Langage C – TP C 4 - correction **exercice 1 question 1**

```
#include <stdio.h>
#include <stdlib.h>                    // définition de la fonction rand()

void main(void)
{
     int i;
     for(i=0;i<10;i++)
     {                                 // facultative car une seule instruction
          printf("%d ",rand()); // nombre aléatoire compris entre 0 et RAND_MAX=32767(10)
     }                                 // facultative
}
```

/* A chaque nouvelle exécution du programme, ce sont les mêmes nombres aléatoires qui sont générés. */

Langage C – TP C 4 - correction **exercice 1 question 2**

```
#include <stdio.h>
#include <stdlib.h>                    // définition des fonctions rand() et srand()
#include <time.h>                      // définition de time(NULL)

void main(void)
{
     int i;
     srand(time(NULL));                // à appeler une seule fois, en début de programme
     for(i=0;i<10;i++)
     {                                 // facultative car une seule instruction
          printf("%d ",rand());
     }                                 // facultative
}
```

Langage C – TP C 4 - correction **exercice 1 question 3**

```
//                0  ≤     rand()         ≤ RAND_MAX
//                0  ≤     rand()%109 ≤ 108(10)
// -9+0=-9(10)  ≤ -9+rand()%109 ≤ -9+108(10)=99(10)
```

```
#include <stdio.h>
#include <stdlib.h>                    // définition des fonctions rand() et srand()
#include <time.h>                      // définition de time(NULL)

void main(void)
{
    int i;
    srand(time(NULL));                 // à appeler une seule fois, en début de programme
    for(i=0;i<10;i++)
    {                                  // facultative car une seule instruction
        printf("%d ",-9+rand()%109);   // nombre aléatoire compris entre -9(10) et 99(10)
    }                                  // facultative
}
```

Langage C – TP C 4 - correction **exercice 2 question 1**

// Un tableau est une suite ordonnée de valeurs du même type.

// 2 15 3 -5 0 58 62 1 47 96 3 -8 94 55 -9 35 18 0 61 30

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

#define NB_ELEMENTS 20

void main(void)
{
    int tableau_entier[NB_ELEMENTS];
    int i;
    srand(time(NULL));
    for(i=0;i<NB_ELEMENTS;i++)
    {
        tableau_entier[i]=-9+rand()%109;        // initialisation des valeurs du tableau d'entiers
        printf("%d ",tableau_entier[i]);        // affichage des valeurs du tableau d'entiers
    }
}
```

Langage C – TP C 4 - correction **exercice 2 question 2**

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

#define NB_ELEMENTS 20

void remplir(int*,int);
void afficher(int*,int);

void main(void)
{
    int tableau_entier[NB_ELEMENTS];
    srand(time(NULL));
    remplir(tableau_entier,NB_ELEMENTS);
    afficher(tableau_entier,NB_ELEMENTS);
}

/* procédure qui remplit un tableau d'entiers avec des valeurs aléatoires comprises entre -9(10) et 99(10) */
void remplir(int* tab_entier,int taille)
{
    int i;
    for(i=0;i<taille;i++)
    {                                          // facultative car une seule instruction
        tab_entier[i]=-9+rand()%109;
    }                                          // facultative
}
/* procédure qui affiche le contenu d'un tableau d'entiers */
void afficher(int* tab_entier,int taille)
{
    int i;
    for(i=0;i<taille;i++)
    {                                          // facultative car une seule instruction
        printf("%d ",tab_entier[i]);
    }                                          // facultative
}
```

Langage C – TP C 4 - correction **exercice 3**

```
#define NB_ELEMENTS 20
int tableau_entier[]={2,15,3,-5,0,58,62,1,47,96,3,-8,94,55,-9,35,18,0,61,30};
afficher(tableau_entier,NB_ELEMENTS);
        // 2  15  3  -5  0  58  62  1  47  96  3  -8  94  55  -9  35  18  0  61  30
afficher_envers(tableau_entier,NB_ELEMENTS);
        // 30  61  0  18  35  -9  55  94  -8  3  96  47  1  62  58  0  -5  3  15  2
```

```
/* procédure qui affiche à l'envers le contenu d'un tableau d'entiers sans en modifier le
     contenu */
void afficher_envers(int* tab_entier,int taille)
{
    int i;
    for(i=taille-1;i>=0;i--)
    {                                   // facultative car une seule instruction
        printf("%d ",tab_entier[i]);
    }                                   // facultative
}
```

