***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");

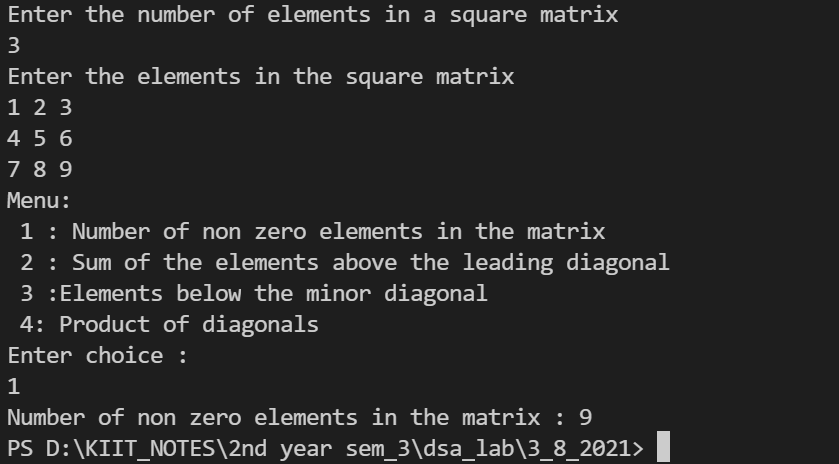
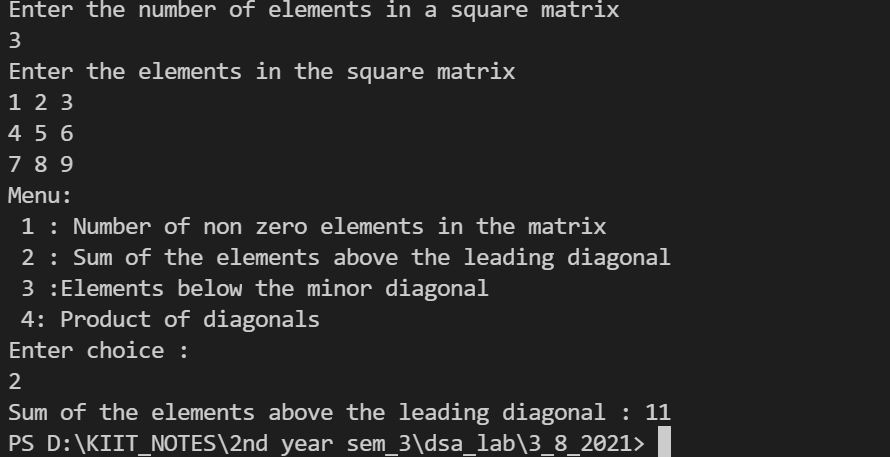
break;

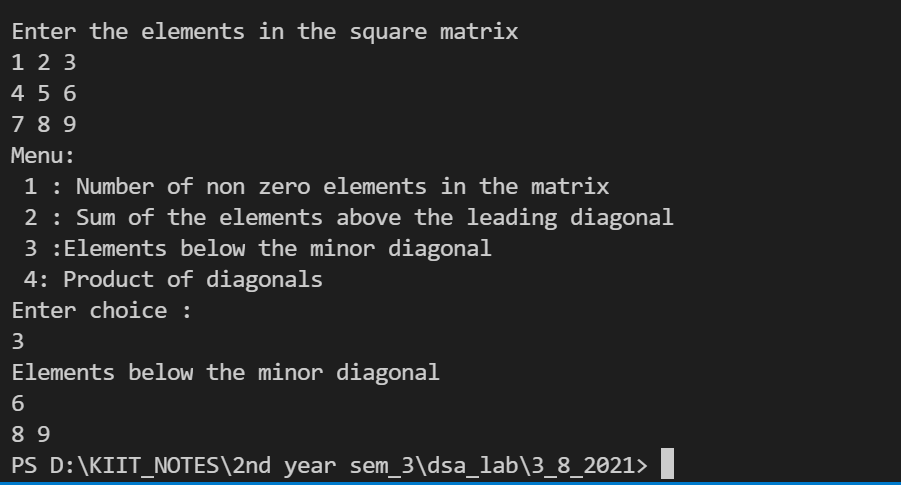
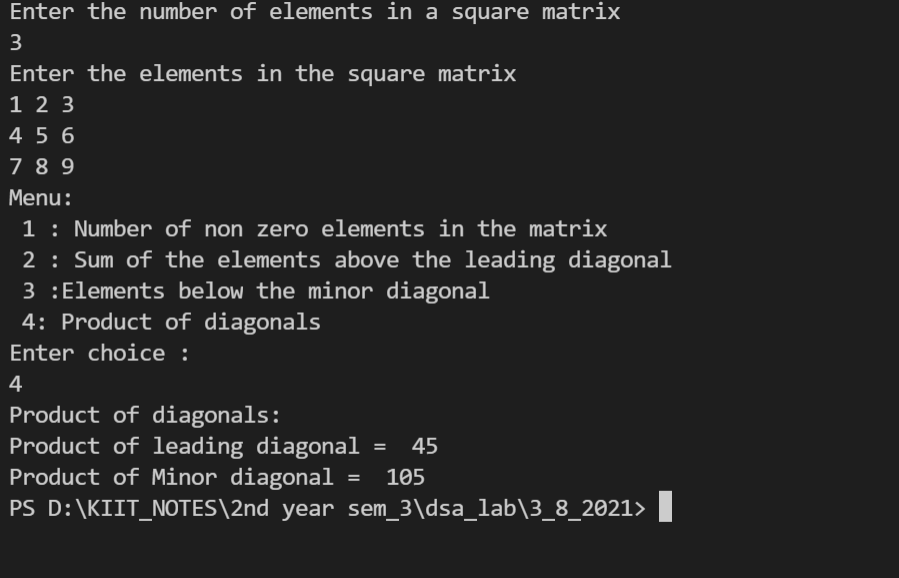
}

