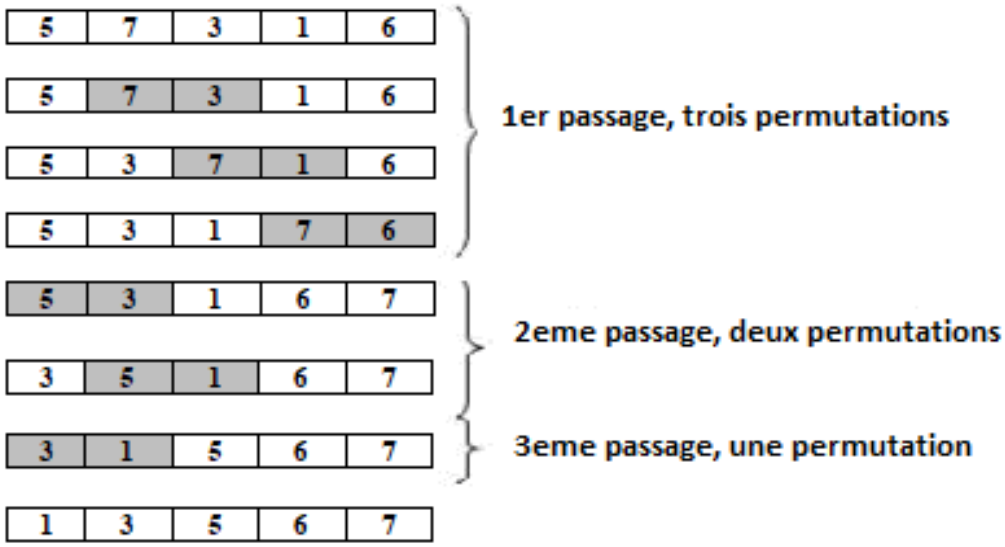


Exercice 4. *Tri Bulle*

Cette méthode consiste à parcourir le tableau à trier et à comparer chaque élément avec son voisin de droite. Lorsque deux éléments consécutifs ne sont pas dans l'ordre croissant, ils sont échangés. Après un parcours complet du tableau, on recommence l'opération. Lorsqu'un parcours n'a donné lieu à aucun échange, cela signifie que le tableau est trié : le processus s'arrête.

Donner un programme c permettant d'ordonner les éléments selon ce mécanisme.

Exemple:

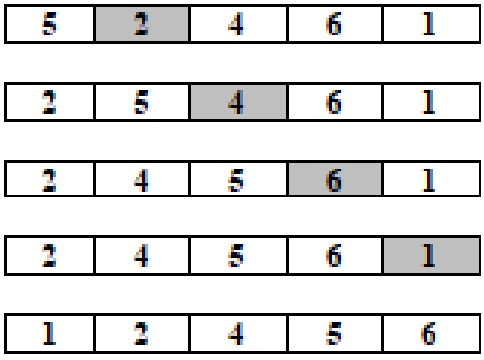


Exercice 5. *Tri par Insertion*

Cette méthode consiste à traiter une à une les valeurs du tableau et à les insérer, au bon endroit, dans le sous-tableau trié constitué des valeurs précédemment triées. Les valeurs sont piochées dans l'ordre où elles apparaissent dans le tableau. Soit i l'indice de la valeur piochée, les (i − 1) premières valeurs du tableau constituent le sous-tableau trié dans lequel va être inséré la i-ème valeur. Au début de l'algorithme, il faut considérer que le sous-tableau constitué du seul premier élément est trié. Ensuite, on insère le second élément (i = 2), puis le troisième (i = 3) . . .etc. Ainsi, i varie de 2 à n, où n est le nombre total d'éléments du tableau.

Donner un programme permettant d'ordonner les éléments selon ce mécanisme.