Langage C – TP C 4 - correction **exercice 4**

```
#define NB_ELEMENTS 20
int tableau_entier[]={2,15,3,-5,0,58,62,1,47,96,3,-8,94,55,-9,35,18,0,61,30};
inverser(tableau_entier,NB_ELEMENTS);
                                                    // i=0          tableau_entier[0]↔tableau_entier[19]
                                                    // i=1          tableau_entier[1]↔tableau_entier[18]
                                                    //                                                  …
                                                    // i=20/2-1=9   tableau_entier[9]↔tableau_entier[10]
                                  // si le tableau a 19 valeurs (19/2-1=9-1=8), tableau_entier[0]↔tableau_entier[18] …
                                                  tableau_entier[8]↔tableau_entier[10]  et tableau_entier[9] est inchangé
afficher(tableau_entier,NB_ELEMENTS);
                                          // 30  61  0  18  35  -9  55  94  -8  3  96  47  1  62  58  0  -5  3  15  2
```

```
/* procédure qui inverse le contenu d'un tableau d'entiers */
void inverser(int* tab_entier,int taille)
{
    int i,tmp;
    for(i=0;i<taille/2;i++)
    {
        tmp=tab_entier[i];
        tab_entier[i]=tab_entier[taille-1-i];
        tab_entier[taille-1-i]=tmp;
    }
}
```

```
{
    int i=0,j=taille-1,tmp;
    while(i<j)
    {
        tmp=tab_entier[i];
        tab_entier[i]=tab_entier[j];
        tab_entier[j]=tmp;
        i++;
        j--;
    }
}
```

Langage C – TP C 4 - correction **exercice 5**

```
#define NB_ELEMENTS 20
int tableau_entier[NB_ELEMENTS];
printf("valeur maximale=%d\n",maximum(tableau_entier,NB_ELEMENTS));

/* fonction qui recherche le maximum parmi les valeurs d'un tableau d'entiers
   et qui renvoie cette valeur au programme principal pour affichage */
int maximum(int* tab_entier,int taille)
{
    int i,max=tab_entier[0];
    for(i=1;i<taille;i++)
    {                                   // facultative car une seule instruction
        if(tab_entier[i]>max)
        {                               // facultative
            max=tab_entier[i];
        }                               // facultative
    }                                   // facultative
    return max;
}
```

Langage C – TP C 4 - correction **exercice 6**

```
/* procédure qui détermine et affiche le minimum et le maximum
   parmi les valeurs d'un tableau d'entiers en utilisant les deux
   fonctions précédentes */
void min_max_afficher(int* tab_entier,int taille)
{
    printf("minimum=%d,maximum=%d\n",minimum(tab_entier,taille),maximum(tab_entier,taille));
}
```

Langage C – TP C 4 - correction **exercice 6**

```
/* procédure qui détermine et affiche le minimum et le maximum parmi les valeurs d'un tableau
     d'entiers en utilisant les deux fonctions précédentes */
void min_max_afficher(int* tab_entier,int taille)
{
    printf("minimum=%d, maximum=%d\n",minimum(tab_entier,taille),maximum(tab_entier,taille));
}
-----------------------------------------------------------------------------------------------
#define NB_ELEMENTS 20
int tableau_entier[NB_ELEMENTS];
int minmax_entier[2];
                    // minmax_entier[0] pour le minimum, minmax_entier[1] pour le maximum
min_max_transmettre(tableau_entier,NB_ELEMENTS,minmax_entier);
printf(« minimum=%d, maximum=%d\n",minmax_entier[0],minmax_entier[1]);

/* procédure qui recherche le minimum et le maximum parmi les valeurs d'un tableau d'entiers et
     qui « transmet » ces valeurs au programme principal pour affichage */
void min_max_transmettre(int* tab_entier,int taille,int* mm_entier)
{
   mm_entier[0]=minimum(tab_entier,taille);
   mm_entier[1]=maximum(tab_entier,taille);
}
```

Langage C – TP C 4 - **éléments de correction**

- **Prototype des fonctions du programme final**

```
void remplir(int*,int);

void afficher(int*,int);
void afficher_envers(int*,int);

void inverser(int*,int);

int maximum(int*,int);
int minimum(int*,int);

void min_max_afficher(int*,int);
void min_max_transmettre(int*,int,int*);

void max_max_afficher(int*,int);

void tri_selectif(int*,int);
void tri_bulle(int*,int);
void tri_par_comptage(int*,int);
```