free(arr);

return 0;

}

***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]) ;

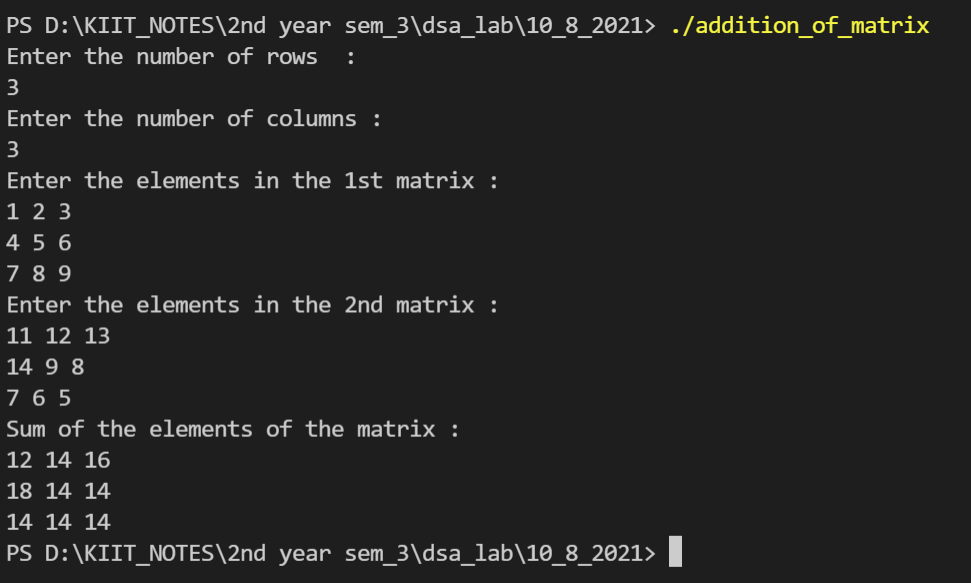
}

puts(" ");

}

return 0;

