LAB TASK

TASK 01:

Write a C program that includes a function isSymmetric() which takes a 3x3 2D array matrix, returns 1 if the matrix is symmetric and 0 otherwise. A symmetric matrix is equal to its transpose, meaning that the element at position (i,j) is the same as the element at (j,i) for all I and j.

SOURCE CODE:

**#include<stdio.h>**

**int isSymmetric(int arr[3][3]);**

**int main()**

**{**

**int arr[3][3];**

**int i,j;**

**printf("Enter elements of matrix:\n");**

**for(i=0;i<3;i++)**

**{**

**for(j=0;j<3;j++)**

**scanf("%d",&arr[i][j]);**

**}**

**printf("Matrix is:\n");**

**for(i=0;i<3;i++)**

**{**

**for(j=0;j<3;j++)**

**printf("%d\t",arr[i][j]);**

**printf("\n");**

**}**

**int r;**

**r = isSymmetric(arr);**

**if(r==0)**

**printf("Matrix is not symmetric");**

**else**

**printf("Matrix is symmetric");**

**}**

**int isSymmetric(int arr[3][3]){**

**int i,j;**

**int isymmetric =1;**

**int tr[3][3];**

**for(i=0;i<3;i++)**

**{**

**for(j=0;j<3;j++)**

**tr[j][i] = arr[i][j];**

**}**

**for(i=0;i<3;i++)**

**{**

**for(j=0;j<3;j++)**

**{**

**if(arr[i][j]!=tr[i][j])**

**{**

**return 0;**

**}**

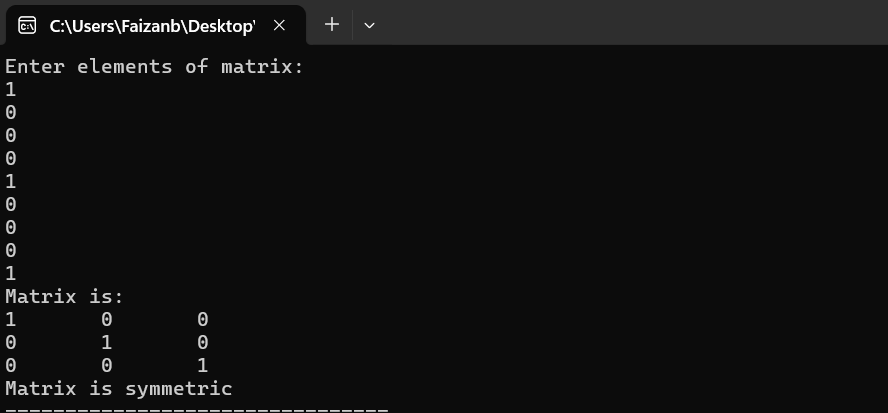
**}**

**}**

**return 1;**

**}**

**OUTPUT**



TASK 02:

Write a C program that includes a function findSaddlePoint() which takes a 3x3 2D array matrix.A saddle point is an element that is the smallest in its row and the largest in its column. The function should return the row and column indices of a saddle point if one exists, or indicate if there is no saddle point.

SOURCE CODE:

**#include <stdio.h>**

**void findsaddlepoint(int arr[3][3]);**

**int main() {**

**int arr[3][3];**

**int i,j;**

**printf("Enter elemnts of array:\n");**

**for(i=0;i<3;i++){**

**for(j=0;j<3;j++)**

**scanf("%d",&arr[i][j]);**

**}**

**for(i=0;i<3;i++){**

**for(j=0;j<3;j++)**

**printf("%d\t",arr[i][j]);**

**printf("\n");**

**}**

**findsaddlepoint(arr);**

**}**

**void findsaddlepoint(int arr[3][3])**

**{**

**int i,j,k,f;**

**for (i = 0; i < 3; i++) {**

**int min = arr[i][0];**

**int c = 0;**

**for (j = 0; j < 3; j++) {**

**if (arr[i][j] < min) {**

**min = arr[i][j];**

**c= j;**

**}**

**}**

**int max = 0;**

**for (k = 0; k <3; k++) {**

**if (arr[k][c] >max) {**

**max = arr[k][c];**

**}**

**}**

**if (min==max) {**

**printf("Saddle Point: %d at position (%d, %d)\n", min, i, c);**

**f = 1;**

**}**

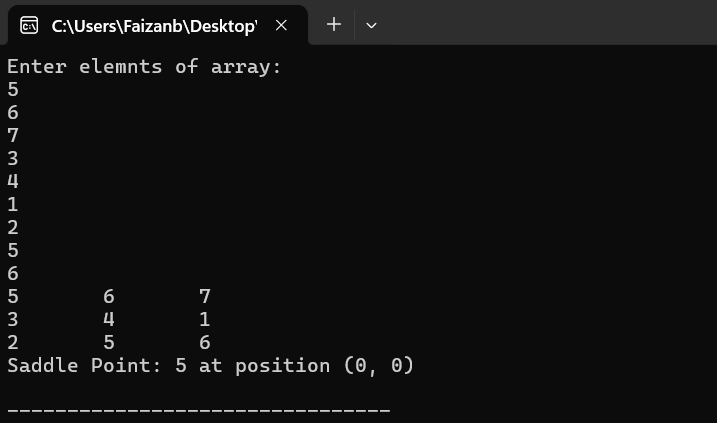
**if(f!=1)**

**printf("Matrix has no saddle point...");**

**}**

**}**

**OUTPUT**

****

TASK 03:

Write a C program that counts the frequency of each character in a sentence, ignoring spaces.The string should be stored in a 2D character array.

