# Chapter 2: Pointers
## Exercise series

**TD**

**Exercise #1.** Either the following declarations and initializations

```
int A[] = {1, 3, 5, 2, 8, 4, 9, 0};   int *P =A,*Q,*V;
```

Determine the correct and incorrect instructions and then give the results of executing the correct instructions on the vector and pointers

```
*P+2=A[3];  error i assignement (exp=value???)
P=P+A[2];
Q++=&A[5];   error (we doesn't use a not initialised variable)
Q=&A[3];
A[6]=Q-P;
*(V+1)=*(P+1); V not initialized
V=Q-2 ;
P=P-*(A+1);
for (Q=P-3 ;Q<=V ;Q++)    (*Q)++ ;
P=P+(Q-V)+2 ;
```

| 2 | 4 | 5 | 2 | 8 | 4 | -2 | 0 |
|---|---|---|---|---|---|---|---|
| | V | Q | | | P | | |

**Exercise #2**. Complete the following table which shows the value of each variable after each statement.

| **Instruction** | **A** | **b** | **c** | **p1** | **p2** |
|---|---|---|---|---|---|
| int a, b, c,*p1,*p2; | / | / | / | / | / |
| a=1;b=2;c=3; | 1 | 2 | 3 | / | / |
| p1=&a;p2=&c; | 1 | 2 | 3 | &a | &c |
| *p1=(*p2)++; | 3 | 2 | 4 | &a | &c |
| p1=p2; p2=&b; | 3 | 2 | 4 | &c | &b |
| *p1-=*p2; | 3 | 2 | 2 | &c | &b |
| **Instruction** | **A** | **b** | **C** | **p1** | **p2** |
| ++*p2; | 3 | 3 | 2 | &c | &b |
| *p1*=*p2; | 3 | 3 | 6 | &c | &b |
| a=+++*p2**p1; | 24 | 4 | 6 | &c | &b |
| p1=&a; | 24 | 4 | 6 | &a | &b |
| *p2=*p1/=*p2; | 6 | 6 | 6 | &a | &b |

**Exercise #3 :** Write a program that fills an array T with real numbers, then creates two arrays TP and TN, and places all positive numbers in TP and all negative numbers in TN, and leaves the zero numbers as is

```
#include <stdio.h>
#include <stdlib.h>
int main() {
int *T,*TP,*TN, n, i, j, k;
printf("entrez nbr des elements\n");
scanf("%d",&n);
T=(int*)malloc(n*sizeof(int));
for( i=0; i<n; i++) {
printf("T[%d]=",i);
scanf("%d",T+i);
}
```

```
 TP=(int*)malloc(n*sizeof(int));
 TN=(int*)malloc(n*sizeof(int));
 j=k=0;
 for( i=0; i<n; i++)
 if(T[i]>0)
 TP[j++]=T[i];
 else if(T[i]<0)
 TN[k++]=T[i];
 TP=(int*)realloc(TP, j*sizeof(int));
 TN=(int*)realloc(TN, k*sizeof(int));
 printf("\nla table des positifs\n");
 for( i=0; i<j; i++)
 printf("%d\t",TP[i]);
 printf("\nla table des négatifs\n");
 for( i=0; i<k; i++)
 printf("%d\t",TN[i]);
 reurn 0;
```

**Exercise #4 :** Write a function swaps the values of two variables, Use this function to reverse the elements of a dynamic vector.

```
#include <stdio.h>
#include <stdlib.h>
void swap( int *x,int *y){
int t;
t=*x;
*x=*y;
*y=t;}
int main()
{  int n,i,*T;
   printf ("enter the vector's size ");
   scanf("%d",&n);
   T=(int *)malloc(n*sizeof (int));
   printf("enter elementsvof T:");
   for(i=0;i<n;i++)
   scanf("%d",T+i);
   for(i=0;i<n;i++)
   printf("%d ",*(T+i));
   for(i=0;i<=(n/2);i++)
   swap(T+i, T+n-i-1);
   printf("result :\n");
   for(i=0;i<n;i++)
   printf("%d ",*(T+i));
   return 0;
}
```

**Exercice #5:** Using the pointer formalism:

- Write a recursive function that determines the last character in a string.

```
#include <stdio.h>
#include <stdlib.h>
char last (char *t)
{  if(*t=='\0')
       return *(t-1);
   else
      return last(t+1);}
int main()
{   int i;
    char *T="bonjour";
    printf ("last char :%c",last(T));
    return 0;
}
```

- Write a recursive function that calculates the sum of the elements of a vector.

