



Green University

ASSIGNMENT

SHEET

Section : DE-221

Name : Khondokar Saim

ID : 221902353

Lecturer : Mahbubur Rahman

Subject : CSE-103

Date : 11 / 09 / 2022

Problem-1 :

➤ C Program to find out number of total characters in a string

CODE :

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

int main()
{
    char str[100]; //character array
    int i,totChar; //variable declaration

    totChar=0; //variable initialization

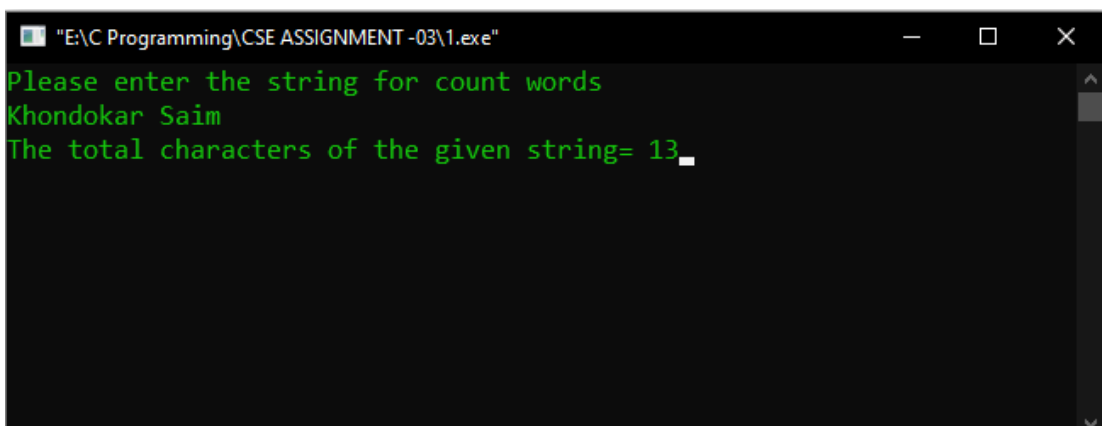
    printf("Please enter the string for count words\n");
    gets(str); //get and store string from user

    for(i=0; str[i] != '\0'; i++)
    {
        if(str[i]!=' ') // this condition is used to avoid counting space
        {
            totChar++; //totChar=totChar+1
        }
    }

    printf("The total characters of the given string= %d",totChar);
    getch(); //display total characters of the string

    return 0;
}
```

RESULT :



```
"E:\C Programming\CSE ASSIGNMENT -03\1.exe"
Please enter the string for count words
Khondokar Saim
The total characters of the given string= 13_
```

Problem- 2 :

➤ C Program to compare two strings (same or not)

CODE :

```
#include <stdio.h>
#include <string.h>
int main()
{
    char a[100], b[100];

    printf("Enter a 1st string = ");
    gets(a);

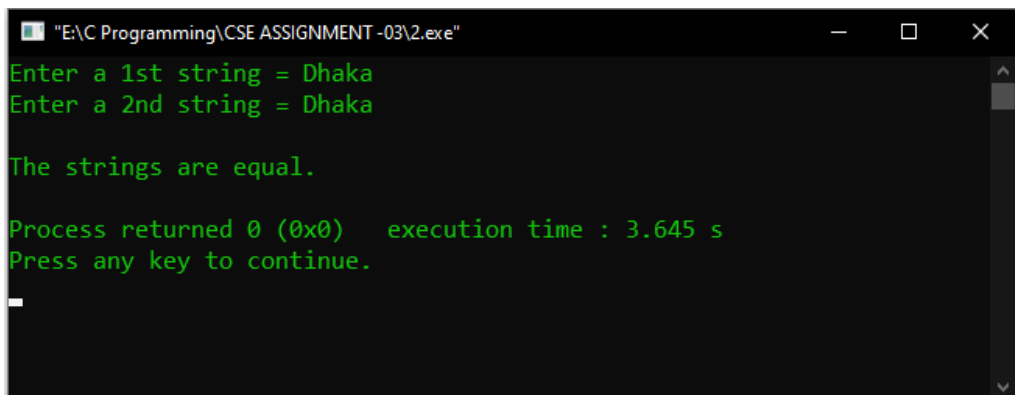
    printf("Enter a 2nd string = ");
    gets(b);