SOURCE CODE:

**#include<stdio.h>**

**#include<string.h>**

**#include<ctype.h>**

**int main()**

**{**

**char str[20];**

**printf("Enter a string:");**

**scanf("%[^\n]",str);**

**int freq[128] = {0};**

**int i,j;**

**for(j=0;str[j]!='\0';j++)**

**str[j] = tolower(str[j]);**

**for(j=0;str[j]!='\0';j++)**

**{**

**if(str[j]==' ')**

**{**

**continue;**

**}**

**else**

**{**

**char ch = str[j];**

**freq[(int)ch]++;**

**}**

**}**

**int k = 1;**

**for(j=0;str[j]!='\0';j++)**

**{**

**char ch = str[j];**

**freq[(int)ch];**

**if(freq[(int)ch]!=0)**

**{**

**printf("%c:%d\n",ch,freq[(int)ch]);**

**freq[(int)ch] = 0;**

**k = 0;**

**}**

**}**

**}**

**OUTPUT:**

A screenshot of a computer

Description automatically generated

TASK 04:

Write a program that uses functions to remove all spaces from a given string. Use strlen() for length.

**SOURCE CODE:**

#include<stdio.h>

#include<string.h>

int main()

{

char str[20];

printf("Enter a string:");

scanf("%[^\n]s",str);

int len;

len = strlen(str);

int i;

for(i=0;i<len;i++)

{

if(str[i]==' ')

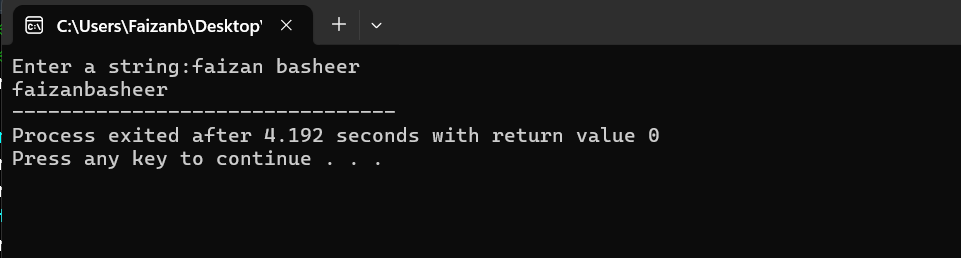
continue;

printf("%c",str[i]);

}

}

OUTPUT:



TASK 05:

Write a program that defines a function to sort each row of a 2D character array alphabetically.Prompt the user to enter 4 strings, each with a maximum of 6 characters. The function should sort each row individually in alphabetical order.

SOURCE CODE:

#include<stdio.h>

#include<string.h>

void sortrow(char str[4][6]);

int main()

{

char str[4][6];

printf("Enter 4 strings:\n");

int i;

for(i=0;i<4;i++)

{

scanf("%s",str[i]);

}

sortrow(str);

}

void sortrow(char str[4][6])

{

int i,j,k;

char t;

for(i=0;i<4;i++)

{

for(j=0;j<strlen(str[i]);j++)

{

for(k=j+1;k<strlen(str[i]);k++)

{

if(str[i][j]>str[i][k])

{

t=str[i][j];

str[i][j]=str[i][k];

str[i][k]=t;

}

}

}

}

printf("Alphabetically sorted row:\n");

for(i=0;i<4;i++)

{

printf("%s\n",str[i]);

}

}

**OUTPUT**

**A screenshot of a computer

Description automatically generated**

TASK 06:

Write a program that defines a function to find and replace a character in a 2D character array.Prompt the user to enter a 3x3 grid of characters, along with the character to find and the character to replace it with. The function should take the 2D array, the character to find, and the replacement character as arguments. Display the modified 2D array.

SOURCE CODE:

**#include<stdio.h>**

**void char\_replacement(char grid[3][3],char find,char replace);**

**int main()**

**{**

**char grid[3][3];**

**int i,j;**

**printf("Enter 9 character\n");**

**for (i =0;i<3; i++){**

**for(j=0;j<3;j++)**

**scanf(" %c",&grid[i][j]);}**

**for (i =0;i<3; i++){**

**for(j = 0;j<3;j++)**

**printf("%c\t",grid[i][j]);**

**printf("\n");}**

**char find,replace;**

**printf("Enter a character to find:");**

**scanf(" %c",&find);**

**printf("Enter a character to replace with :");**

**scanf(" %c",&replace);**

**char\_replacement(grid,find,replace);**

**}**

**void char\_replacement(char grid[3][3],char find,char replace){**

**int i,j;**

**int found = 0;**

**for (i =0;i<3; i++){**

**for(j = 0;j<3;j++)**

**if(grid[i][j]==find){**

**found = 1;**

**grid[i][j] = replace;**

**}**

**}**

**if(found==0)**

**{**

**printf("Character %c not found in grid.",find);**

**return;**

**}**

**printf("Update grid:\n");**

**for (i =0;i<3; i++)**

**{**

**for(j = 0;j<3;j++)**

**printf("%c\t",grid[i][j]);**

**printf("\n");}**

**}**

**OUTPUT:**

**A screenshot of a computer screen

Description automatically generated**