Exemple:



Solution

Exercice 1.

$$\text{Exp(x)} = 1 + \frac{x}{1!} + \frac{x^2}{2!} + ... + \frac{x^n}{n!} + ...$$

$$\Rightarrow \exp(x)=\sum(x^i)/(i\ !)\ ;\ i=0,1,2,..\ n$$

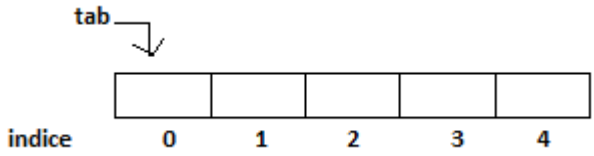
```
#include <stdio.h>
int main()
{
    int  n, i, j, F;
    float S, x, P ;
/* ****************************** la saisie des données************************** */
printf ("tapez un entier \n");
scanf ("%d", &n);
printf ("tapez le parametre x \n");
scanf ("%f", &x);
/* ********************************** Traitement ******************************** */
for ( i =0, S=0; i<=n; i = i + 1)
{
  P=1;                  // ************** puiss(x,i)
for ( j =1; j<= i; j++)
    P=P*x;
F=1;                    // ************ fact(i)
for ( j =1; j<= i; j++)
    F=F*j;
S=S+ (P/F);
}
/* ***************** affichage du résultat ************************************ */
printf ("après le traitement : \n");
printf ("exp(%f)=%f ", x, S);
return 0;
}
```

Exercice 2.

Rappel:

Tableau= ensemble de cases mémoires

- **même type**
- **même nom**
- **chaque élément à un indice**



Tab= tableau de n éléments

Accès aux composantes

Tab[0] 1ere case

Tab[1] 2eme case

Tab[2] 3eme case

...

Tab[n-1] dernière case

Parcours de tableau:

for(i=0 ;i<n ;i++) ➔ parcourir le tableau

Exemple :

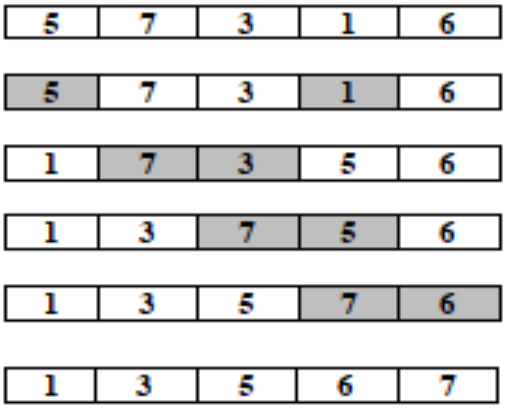
Au départ : **tab = 7 4 7 8 4 6 3 9 6**

Après modification : **tab = 4 8 4 6 6 7 3 9 7**

```
#include <stdio.h>
/* lecture d'un tableau et affichage des pairs d'abord, les impairs ensuite */
int main()
{
    int tab[100];
    int i, j, X, N ;
```

```
/* ********************************** la saisie des données********************************** */
printf ("tapez le nombre d'elements \n");
scanf("%d", &N);
printf ("tapez les elements \n");
for ( i = 0; i < N; i++ )
{
printf ("tab[%d]= ", i);
scanf("%d", &tab[i]);
}
/* ******************* affichage du tableau dans l 'ordre ********************************** */
printf ("au depart : \n");
for ( i = 0; i< N;i = i + 1)
{
printf ("%d ", tab[i ]);
}
printf ("\n");
/* ********************************** Traitement ************************************************ */
for ( i =0; i< N-1 ; i = i + 1)
{
   if (tab[ i ]%2 != 0)   // nombre impair à la position i
    {
     j =i+1;
    while (j< N) && (tab[ j ]%2 != 0))
          j++ ;
    if(j<N)   // nombre pair à la position j
              {
              X=tab[i] ;
              tab[i]=tab[j] ;
              tab[j]=X ;
              }
     }
}
/* ***************** affichage du tableau après le traitement ************** */
printf ("après modification : \n");
for ( i = 0; i< N;i = i + 1)
    printf ("%d   ", tab[i ]);
return 0;
}
```

Exercice 3.