    if (strcmp(a,b) == 0)
        printf("\nThe strings are equal.\n");
    else
        printf("\nThe strings are not equal.\n");

    return 0;
}
```

RESULT :

Part - 1

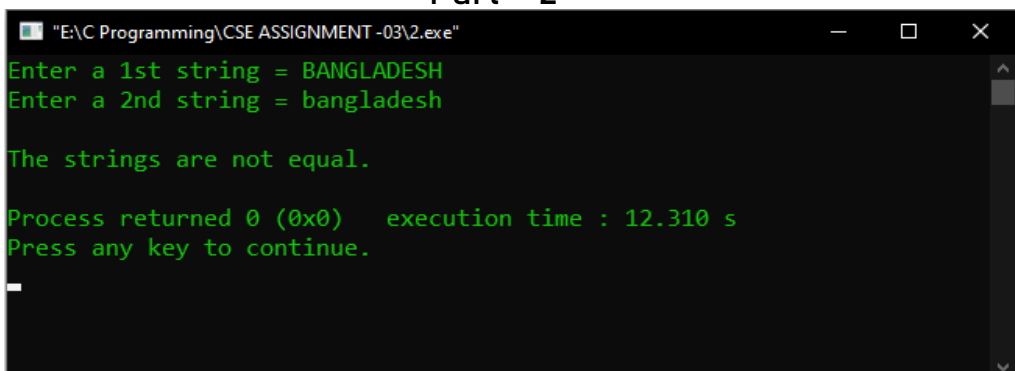
A screenshot of a Windows command prompt window titled "E:\C Programming\CSE ASSIGNMENT -03\2.exe". The window shows the execution of a C program. The user enters "Dhaka" for the first string and "Dhaka" for the second string. The program outputs "The strings are equal." followed by "Process returned 0 (0x0) execution time : 3.645 s" and "Press any key to continue." The cursor is on a new line below the prompt.

```
"E:\C Programming\CSE ASSIGNMENT -03\2.exe"
Enter a 1st string = Dhaka
Enter a 2nd string = Dhaka

The strings are equal.

Process returned 0 (0x0) execution time : 3.645 s
Press any key to continue.
_
```

Part - 2

A screenshot of a Windows command prompt window titled "E:\C Programming\CSE ASSIGNMENT -03\2.exe". The window shows the execution of the same C program. The user enters "BANGLADESH" for the first string and "bangladesh" for the second string. The program outputs "The strings are not equal." followed by "Process returned 0 (0x0) execution time : 12.310 s" and "Press any key to continue." The cursor is on a new line below the prompt.

```
"E:\C Programming\CSE ASSIGNMENT -03\2.exe"
Enter a 1st string = BANGLADESH
Enter a 2nd string = bangladesh

The strings are not equal.

