Level: 1st year computer science **Material** : ADS2	**Correction type** **TD/TP No.: 01**	**Academic year:** 2023/2024 **Chapter 1** : Subprograms

Exercise 1: (TP)

Create the " displayNbs " subroutine to showcase on the screen numbers that fall below a specified limit.

```
#include <stdio.h>
void displayNbs (int n){
  int i ;
  for (i=0 ;i<n ;i++)
    printf("%d ",i);
}
int main(){
  int x ;
  prinf("enter a nbr\n");
  scanf("%d",&x);
  displayNbs (x);
}
```

Exercise 2: (TD)

mirror subroutine which takes a natural number, and displays its inverted image on the screen, for example 5973, and the program will display 3795

```
#include <stdio.h>
void mirror(int n){
  do {
    printf("%d", n % 10);
    n=n / 10;
  } while (n>0);
}
int main() {
  int x;
  printf("enter a number: ");
  scanf("%d", &x);
  mirror(x);
}
```

Exercise 3: (TD/TP)

displayTab subroutine to display the elements of an array

```
#include <stdio.h>
void displayTab (float t[],int n){
  int i ;
  for (i=0 ;i<n ;i++)
    printf("%f\t",t[i]);
}
float sumTab (float t[],int n){
  int i ; float s=0;
  for (i=0 ;i<n ;i++)
    s=s+t[i];
  return s;
}
int main(){
  float tab[100];
  int N,i ;
  printf("enter the number of elements <=100");
  scanf("%d", &N);
  for ( i=0;i<N;i++){
    printf("%d=>" ,i);
    scanf("%f", &tab[i]);
  }
  displayTab (tab,N);
}
```

Exercise 4: (TD)

min procedure that returns the max between 2 real numbers

– Using a global variable
– Using pass by variable
– Rewrite this procedure as a function

```
1 min ( float x, float y) {
2   if (x <= y)
3     printf( "%f \n " , x);
4   else
5     printf( "%f \n " , y);
6 }
```

```
1 float m;
2 min ( float x, float y) {
3   if (x <= y)
4     m = x;
5   else
6     m = y;
7 }
```

```
1 min ( float x, float y, float *m) {
2   if (x <= y)
3     *m = x;
4   else
5     *m = y;
6 }
```

```
1 float min ( float x, float y) {
2   if (x <= y)
3     return x;
4   return y;
5 }
```

Exercise 5: (TD/TP)

– Define a structure to hold the coordinates of a point (x, y).
– Write a " `norm` " subroutine to calculate the norm of a vector.

```
#include <stdio.h>
#include <math.h>
typedef struct{
  float x,y;          .75
} point;
float standard ( point p1, point p2) {
  return sqrt((p1.x-p2.x)*(p1.x-p2.x)+(p1.y-p2.y)*(p1.y-p2.y)) ;
}
int main(){
  point p1,p2;
  printf("enter coordinates of 1st point\n");
  scanf("%f%f",&p1.x,&p1.y);
  printf("enter coordinates of 2nd point\n");
  scanf("%f%f",&p2.x,&p2.y);
  printf("norm=%.2f", norm(p1, p2));
}
```

Exercise 6: (TP)

– Write the `isSeparator` subroutine to see if a character is a separator or not. Separators are (.?!, and space)
– `countWord` subroutine to count the number of words in a sentence

```
#include <stdio.h>
#include <string.h>
int isSeparator ( char x){
  return x== ' ' ||x== ','
||x== '.' ||x== '!' ||x== '?' ;
}
int countWord ( char s[]) {
  int i, nbWrd, n;
  i = nbWrd = 0 ;
```

```
  n = strlen(s);
  for (i=0;i<n-1;i++)
    if (!isSeparator(s[i])&& isSeparator (s[i+1]))
      nbWrd++;
  if (n>0 && !isSeparator (s[n-1]))
    nbWrd++;
  return nbWrd;
}
int main(){
  char s[100];
  printf("enter a sentence\n");
  gets(s);
  printf("the number of words is: %d", countWord (s) );
}
```

Exercise 7: (TD)

- `fact` subroutine to calculate the factorial of a number.
- `power_1` subroutine to calculate -1 to the power of y
- `power` subroutine to calculate x to the power of y
- Write the subroutine `cos` to calculate the following sum:

$$S = \sum_{i=0}^{n} \frac{(-1)^i x^i}{(2i)!}$$

```
#include <stdio.h>
int fact ( int y) {
  int I, p = 1 ;
  for (i = 2 ; i < y; i++)
    p = p * i;
  return p;
}
int power_1 ( int y) {
  if (n%2)
    return -1;
  return 1;
}
float power ( float x, int y) {
  float p = 1 ;
  int i;
  for (i = 0 ; i < y; i++)
    p = p * x;
  return p;
}
float cos ( float x, int n) {
  float s = 0 ;
  int i;
  for (i = 0 ; i < n; i++)
    s = s * power_1(i) * power (x, i)/ fact ( i );
  return s;
}
int main(){
 float x;
 printf("enter a number ");
 scanf("%f", &x);
 printf("the cosunis= %.2f ",cos(x,10000));
}
```

Exercise 8:

```
#include <stdio.h>
int isPrime(int num) {
 int i;
 if (num < 2) return 0;
 for (i = 2; i * i <= num; i++) {
  if (num % i == 0) return 0;
```

```
 return 1;
}
int main() {
 int i, n;
 printf("enter an even number:\n");
 scanf("%d", &n);
 printf("Pairs of prime numbers whose sum is %d:\n", n);
 for (i = 2; i <= n / 2; i++)
  if (isPrime(i) && isPrime(n - i))
   printf("(%d, %d)\n", i, n - i);
 return 0;
}
```

Exercise 9:

– If you know that a perfect number is a number that equals the sum of its divisors except 1 and itself. Write the `isPerfect` subroutine to see if the number is perfect or not

– Write a program to display all perfect numbers less than N

```
#include <stdio.h>
int isPerfect ( int n) {
  int s, i;
  s = 0 ;
  for (i = 1 ; i <= n / 2 ; i++)
    if (n % i == 0 ) s += i;
  return n == s;
}
int main(){
  int I,n;
  printf("enter a number ");
  scanf("%d", &n);
  printf("here is the list of perfect numbers\n");
  for (i = 1; i <= n; i++)
    if ( isPerfect (i))
      printf(" %d\t ",i);
}
```