2014/202015
SMI-S3
Travaux Dirigés de Langage C
Série N°5

Exercice 1 : *Maximum et minimum des valeurs d'un tableau*

Ecrire un programme qui détermine la plus grande et la plus petite valeur dans un tableau d'entiers A. Afficher ensuite la valeur et la position du maximum et du minimum. Si le tableau contient plusieurs maxima ou minima, le programme retiendra la position du premier maximum ou minimum rencontré.

Exercice 2

Un tableau A de dimension N+1 contient N valeurs entières triées par ordre croissant; la $(N+1)^{ième}$ valeur est indéfinie. Insérer une valeur VAL donnée au clavier dans le tableau A de manière à obtenir un tableau de N+1 valeurs triées.

Exercice 3

```
main()  {
    int A = 1;
    int B = 2;
    int C = 3;
    int *P1, *P2;
    P1=&A;
    P2=&C;
    *P1=(*P2)++;
    P1=P2;
    P2=&B;
    *P1-=*P2;
    ++*P2;
    *P1*=*P2;
    A=++*P2**P1;
    P1=&A;
    *P2=*P1/=*P2;
    return 0;    }
```

Copiez le tableau suivant et complétez-le pour chaque instruction du programme ci-dessus..

	A	B	C	P1	P2
Init.	1	2	3	/	/
P1=&A	1	2	3	&A	/
P2=&C					
*P1=(*P2)++					
P1=P2					
P2=&B					
*P1-=*P2					
++*P2					
P1=*P2					
A=++*P2**P1					
P1=&A					
*P2=*P1/=*P2					

Exercice 4

Soit P un pointeur qui 'pointe' sur un tableau A:

```
int A[] = {12, 23, 34, 45, 56, 67, 78, 89, 90};
int *P;
P = A;
```

Quelles valeurs ou adresses fournissent ces expressions:

a) *P+2
b) *(P+2)
c) &P+1
d) &A[4]-3
e) A+3
f) &A[7]-P
g) P+(*P-10)
h) *(P+*(P+8)-A[7])

Exercice 5 : Ecrire un programme qui lit deux tableaux A et B et leurs dimensions N et M au clavier et qui ajoute les éléments de B à la fin de A. Utiliser le formalisme pointeur à chaque fois que cela est possible.

Exercice 6 : Pourquoi les créateurs du standard ANSI-C ont-ils décidé de légaliser les pointeurs sur le premier élément derrière un tableau? Donner un exemple.

Exercice 7 : Ecrire un programme qui lit un entier X et un tableau A du type **int** au clavier et élimine toutes les occurrences de X dans A en tassant les éléments restants. Le programme utilisera les pointeurs P1 et P2 pour parcourir le tableau.

Exercice 5.8 Fusion de deux tableaux triés

Problème: On dispose de deux tableaux A et B (de dimensions respectives N et M), triés par ordre croissant. Fusionner les éléments de A et B dans un troisième tableau FUS trié par ordre croissant.

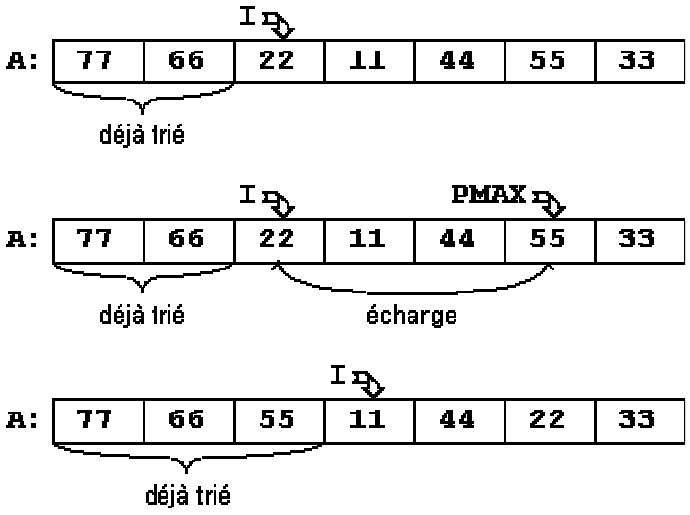
Méthode: Utiliser trois indices IA, IB et IFUS. Comparer A[IA] et B[IB]; remplacer FUS[IFUS] par le plus petit des deux éléments; avancer dans le tableau FUS et dans le tableau qui a contribué son élément. Lorsque l'un des deux tableaux A ou B est épuisé, il suffit de recopier les éléments restants de l'autre tableau dans le tableau FUS.