Process returned 0 (0x0) execution time : 12.310 s
Press any key to continue.
_
```

Problem- 3 :

➤ C Program to use the functions strcpy(),strcmp(),stccat(),strncpy()

CODE :

1. strcpy() :

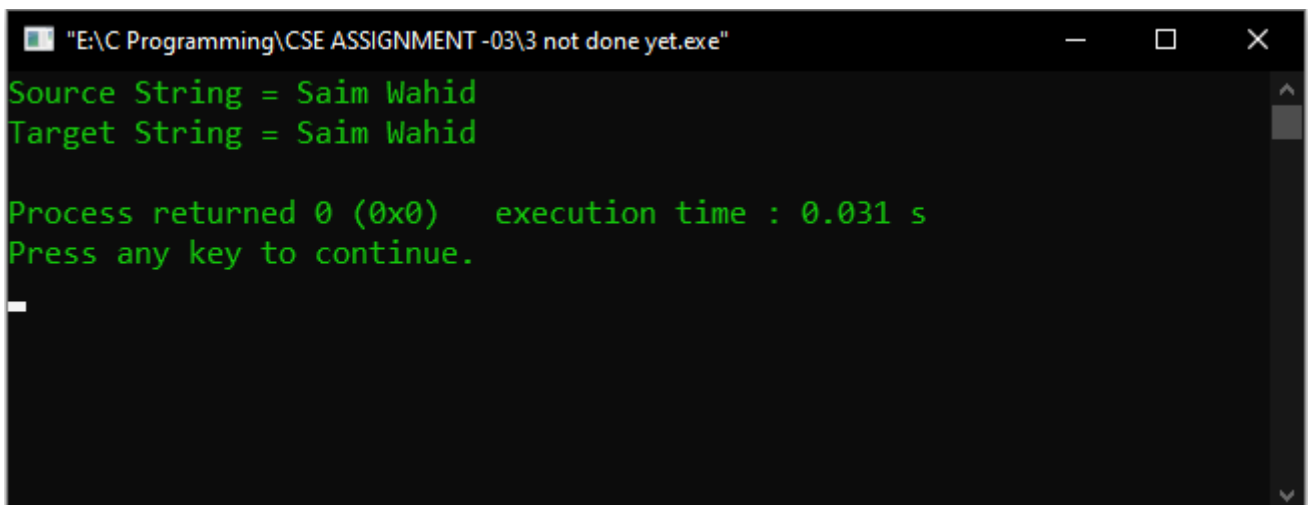
strcpy C function copies the contents of one string into another. The base addresses of the source and target strings should be supplied to this function.

```
#include<stdio.h>
int main()
{
    char source[] = "Saim Wahid" ;
    char target[20] ;

    /* This function copies the base address of source to target */
    strcpy (target, source) ;

    printf ("Source String = %s", source) ;
    printf ("\nTarget String = %s\n", target) ;
    return 0;
}
```

RESULT :



```
"E:\C Programming\CSE ASSIGNMENT -03\3 not done yet.exe"
Source String = Saim Wahid
Target String = Saim Wahid

Process returned 0 (0x0)   execution time : 0.031 s
Press any key to continue.
_
```

2. strcmp() :

The strcmp() function is used to compare two strings two strings str1 and str2. If two strings are same then strcmp() returns 0, otherwise, it returns a non-zero value.

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

int main()
{
    char strg1[50], strg2[50];

    printf("Enter first string: ");
    gets(strg1);

    printf("Enter second string: ");
    gets(strg2);

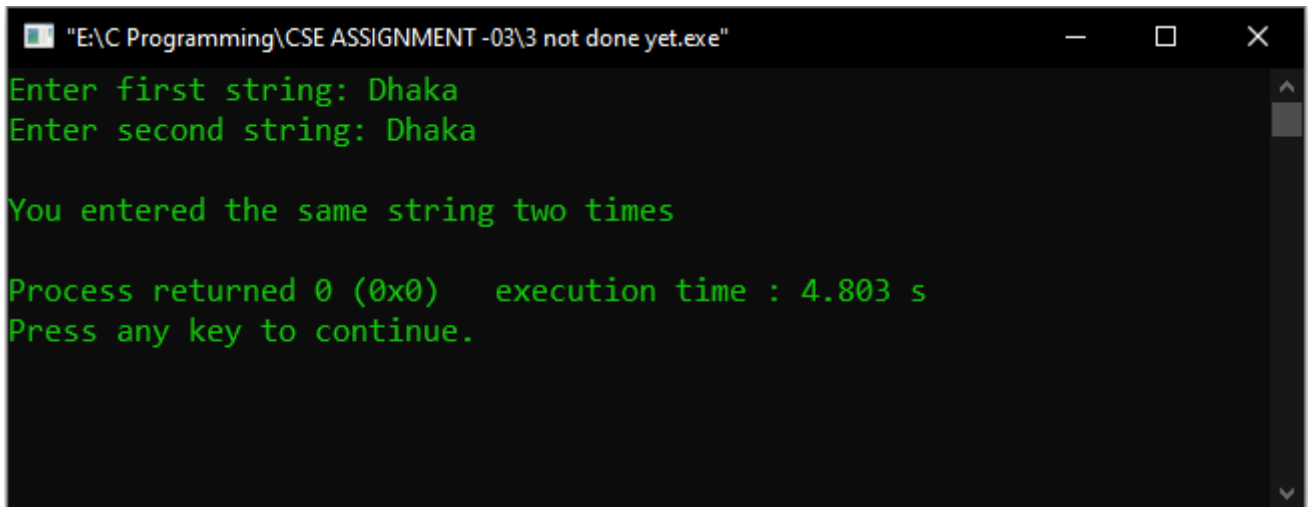
    if(strcmp(strg1, strg2)==0)
    {
        printf("\nYou entered the same string two times\n");
    }

    else
    {
        printf("\nEntered strings are not same!\n");
    }

    // signal to operating system program ran fine
    return 0;
}
```

