NAME - AKRITI CHOUDHARY

ROLL NUMBER - 2005776

SUBJECT - DSA LAB

DATE - 10/8/2021

CLASS - B14

BRANCH - CSE

Question1:Let A be n*n square matrix array. WAP by using appropriate user defined functions for the following:
a) Find the number of nonzero elements in A
b) Find the sum of the elements above the leading diagonal.
c) Display the elements below the minor diagonal.
d) Find the product of the diagonal elements

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

int countNonZero(int *s[], int size)
{
   int count = 0;
   for (int i = 0; i < size; ++i)
   {
      for (int j = 0; j < size; ++j)
      {
         if (s[i][j] != 0)
         {
            count++;
         }
      }
   }
   return count;
}
int SumAboveLeadingDiagonal(int *s[], int size)
{
   int sum = 0;
   for (int i = 0; i < size; ++i)
   {
      for (int j = i + 1; j < size; ++j)
      {
         sum += s[i][j];
      }
   }
   return sum;
}
void DisplayBelowMinorDiagonal(int *s[], int size)
{
   int sum = 0;
   for (int i = 1; i < size; ++i)
   {
      for (int j = size - i; j < size; ++j)
      {
         printf("%d ", s[i][j]);
      }
      puts(" ");
   }
}
void ProductOfDiagonal(int *s[], int size)
{
   int leadingPro = 1;
   int MinorPro = 1;
   for (int i = 0; i < size; ++i)
   {
      for (int j = 0; j < size; ++j)
```

```
      {
         if (i == j)
         {
            leadingPro *= s[i][j];
         }
         if (i == (size - j - 1))
         {
            MinorPro *= s[i][j];
         }
      }
   }
   printf("Product of leading diagonal =  %d\n", leadingPro);
   printf("Product of Minor diagonal =  %d\n", MinorPro);
}

int main()
{

   puts("Enter the number of elements in a square matrix");
   int n;
   scanf("%d", &n);

   int **arr = (int **)malloc(n * sizeof(int *));
   for (int i = 0; i < n; i++)
      arr[i] = (int *)malloc(n * sizeof(int));

   puts("Enter the elements in the square matrix");
   for (int i = 0; i < n; ++i)
   {
      for (int j = 0; j < n; ++j)
      {
         scanf("%d", &arr[i][j]);
      }
   }
   printf("Menu: \n 1 : Number of non zero elements in the matrix \n 2 : Sum of the elements above the
leading diagonal \n 3 :Elements below the minor diagonal\n 4: Product of diagonals\n");
   int ch;
   printf("Enter choice :\n");
   scanf("%d", &ch);
   switch (ch)
   {
   case 1:
      printf("Number of non zero elements in the matrix : %d \n", countNonZero(arr, n));
      break;
   case 2:
      printf("Sum of the elements above the leading diagonal : %d \n", SumAboveLeadingDiagonal(arr,
n));
      break;
   case 3:
      printf("Elements below the minor diagonal \n");
      DisplayBelowMinorDiagonal(arr, n);
      break;
   case 4:
      printf("Product of diagonals: \n");
      ProductOfDiagonal(arr, n);
      break;
   default:
      printf("The choice is invalid \n");
```

```
        break;
    }

    free(arr);
    return 0;
}
```

```
Enter the number of elements in a square matrix
3
Enter the elements in the square matrix
1 2 3
4 5 6
7 8 9
Menu:
 1 : Number of non zero elements in the matrix
 2 : Sum of the elements above the leading diagonal
 3 :Elements below the minor diagonal
 4: Product of diagonals
Enter choice :
1
Number of non zero elements in the matrix : 9
PS D:\KIIT_NOTES\2nd year sem_3\dsa_lab\3_8_2021>
```

```
Enter the number of elements in a square matrix
3
Enter the elements in the square matrix
1 2 3
4 5 6
7 8 9
Menu:
 1 : Number of non zero elements in the matrix
 2 : Sum of the elements above the leading diagonal
 3 :Elements below the minor diagonal
 4: Product of diagonals
Enter choice :
2
Sum of the elements above the leading diagonal : 11
PS D:\KIIT_NOTES\2nd year sem_3\dsa_lab\3_8_2021>
```

```
Enter the elements in the square matrix
1 2 3
4 5 6
7 8 9
Menu:
 1 : Number of non zero elements in the matrix
 2 : Sum of the elements above the leading diagonal
 3 :Elements below the minor diagonal
 4: Product of diagonals
Enter choice :
3
Elements below the minor diagonal
6
8 9
PS D:\KIIT_NOTES\2nd year sem_3\dsa_lab\3_8_2021>
```