Exercice 5.9 Tri par sélection du maximum

Problème: Classer les éléments d'un tableau A par ordre décroissant.

Méthode: Parcourir le tableau de gauche à droite à l'aide de l'indice I. Pour chaque élément A[I] du tableau, déterminer la position PMAX du (premier) maximum à droite de A[I] et échanger A[I] et A[PMAX].

Exemple:



2014/202015
SMI-S3
Corrigé des Travaux Dirigés de Langage C
Série N°5

Solution 1

```
#include <stdio.h>
main() {
int A[50];              /* tableau donné  */
 int N;                 /* dimension      */
 int I;                 /* indice courant */
 int MIN;               /* position du minimum */
 int MAX;               /* position du maximum */
/* Saisie des données */
 printf("Dimension du tableau (max.50) : ");
 scanf("%d", &N );
 for (I=0; I<N; I++)
    {
     printf("Elément %d : ", I);
     scanf("%d", &A[I]);
    }
 /* Affichage du tableau */
 printf("Tableau donné :\n");
 for (I=0; I<N; I++)
     printf("%d ", A[I]);
 printf("\n");
 /* Recherche du maximum et du minimum */
 MIN=0;
 MAX=0;
 for (I=0; I<N; I++)
     {
      if(A[I]>A[MAX])
                MAX=I;
      if(A[I]<A[MIN])
                MIN=I;
     }
  /* Edition du résultat */
 printf("Position du minimum : %d\n", MIN);
 printf("Position du maximum : %d\n", MAX);
 printf("Valeur   du minimum : %d\n", A[MIN]);
 printf("Valeur   du maximum : %d\n", A[MAX]);
 return 0;
 }
```

Solution 2

```
#include <stdio.h>
main()
{
 /* Déclarations */
 int A[50]; /* tableau donné    */
 int VAL;   /* valeur à insérer */
 int N;     /* dimension        */
 int I;     /* indice courant   */
 /* Saisie des données */
 printf("Dimension N du tableau initial (max.50) : ");
```

```
scanf("%d", &N );
for (I=0; I<N; I++)
   {
    printf("Elément %d : ", I);
    scanf("%d", &A[I]);
   }
printf("Elément à insérer : ");
scanf("%d", &VAL );
/* Affichage du tableau */
printf("Tableau donné : \n");
for (I=0; I<N; I++)
   printf("%d ", A[I]);
printf("\n");
/* Déplacer les éléments plus grands que */
/* VAL d'une position vers l'arrière.    */
for (I=N ; (I>0)&&(A[I-1]>VAL) ; I--)
      A[I]=A[I-1];
/* VAL est copié à la position du dernier */
/* élément déplacé. */
A[I]=VAL;
/* Nouvelle dimension du tableau ! */
N++;
 /* Edition des résultats */
printf("Tableau résultat :\n");
for (I=0; I<N; I++)
   printf("%d ", A[I]);
printf("\n");
return 0;
}
```

Solution 3

	A	B	C	P1	P2
Init.	**1**	**2**	**3**	**/**	**/**
P1=&A	**1**	**2**	**3**	**&A**	**/**
P2=&C	**1**	**2**	**3**	**&A**	**&C**
***P1=(*P2)++**	**3**	**2**	**4**	**&A**	**&C**
P1=P2	**3**	**2**	**4**	**&C**	**&C**
P2=&B	**3**	**2**	**4**	**&C**	**&B**
***P1-=*P2**	**3**	**2**	**2**	**&C**	**&B**
++*P2	**3**	**3**	**2**	**&C**	**&B**
***P1*=*P2**	**3**	**3**	**6**	**&C**	**&B**
A=++*P2P1**	**24**	**4**	**6**	**&C**	**&B**
P1=&A	**24**	**4**	**6**	**&A**	**&B**
***P2=*P1/=*P2**	**6**	**6**	**6**	**&A**	**&B**