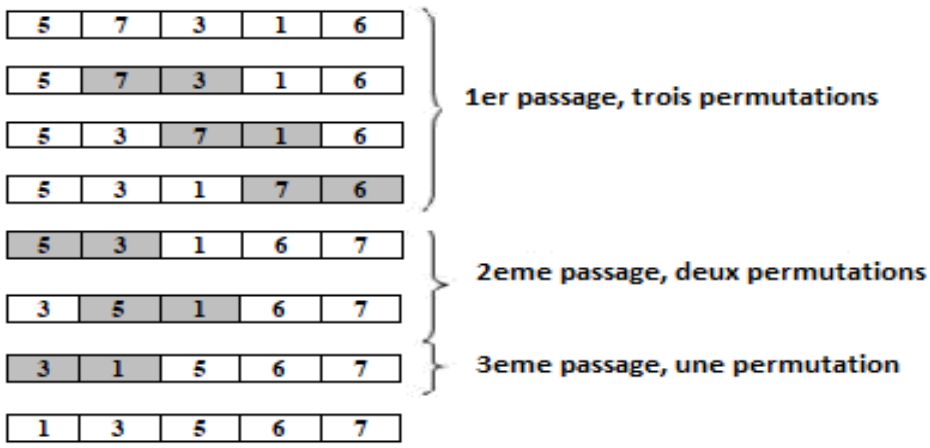



```
#include <stdio.h>
int main()
{
    int tab [100];
    int i, j, N, pos, tmp ;
/* ********************************** la saisie des données****************************** */
printf ("tapez le nombre d'elements \n");
scanf("%d", &N);

printf ("tapez les elements \n");
for ( i = 0; i < N; i++)
{
printf ("tab[%d]= ", i);
scanf("%d", &tab[i]);
}
/************************************** Traitement ************************************ */
for ( i = 0; i < N-1; i ++)
{
   pos=i+1 ;
     for ( j = i+1; j < N; j ++)
      {
        if(tab[j]< tab[pos])
              pos=j ;
      }
```

```
        if(tab[pos]<tab[i])
                {
                        tmp=tab[i] ;
                        tab[i]=tab[pos] ;
                        tab[pos]=tmp ;
                }
    }
/* ********************** affichage du tableau après le traitement ************************ */
for ( i = 0; i< N;i = i + 1)
    printf ("%d ", tab[i]);
return 0;
}
```

Exercice 4.

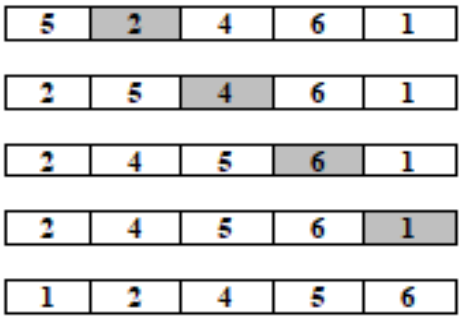



```
#include < stdio.h>
int main()
{   int tab[100];
    int i, j,  N, tmp, trouve;
/* ****************************** la saisie des données************************** */
printf ("tapez le nombre d'elements \n");
scanf("%d", &N);
printf ("tapez les elements \n");
for ( i = 0; i < N; i++)
{   printf ("tab[%d]= ", i);
    scanf("%d", &tab[i]);
}
```

```
/* ************************************** Traitement ********************************** */
do
      {
        trouve=0 ;
         for ( i = 0; i < N-1; i ++){
              If(tab[i]> tab[i+1])
                        {
                       tmp=tab[i] ;
                       tab[i]=tab[i+1] ;
                       tab[i+1]=tmp ;
                        trouve=1 ;
                      }
             }
}  while (trouve ! =0) ;

/* ********************* affichage du tableau après le traitement ********************** */
for ( i = 0; i< N; i++)
printf ("%d ", tab[i ]);
return 0;
}
```

Exercice 5.




```
#include < stdio.h>
int main()
{
int tab[100];
int i, j, v;
/* ********************************  la saisie des données ****************************** */
printf ("tapez le nombre d'elements \n");
scanf("%d", &N);
```

```
printf ("tapez les elements \n");
for ( i = 0; i < N; i = i + 1)
{
    printf ("tab[%d]= ", i);
    scanf("%d", &tab[i]);
}
/* ********************************************** Traitement ********************************** */
for ( i = 1; i < N; i ++)
{
    v= tab[i] ; // conserver la valeur à insérer
// **************  Décaler toutes les valeurs supérieures à droite **********************/
    j=i ;
    while (tab[j-1]> v)
     {
        tab[j]=tab[j-1] ;
         j-- ;
     }
/*************************************************************************************************/
tab[i]=v ; // insérer la valeur à la position convenable
}
/* ***************** affichage du tableau après le traitement ************** */
for ( i = 0; i< N;i = i + 1)
       printf ("%d ", tab[i ]);
return 0;
}
```