RESULT :

Part -1

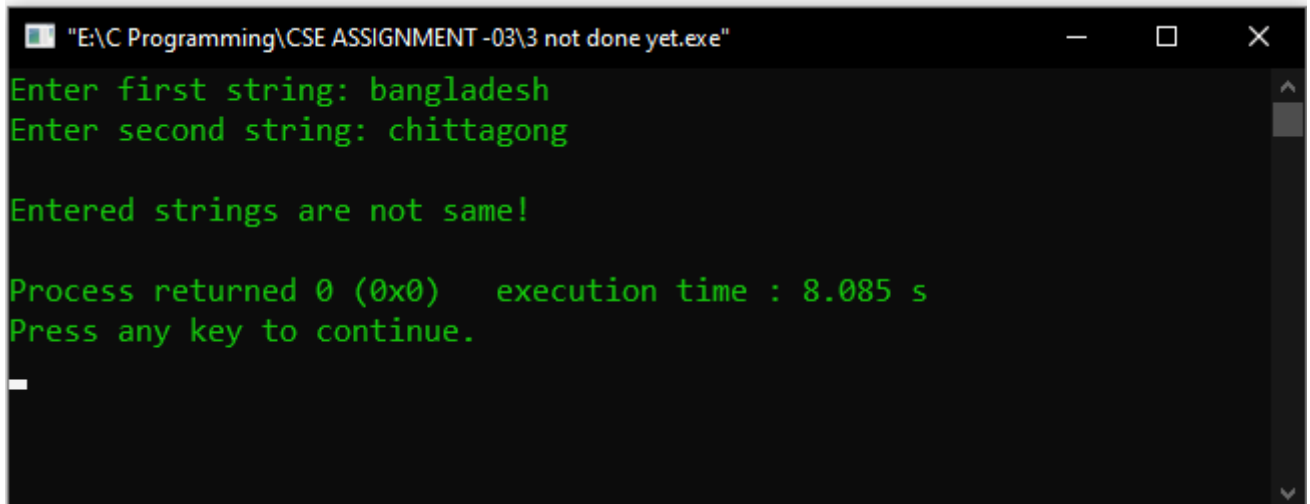


```
"E:\C Programming\CSE ASSIGNMENT -03\3 not done yet.exe"
Enter first string: Dhaka
Enter second string: Dhaka

You entered the same string two times

Process returned 0 (0x0)   execution time : 4.803 s
Press any key to continue.
```

Part -2



```
"E:\C Programming\CSE ASSIGNMENT -03\3 not done yet.exe"
Enter first string: bangladesh
Enter second string: chittagong

Entered strings are not same!

Process returned 0 (0x0)   execution time : 8.085 s
Press any key to continue.
```

3. strcat() :

The strcat() function concatenates the destination string and the source string, and the result is stored in the destination string.

