DSA ASSIGNMENT-2

TANISHA KARMAKAR

21051950

CSE 37

Q1. WAP to find the largest number and counts the occurrence of the largest number in a dynamic array of n integers using a single loop.

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
  int n, biggest, count=1;
  int *arr;
  printf("Enter the number of elements: ");
  scanf("%d", &n);
   arr=(int*) malloc (sizeof(int)*n);
   biggest=0;
   printf("enter the elements of the array: \n");
   for(int i=0; i<n; i++)
   {
     scanf("%d", &arr[i]);
     if (arr[i]>biggest)
       biggest=arr[i];
     else if(arr[i]==biggest)
     count++ ;
   }
   printf("The largest element=%d\n",biggest);
   printf("Largest number count=%d times",count);
    return 0;
}
```

Output:

```
Enter the number of elements: 3 4 5 5 1 2
enter the elements of the array:
The largest element=5
Largest number count=2 times
Press any key to continue . . .
```

Q2. Given a dynamic array, WAP to print the next greater element (NGE) for every element. The next greater element for an element x is the first greater element on the right side of x in array. Elements for which no greater element exist, consider next greater element as -1. E.g. For the input array [2, 5, 3, 9, 7], the next greater elements for each elements are as follows.

Element	NGE
2	5
5	9
3	9
9	-1
7	-1

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

int main()
{
    int n;
    float *a;
    int i, j, found;

    printf("\nInput Number of elements      : ");
    scanf("%d", &n);

    printf("\n");
    a = malloc(n * sizeof(a));
```

```
    if (a == NULL)
    {
        printf("\nMemory not allocated.\n\n");
        exit(0);
    }
    else
    {
        for (i = 0; i < n; i++)
        {
            printf("Input the elements %d of Array : ", i + 1);
            scanf("%f", &a[i]);
        }
        printf("Element      NGE\n");
        for (i = 0; i < n; i++)
        {
            found = 0;
            for (j = i + 1; j < n; j++)
            {
                if (a[i] < a[j])
                {
                    printf("  %g          %g\n", a[i], a[j]);
                    found = 1;
                    break;
                }
            }
            if (found == 0)
            {
                printf("  %g         -1\n", a[i]);
            }
        }
    }
    return 0;
}
```

Output:

```
Input Number of elements      : 5

Input the elements 1 of Array : 2
Input the elements 2 of Array : 5
Input the elements 3 of Array : 3
Input the elements 4 of Array : 9
Input the elements 5 of Array : 7
Element       NGE
  2           5
  5           9
  3           9
  9          -1
  7          -1

Press any key to continue . . .
```

Q3. WAP to store n student's information (i.e. student's roll no, name, gender, marks etc) of an educational institute and display all the data, using array of structure.

```
#include <stdio.h>
#include <string.h>
struct Student{
    char name[20];
    int roll;
    float cgpa;
 };

int main()
{
  int i;
  int n=0;
  printf("Enter Number of Students: ");
  scanf("%d",&n);
  struct Student s[n];
  for(i=0;i<n;i++){
     printf("\nEnter Name:");
```

```
    scanf("%s",&s[i].name);
    printf("\nEnter roll:");
    scanf("%d",&s[i].roll);
    printf("\nEnter cgpa:");
    scanf("%f",&s[i].cgpa);

  }
for(int i = 1; i <= n; i++)
{
 printf("Name of student %d is: %s\n",i,s[i].name );
 printf("Roll number of student %d is: %d\n",i,s[i].roll );
 printf("CGPA of student %d is: %f\n",i,s[i].cgpa );

}
return 0;
}
```

Output:

```
Enter Number of Students: 2

Enter Name:Tanisha

Enter roll:21051950

Enter cgpa:9.1

Enter Name:Arshia

Enter roll:2183280

Enter cgpa:8.9
Name of student 1 is: Arshia
Roll number of student 1 is: 2183280
CGPA of student 1 is: 8.900000
Name of student 2 is: ?!??D
Roll number of student 2 is: 2
CGPA of student 2 is: 0.000000
```

Q4. WAP to store n employee's data such as employee name, gender, designation, department, basic pay. Calculate the gross pay of each employees as follows:

Gross pay = basic pay + HR + DA
HR=25% of basic and DA=75% of basic.

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


struct employee{

char name[50];
char gen[10];
char dg[50];
char dpt[50];
float pay;
};



int main(){

    int i;
    int x=0;
    printf("Enter Number of Employee: ");
    scanf("%d",&x);
    struct employee e[x];
    for(i=0;i<x;i++){
        printf("\nEnter Name:");
        scanf("%s",&e[i].name);
        printf("\nEnter Gender(F/M/Not Say):");
        scanf("%s",&e[i].gen);
        printf("\nEnter Designation:");
```

```
        scanf("%s",&e[i].dg);
        printf("\nEnter Department:");
        scanf("%s",&e[i].dpt);
        printf("\nEnter Basic Pay:");
        scanf("%f",&e[i].pay);
    }
    printf("\nGross pay:");
    for(i=0;i<x;i++){
        int hr,da;
        int p= e[i].pay;
        hr = (25/100)*p;
        da = (75/100)*p;
        e[i].pay = hr+da+p;
        printf("\nName:%s, Gross Pay: %f ",e[i].name,e[i].pay);

    }
    return 0;
}
```

Output:

```
Enter Number of Employee: 1

Enter Name:Tanisha

Enter Gender(F/M/Not Say):F

Enter Designation:Manager

Enter Department:Sales

Enter Basic Pay:20,000

Gross pay:
Name:Tanisha, Gross Pay: 20.000000
Press any key to continue . . .
```

Q5. WAP to declare one distance structure (with members kilometer and meter) and create the variables for addition of two distances using Pointers to structure.

10km500m--D1
21km600m--D2
D3.meter=D1.meter+D2.meter (100)
D3.KM=D1.KM+D2.KM (32)
If(D3.meter>=1000)
D3.KM++;
D3.meter=D3.meter-1000;

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

struct Distance{
   int km;
   int m;
};

int main(){
   struct Distance *d1 = NULL;
   struct Distance *d2 = NULL;
   struct Distance *d3 = NULL;

   d1 = (struct Distance*)malloc(sizeof(struct Distance));
   d2 = (struct Distance*)malloc(sizeof(struct Distance));
   d3 = (struct Distance*)malloc(sizeof(struct Distance));

   printf("Enter metres for first distance: ");
   scanf("%d", &d1->m);
   printf("Enter kilometres for first distance: ");
   scanf("%d", &d1->km);
   printf("Enter metres for second distance: ");
   scanf("%d", &d2->m);
   printf("Enter kilometres for second distance: ");
```

```
    scanf("%d", &d2->km);

    printf("Total metres: %d \n", d1->m);
    printf("Total kilo metres: %d \n", d1->km);


    while(d3->m >= 1000){
        d3->km = d3->km + 1000;
        d3->m = d3->m = 1000;
    }

    printf("Total metres: %d \n", d3->m);
    printf("Total kilo metres: %d \n", d3->km);

}
```

Output:

```
distance
Enter metres for first distance: 2000
Enter kilometres for first distance: 3
Enter metres for second distance: 3000
Enter kilometres for second distance: 4
Total metres: 2000
Total kilo metres: 3
```