Solution 4

Soit P un pointeur qui 'pointe' sur un tableau A:

```
int A[] = {12, 23, 34, 45, 56, 67, 78, 89, 90};
int *P;
P = A;
```

Quelles valeurs ou adresses fournissent ces expressions:

a) ***P+2** => la valeur 14

b) ***(P+2)** => la valeur 34

c) **&P+1** => l'adresse du pointeur derrière le pointeur P (rarement utilisée)

d) **&A[4]-3** => l'adresse de la composante A[1]

e) **A+3** => l'adresse de la composante A[3]

f) **&A[7]-P** => la valeur (indice) 7

g) **P+(*P-10)** => l'adresse de la composante A[2]

h) ***(P+*(P+8)-A[7])** => la valeur 23

Solution 5

```
#include <stdio.h>
main()
{
 /* Déclarations */
 int A[100], B[50]; /* tableaux */
 int N, M;  /* dimensions des tableaux */
 int I;     /* indice courant */

 /* Saisie des données */
 printf("Dimension du tableau A (max.50) : ");
 scanf("%d", &N );
 for (I=0; I<N; I++)
    {
     printf("Elément %d : ", I);
     scanf("%d", A+I);
    }
 printf("Dimension du tableau B (max.50) : ");
 scanf("%d", &M );
 for (I=0; I<M; I++)
    {
     printf("Elément %d : ", I);
     scanf("%d", B+I);
    }
 /* Affichage des tableaux */
 printf("Tableau donné A :\n");
 for (I=0; I<N; I++)
     printf("%d ", *(A+I));
 printf("\n");
 printf("Tableau donné B :\n");
 for (I=0; I<M; I++)
     printf("%d ", *(B+I));
 printf("\n");
 /* Copie de B à la fin de A */
 for (I=0; I<M; I++)
      *(A+N+I) = *(B+I);
 /* Nouvelle dimension de A */
 N += M;
  /* Edition du résultat */
 printf("Tableau résultat A :\n");
 for (I=0; I<N; I++)
      printf("%d ", *(A+I));
 printf("\n");
 return 0;
}
```

Solution 6

En traitant des tableaux à l'aide de pointeurs, nous utilisons souvent des expressions de la forme:

for (P=A ; P<A+N ; P++) ou **for (P=CH ; *P ; P++)**

```
{                          {
...                        ...
}                          }
```

ou les versions analogues avec while.

Dans ces boucles, le pointeur P est incrémenté à la fin du bloc d'instruction et comparé ensuite à la condition de la boucle. Au moment où la condition est remplie, P pointe déjà à l'extérieur du tableau; plus précisément sur le premier élément derrière le tableau.

Exemple:

```
#include <stdio.h>
main()
```

```
{
 /* Déclarations */
 int A[10]; /* tableau */
 int *P;    /* pointeur dans A */

 /* Saisie des données */
 printf("Introduire 10 entiers : \n");
 for (P=A; P<A+10; P++)
      scanf("%d", P);
 /* Affichage du tableau */
 printf("Tableau donné A :\n");
 for (P=A; P<A+10; P++)
     printf("%d ", *P);
 printf("\n");
 return 0;
}
```

A la fin des boucles, P contient l'adresse A+10 et pointe donc sur l'élément A[10] qui ne fait plus partie du tableau

Solution 7

```
#include <stdio.h>
main() {
 /* Déclarations */
 int A[50];    /* tableau donné        */
 int N;        /* dimension du tableau */
 int X;        /* valeur à éliminer    */
 int *P1, *P2; /* pointeurs d'aide     */
 /* Saisie des données */
 printf("Dimension du tableau (max.50) : ");
 scanf("%d", &N );
 for (P1=A; P1<A+N; P1++)
     {
      printf("Elément %d : ", P1-A);
      scanf("%d", P1);
     }
  printf("Introduire l'élément X à éliminer du tableau : ");
 scanf("%d", &X );
 /* Affichage du tableau */
 for (P1=A; P1<A+N; P1++)
     printf("%d ", *P1);
  printf("\n");
 /* Effacer toutes les occurrences de X et comprimer :  */
/* P2 pour tous les éléments différents de X.          */
 for (P1=P2=A; P1<A+N; P1++)
     {
      *P2 = *P1;
      if (*P2 != X)
          P2++;
     }
  /* Nouvelle dimension de A */
 N = P2-A;
  /* Edition du résultat */
 for (P1=A; P1<A+N; P1++)
     printf("%d ", *P1);
  printf("\n"); }
```