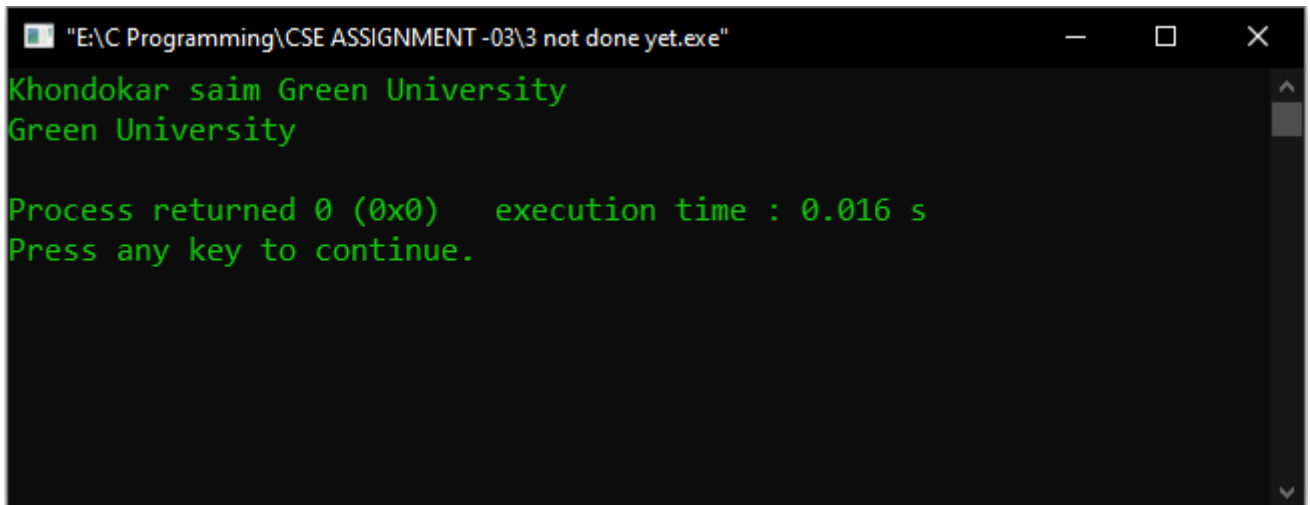
```
//strccat
#include <stdio.h>
#include <string.h>
int main() {
    char str1[100] = "Khondokar saim ", str2[] = "Green University";

    // concatenates str1 and str2
    // the resultant string is stored in str1.
    strcat(str1, str2);

    puts(str1);
    puts(str2);

    return 0;
}
```

RESULT :

A screenshot of a Windows command prompt window. The title bar at the top reads "E:\C Programming\CSE ASSIGNMENT -03\3 not done yet.exe". The window has standard Windows window controls (minimize, maximize, close) on the right. The command prompt shows the output of the program in green text: "Khondokar saim Green University" on the first line and "Green University" on the second line. Below this, it says "Process returned 0 (0x0) execution time : 0.016 s" and "Press any key to continue." at the bottom. A vertical scrollbar is visible on the right side of the command prompt area.

```
"E:\C Programming\CSE ASSIGNMENT -03\3 not done yet.exe"
Khondokar saim Green University
Green University

Process returned 0 (0x0)   execution time : 0.016 s
Press any key to continue.
```

4. strncpy() :

strncpy() function copies portion of contents of one string into another string.