```
Enter the number of elements in a square matrix
3
Enter the elements in the square matrix
1 2 3
4 5 6
7 8 9
Menu:
 1 : Number of non zero elements in the matrix
 2 : Sum of the elements above the leading diagonal
 3 :Elements below the minor diagonal
 4: Product of diagonals
Enter choice :
4
Product of diagonals:
Product of leading diagonal =  45
Product of Minor diagonal =  105
PS D:\KIIT_NOTES\2nd year sem_3\dsa_lab\3_8_2021>
```

Question2:Addition of two matrix .

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

   int r , c;
   puts("Enter the number of rows  :");
   scanf("%d",&r);
   puts("Enter the number of columns :");
   scanf("%d",&c);

   int arr1[r][c];

   puts("Enter the elements in the 1st matrix :");
   for(int i = 0; i < r ; ++i){
      for(int j =0 ; j < c; ++j){
         scanf("%d" ,&arr1[i][j]);
      }
   }

   int arr2[r][c];

   puts("Enter the elements in the 2nd matrix :");
   for(int i = 0; i < r ; ++i){
      for(int j =0 ; j < c; ++j){
         scanf("%d" ,&arr2[i][j]);
      }
   }
   int sum[r][c];
   puts("Sum of the elements of the matrix :");
   for(int i = 0; i < r ; ++i){
      for(int j =0 ; j < c; ++j){
         sum[i][j] = arr1[i][j] + arr2[i][j] ;
      }
   }

     for(int i = 0; i < r ; ++i){
      for(int j =0 ; j < c; ++j){
         printf("%d " ,sum[i][j]) ;
      }
      puts(" ");
   }


   return 0;
}
```

```
PS D:\KIIT_NOTES\2nd year sem_3\dsa_lab\10_8_2021> ./addition_of_matrix
Enter the number of rows  :
3
Enter the number of columns :
3
Enter the elements in the 1st matrix :
1 2 3
4 5 6
7 8 9
Enter the elements in the 2nd matrix :
11 12 13
14 9 8
7 6 5
Sum of the elements of the matrix :
12 14 16
18 14 14
14 14 14
PS D:\KIIT_NOTES\2nd year sem_3\dsa_lab\10_8_2021>
```

Question3: Multiplication of two matrix .

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

   int n;
   puts("Enter the number of rows and columns of  square matrix  :");
   scanf("%d", &n);

   int arr1[n][n];

   puts("Enter the elements in the 1st matrix :");
   for (int i = 0; i < n; ++i)
   {
      for (int j = 0; j < n; ++j)
      {
         scanf("%d", &arr1[i][j]);
      }
   }

   int arr2[n][n];

   puts("Enter the elements in the 2nd matrix :");
   for (int i = 0; i < n; ++i)
   {
      for (int j = 0; j < n; ++j)
      {
         scanf("%d", &arr2[i][j]);
      }
   }
   int multi[n][n];
   for (int i = 0; i < n; ++i)
   {
      for (int j = 0; j < n; ++j)
      {
         multi[i][j] = 0;
         for (int k = 0; k < n; ++k)
         {
            multi[i][j] += arr1[i][k] * arr2[k][j];
         }
      }
   }

   puts("The result matrix:");
   for (int i = 0; i < n; ++i)
   {
      for (int j = 0; j < n; ++j)
      {
         printf("%d ",multi[i][j]);
      }
      puts(" ");
   }

   return 0;
}
```

```
PS D:\KIIT_NOTES\2nd year sem_3\dsa_lab\10_8_2021> ./multiplication_of_matrix
Enter the number of rows and columns of  square matrix  :
3
Enter the elements in the 1st matrix :
1 2 3
4 5 6
7 8 9
Enter the elements in the 2nd matrix :
7
8
3
4 5 6
12 4 56
The result matrix:
51 30 183
120 81 378
189 132 573
PS D:\KIIT_NOTES\2nd year sem_3\dsa_lab\10_8_2021>
```

Question4: WAP to store n employees data such as employee name, gender, designation, department, basic pay etc using structures with dynamically memory allocation. Calculate the gross pay of each employees as follows:
Gross pay=basic pay + HR + DA
HR=25% of basic, DA=75% of basic