Exercice 5.8 Fusion de deux tableaux triés

```
#include <stdio.h>
main() {
  /* Les tableaux et leurs dimensions */
 int A[50], B[50], FUS[100];
 int N, M;
 int IA, IB, IFUS; /* indices courants */
/* Saisie des données */
 printf("Dimension du tableau A (max.50) : ");
 scanf("%d", &N );
 printf("Entrer les éléments de A dans l'ordre croissant :\n");
 for (IA=0; IA<N; IA++)
    {
     printf("Elément A[%d] : ", IA);
     scanf("%d", &A[IA]);
    }
 printf("Dimension du tableau B (max.50) : ");
 scanf("%d", &M );
 printf("Entrer les éléments de B dans l'ordre croissant :\n");
 for (IB=0; IB<M; IB++)
    {
     printf("Elément B[%d] : ", IB);
     scanf("%d", &B[IB]);
    }
 /* Affichage des tableaux A et B */
 printf("Tableau A :\n");
 for (IA=0; IA<N; IA++)
     printf("%d ", A[IA]);
 printf("\n");
 printf("Tableau B :\n");
 for (IB=0; IB<M; IB++)
     printf("%d ", B[IB]);
 printf("\n");
 /* Fusion des éléments de A et B dans FUS */
 /* de façon à ce que FUS soit aussi trié. */
 IA=0; IB=0; IFUS=0;
 while ((IA<N) && (IB<M))
       if(A[IA]<B[IB])
           {
           FUS[IFUS]=A[IA];
           IFUS++;
           IA++;
           }
       else
           {
           FUS[IFUS]=B[IB];
           IFUS++;
           IB++;
           }
 /* Si IA ou IB sont arrivés à la fin de leur tableau, */
 /* alors copier le reste de l'autre tableau.          */
 while (IA<N)
       {
        FUS[IFUS]=A[IA];
        IFUS++;
        IA++;
```

```
        }
 while (IB<M)
        {
         FUS[IFUS]=B[IB];
         IFUS++;
         IB++;
        }
  /* Edition du résultat */
 printf("Tableau FUS :\n");
 for (IFUS=0; IFUS<N+M; IFUS++)
     printf("%d ", FUS[IFUS]);
 printf("\n");
 return 0;}
```

Exercice 5.9 Tri par sélection du maximum

```
#include <stdio.h>
main() {
 /* Déclarations */
 int A[50]; /* tableau donné */
 int N;     /* dimension     */
 int I;     /* rang à partir duquel A n'est pas trié */
 int J;     /* indice courant      */
 int AIDE;  /* pour la permutation */
 int PMAX;  /* indique la position de l'élément */
            /* maximal à droite de A[I]         */
 /* Saisie des données */
 printf("Dimension du tableau (max.50) : ");
 scanf("%d", &N );
 for (J=0; J<N; J++)
    {
     printf("Elément %d : ", J);
     scanf("%d", &A[J]);
    }
 /* Affichage du tableau */
 printf("Tableau donné :\n");
 for (J=0; J<N; J++)
    printf("%d ", A[J]);
 printf("\n");
/* Tri du tableau par sélection directe du maximum. */
 for (I=0; I<N-1; I++)
     {
      /* Recherche du maximum à droite de A[I] */
      PMAX=I;
      for (J=I+1; J<N; J++)
          if (A[J]>A[PMAX]) PMAX=J;
      /* Echange de A[I] avec le maximum */
      AIDE=A[I];
      A[I]=A[PMAX];
      A[PMAX]=AIDE;
     }
  /* Edition du résultat */
 printf("Tableau trié :\n");
 for (J=0; J<N; J++)
     printf("%d ", A[J]);
 printf("\n");
 return 0;}
```