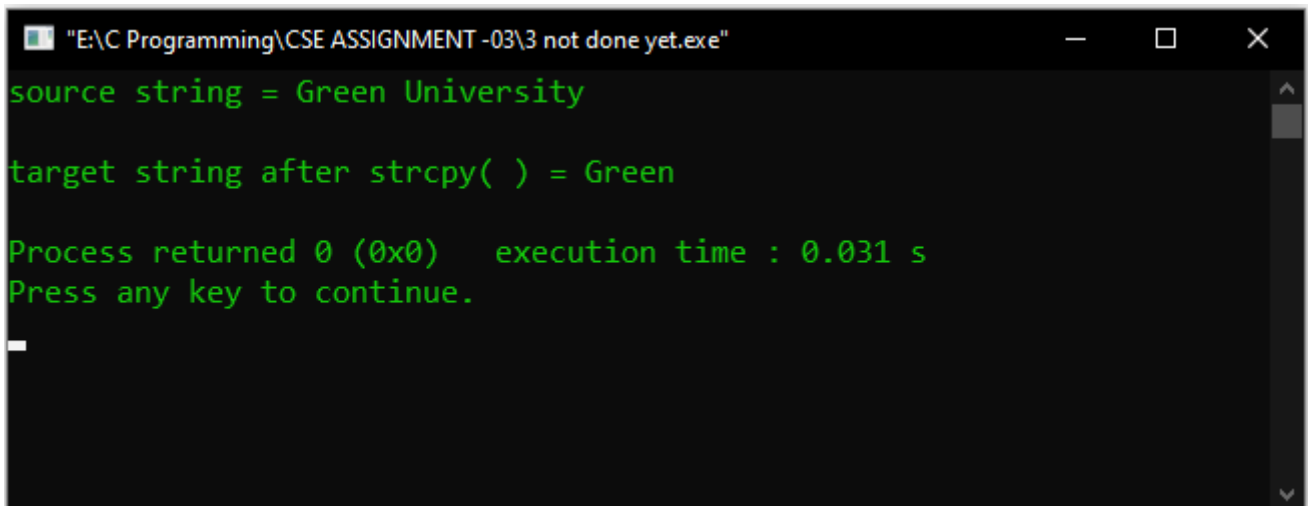
```
//strncpy
#include <stdio.h>
#include <string.h>

int main( )
{
    char source[ ] = "Green University" ;
    char target[20]= "" ;

    printf ( "\source string = %s\n", source ) ;

    strncpy ( target, source, 5 ) ;
    printf ( "\ntarget string after strcpy( ) = %s\n", target ) ;
    return 0;
}
```

RESULT :



```
"E:\C Programming\CSE ASSIGNMENT -03\3 not done yet.exe"
source string = Green University
target string after strcpy( ) = Green
Process returned 0 (0x0)   execution time : 0.031 s
Press any key to continue.
_
```


Problem- 4 :

➤ C Program to arrays of strings

CODE :

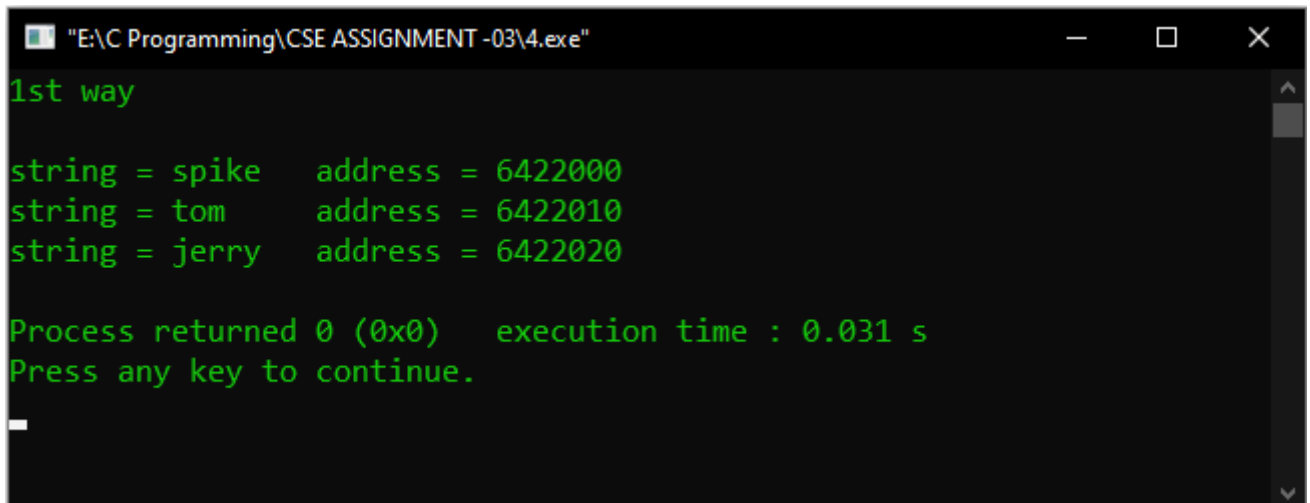
```
#include<stdio.h>
int main()
{
    int i;
    char ch_arr[3][10] = {
        "spike",
        "tom",
        "jerry"
    };

    printf("1st way \n\n");

    for(i = 0; i < 3; i++)
    {
        printf("string = %s \t address = %u\n", ch_arr + i, ch_arr + i);
    }

    return 0;
}
```