Test these functions in a main function.

```
#include <stdio.h>
#include <stdlib.h>
int sum (int *t,int n, int d)
{   if(d==n)
        return 0;
    else
      return *(t+d)+ sum( t,n,d+1);}

int main(){
       int n,i,*T;
       printf ("enter the vector's size ");
       scanf("%d",&n);
       T=(int *)malloc(n*sizeof (int));
       printf("enter elementsvof T:");
       for(i=0;i<n;i++)
           scanf("%d",T+i);
       printf ("sum of vector elements : %d",sum (T,n,0));
       return 0;
}
```

**TP**

**Exercise #1** . Write a program to check the results of exercise 1 of TD.

The program after ignoring the incorrect instructions

```
#include <stdio.h>
     int main(){
     int A[] = {1, 3, 5, 2, 8, 4, 9, 0};
      int i, *P =A,*Q,*V;
     P=P+A[2];
      Q=&A[3];
     A[6]=Q-P;
```

```
    V=Q-2 ;
     P=P-*(A+1);
    for (Q=P-3;Q<=V;Q++) (*Q)++ ;
    P=P+(Q-V)+2 ;
    //display results
     printf(" \n vector A : \n");
    for(i=0;i<8;i++)
    printf("%d  ",A[i]);
    int *s;
    printf(" \n pointer P : \n");
    for (s=P ;s<=A+7 ;s++)
     printf("%d  ",*s);

     printf(" \n pointer  V : \n");
    for (s=V ;s<=A+7 ;s++)
     printf("%d  ",*s);

     printf(" \n Pointer Q : \n");
     for (s=Q ;s<=A+7 ;s++)
     printf("%d  ",*s);
}
```

**Exercise #2 .**Write a program that:

- reads an integer N.
- create a dynamic array of N Integers.
- Add its index to each element.
- Display the table using the formalisms (array/pointer).

```
#include <stdio.h>
#include <stdlib.h>
int main()
{   int n,i,*T;
    printf ("enter the vector size ");
    scanf("%d",&n);
    T=(int *)malloc(n*sizeof (int));
    printf("enter elements of T:");
    for(i=0;i<n;i++)
    scanf("%d",T+i);
    for(i=0;i<n;i++)
    *(T+i)=*(T+i)+i;
    printf("result 1:formalims array\n");
    for(i=0;i<n;i++)
    printf("%d ",T[i]);
    printf("\n result 1:formalims pointer\n");
    for(i=0;i<n;i++)
    printf("%d ",*(T+i));
    return 0;
}
```

**Exercise**#3 : write a program which reads two vectors of integers V1 and V2 then adds at the end of a vector V1 the elements of a vector V2

```
#include <stdio.h>
#include <stdlib.h>
int main()
{   int n1,n2,i,*T1,*T2;
        printf ("enter the vector1 size ");
        scanf("%d",&n1);
        T1=(int *)malloc(n1*sizeof (int));
        printf("enter elements of T1:");
        for(i=0;i<n1;i++)
           scanf("%d",T1+i);
        printf ("enter the vector2 size ");
        scanf("%d",&n2);
        T2=(int *)malloc(n2*sizeof (int));
        printf("enter elements of T2:");
        for(i=0;i<n1;i++)
             scanf("%d",T2+i);
        T1=(int *)realloc(T1,(n1+n2)*sizeof(int));
        for(i=0;i<n2;i++)
            *(T1+i+n1)=*(T2+i);
        printf("result \n");
        for(i=0;i<n1+n2;i++)
             printf("%d ",T1[i]);
        return 0;
}
```

**Exercise #4 :** Ecrire une fonction  récursive qui inverse une chaine de caractère dans une autre chaine.

Ecrire une fonction qui renvoi 1 si deux chaines de caractère sont égaux et 0 sinon.
Ecrire la fonction main qui lit une chaine de caractère et qui utilise ces deux fonctions pour vérifier si cette chaine est palindrome ou non.

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

void reverse( char *s1 ,char *s2,int n ,int d)
{     if (n==d)
            *(s2+d)=0;
      else
            {*(s2+d)=*(s1+n-1-d);
       reverse(s1,s2,n,d+1);}
}

int compch(char *s1,char *s2)
{     if ((*s1==0) && (*s2==0))
```

```
            return 1;
        if(*s1!=*s2)
            return 0;
        else
            return compch(s1+1,s2+1);
}

int main()
{       int n;
        char *S,*T="aba";
        n=strlen(T);
        S=(char *)malloc(n*sizeof (char));
        printf("revrtse string : ");
        reverse(T,S,n,0);
        puts(S);
        if(compch(S,T)==1 )
            printf ("the string is palindrome");
        else
            printf ("the string is not palindrome");
        return 0;
}
```