}



***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]);

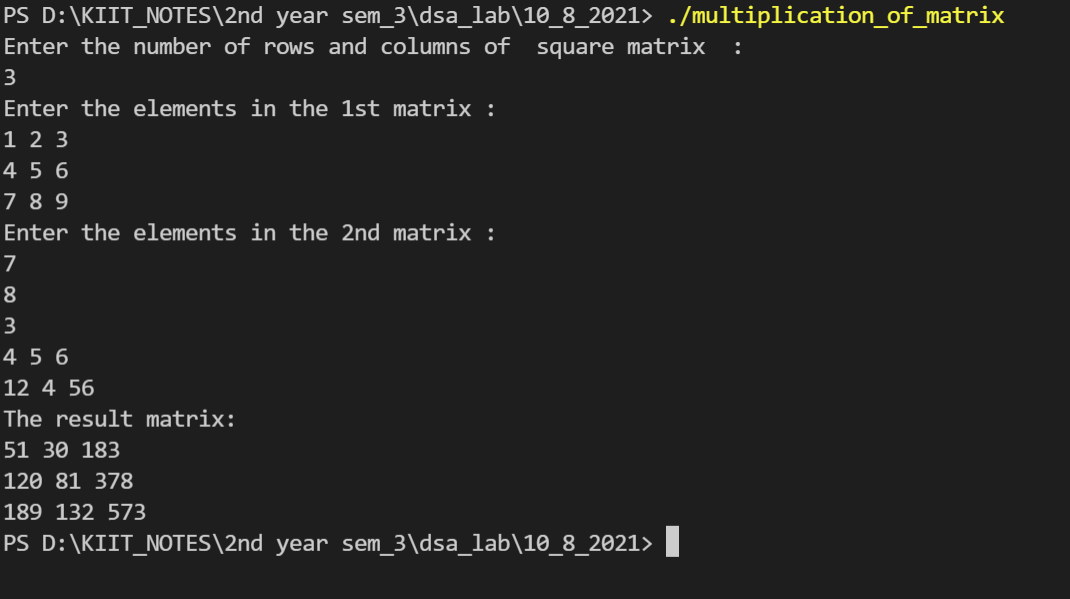
}

puts(" ");

}

return 0;

}



***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(" ");

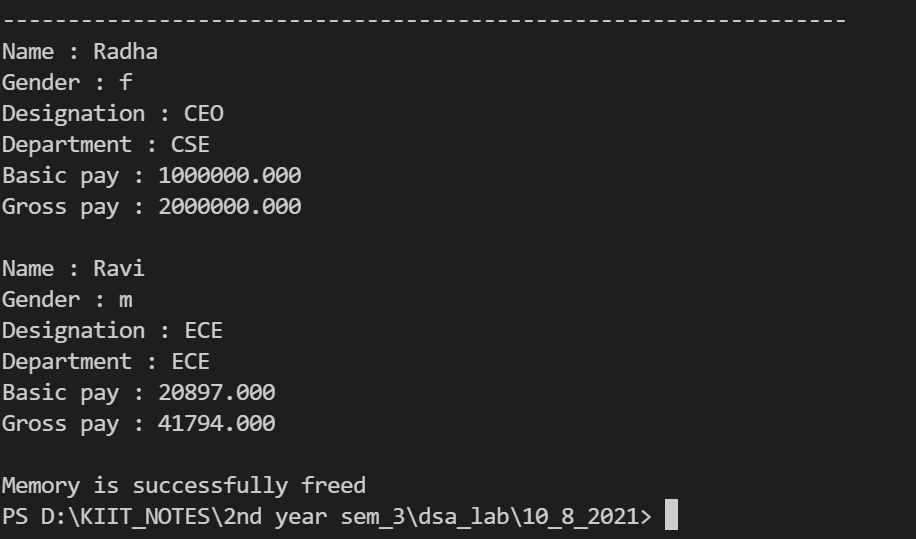
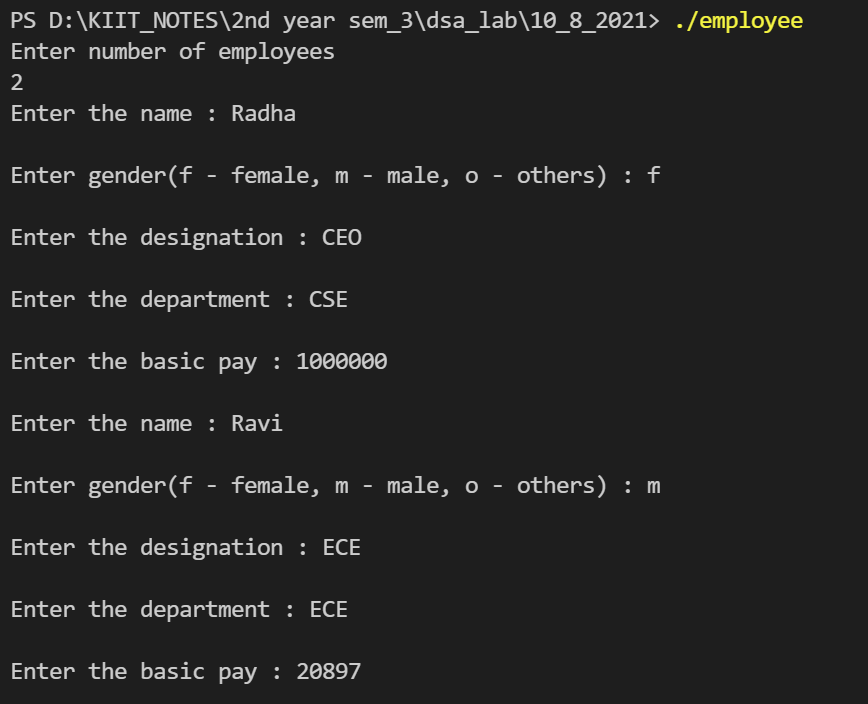
free(arr[i]);

}

puts("Memory is successfully freed");

return 0;

}



***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;

}