RESULT:



```
"E:\C Programming\CSE ASSIGNMENT -03\4.exe"
1st way

string = spike    address = 6422000
string = tom      address = 6422010
string = jerry    address = 6422020

Process returned 0 (0x0)    execution time : 0.031 s
Press any key to continue.
_
```

Problem- 5 :

➤ C Program to access each character of a string and print separately

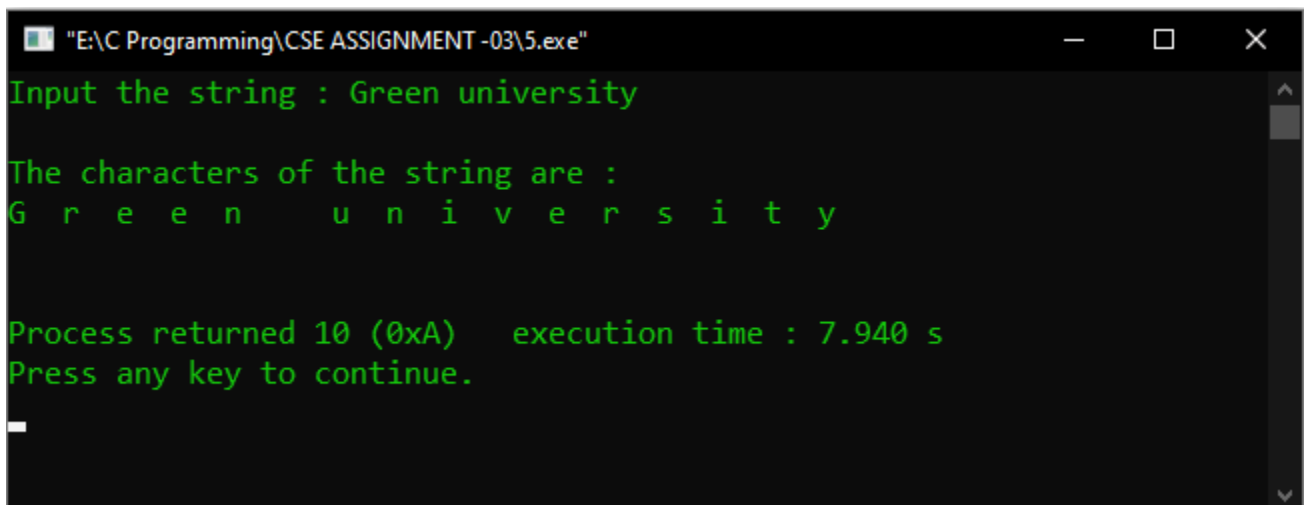
CODE :

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

void main()
{
    char str[100]; /* Declares a string of size 100 */
    int l= 0;

    printf("Input the string : ");
    fgets(str, sizeof str, stdin);
    printf("\nThe characters of the string are : \n");
    while(str[l]!='\0')
    {
        printf("%c ", str[l]);
        l++;
    }
    printf("\n");
}
```

RESULT :