```
#include <stdio.h>
#include <stdlib.h>
struct employee
{
   char name[20];
   char gender;
   char designation[10];
   char department[10];
   float basic_pay;
   float gross_pay;
};

void input(struct employee *em_ptr)
{
   printf("Enter the name : ");
   scanf("%19s", &(em_ptr->name));
   puts("");
   printf("Enter gender(f - female, m - male, o - others) : ");
   scanf(" %c", &(em_ptr->gender));
   puts("");
   printf("Enter the designation : ");
   scanf(" %9s", &(em_ptr->designation));
```

```
    puts("");
    printf("Enter the department : ");
    scanf(" %9s", &(em_ptr->department));
    puts("");
    printf("Enter the basic pay : ");
    scanf(" %f", &(em_ptr->basic_pay));
    puts("");
}

void calculate(struct employee *em_ptr)
{
    float HR, DA;
    HR = 0.25 * em_ptr->basic_pay;
    DA = 0.75 * em_ptr->basic_pay;
    em_ptr->gross_pay = em_ptr->basic_pay + HR + DA;
}
void display(struct employee *em_ptr)
{
    printf("Name : ");
    printf("%s", em_ptr->name);
    puts("");
    printf("Gender : ");
    printf("%c", em_ptr->gender);
    puts("");
    printf("Designation : ");
    printf("%s", em_ptr->designation);
    puts("");
    printf("Department : ");
    printf("%s", em_ptr->department);
    puts("");
    printf("Basic pay : ");
    printf("%.3f", em_ptr->basic_pay);
    puts("");
    printf("Gross pay : ");
    printf("%.3f", em_ptr->gross_pay);
    puts("");
}
int main()
{
    int n;
    puts("Enter number of employees");
    scanf("%d", &n);
    struct employee *arr[n];
    for (int i = 0; i < n; ++i)
    {
        arr[i] = (struct employee *)malloc(sizeof(struct employee));

        input(arr[i]);
        calculate(arr[i]);
    }
    puts("----------------------------------------------------------------");

    for (int i = 0; i < n; ++i)
    {
        display(arr[i]);
        puts(" ");
        free(arr[i]);
    }
```

```
    puts("Memory is successfully freed");
    return 0;
}
```

```
PS D:\KIIT_NOTES\2nd year sem_3\dsa_lab\10_8_2021> ./employee
Enter number of employees
2
Enter the name : Radha

Enter gender(f - female, m - male, o - others) : f

Enter the designation : CEO

Enter the department : CSE

Enter the basic pay : 1000000

Enter the name : Ravi

Enter gender(f - female, m - male, o - others) : m

Enter the designation : ECE

Enter the department : ECE

Enter the basic pay : 20897

----------------------------------------------------------------
Name : Radha
Gender : f
Designation : CEO
Department : CSE
Basic pay : 1000000.000
Gross pay : 2000000.000

Name : Ravi
Gender : m
Designation : ECE
Department : ECE
Basic pay : 20897.000
Gross pay : 41794.000

Memory is successfully freed
PS D:\KIIT_NOTES\2nd year sem_3\dsa_lab\10_8_2021>
```

Question5:WAP to add two distances (in km-meter) by using dynamic memory allocation.

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

struct dist
{
   int k;
   int m;
};

int main()
{
   struct dist *d1, *d2;
   d1 = (struct dist *)malloc(sizeof(struct dist));
   d2 = (struct dist *)malloc(sizeof(struct dist));

   puts("Enter distance in km");
   scanf(" %d", &(d1->k));
   puts("Enter distance in m");
   scanf(" %d", &(d1->m));

   puts("Enter distance in km");
   scanf(" %d", &(d2->k));
   puts("Enter distance in m");
   scanf(" %d", &(d2->m));

   struct dist *sum = (struct dist *)malloc(sizeof(struct dist));
   sum->k = d1->k + d2->k;
   sum->m = d1->m + d2->m;

   if (sum->m >= 1000)
   {
      sum->k = sum->k + 1;
      sum->m = sum->m - 1000;
   }
   printf("distance in km = %d \n", sum->k);
   printf("distance in m = %d \n", sum->m);

   free(d1);
   free(d2);
   free(sum);
   puts("Memory is freed successfully");
   return 0;
}
```

```
PS D:\KIIT_NOTES\2nd year sem_3\dsa_lab\10_8_2021> ./distance
Enter distance in km
12
Enter distance in m
900
Enter distance in km
4
Enter distance in m
468
distance in km = 17
distance in m = 368
Memory is freed successfully
PS D:\KIIT_NOTES\2nd year sem_3\dsa_lab\10_8_2021>
```