```
"E:\C Programming\CSE ASSIGNMENT -03\5.exe"
Input the string : Green university

The characters of the string are :
G r e e n   u n i v e r s i t y

Process returned 10 (0xA)   execution time : 7.940 s
Press any key to continue.
_
```

Problem- 6 :

➤ C Program Caesar cipher encryption and decryption

CODE :

Encryption

```
#include<stdio.h>
int main()
{
    char message[100], ch;
    int i, key;

    printf("Enter a message to encrypt: ");
    gets(message);

    printf("Enter key: ");
    scanf("%d", &key);

    for(i = 0; message[i] != '\0'; ++i){
        ch = message[i];

        if(ch >= 'a' && ch <= 'z'){
            ch = ch + key;

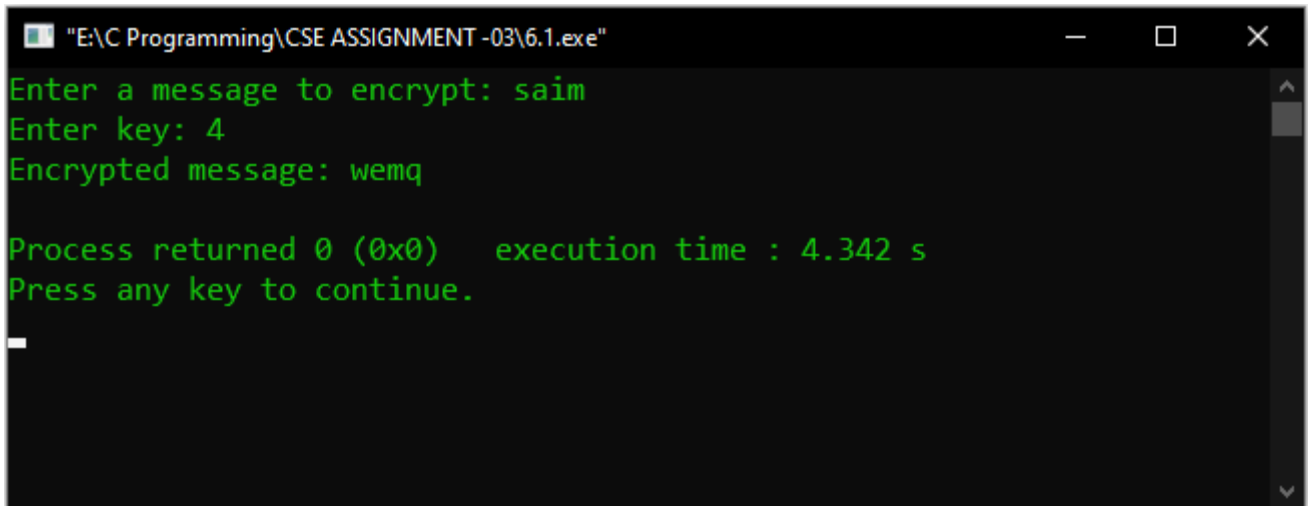
            if(ch > 'z'){
                ch = ch - 'z' + 'a' - 1;
            }
            message[i] = ch;
        }

        else if(ch >= 'A' && ch <= 'Z'){
            ch = ch + key;

            if(ch > 'Z'){
                ch = ch - 'Z' + 'A' - 1;
            }
            message[i] = ch;
        }

        printf("Encrypted message: %s\n", message);
        return 0;
    }
```

RESULT:



```
"E:\C Programming\CSE ASSIGNMENT -03\6.1.exe"
Enter a message to encrypt: saim
Enter key: 4
Encrypted message: wemq

Process returned 0 (0x0)   execution time : 4.342 s
Press any key to continue.
```

Decryption

```
#include<stdio.h>
int main()
{
    char message[100], ch;
    int i, key;
    printf("Enter a message to decrypt: ");
    gets(message);

    printf("Enter key: ");
    scanf("%d", &key);

    for(i = 0; message[i] != '\0'; ++i){
        ch = message[i];

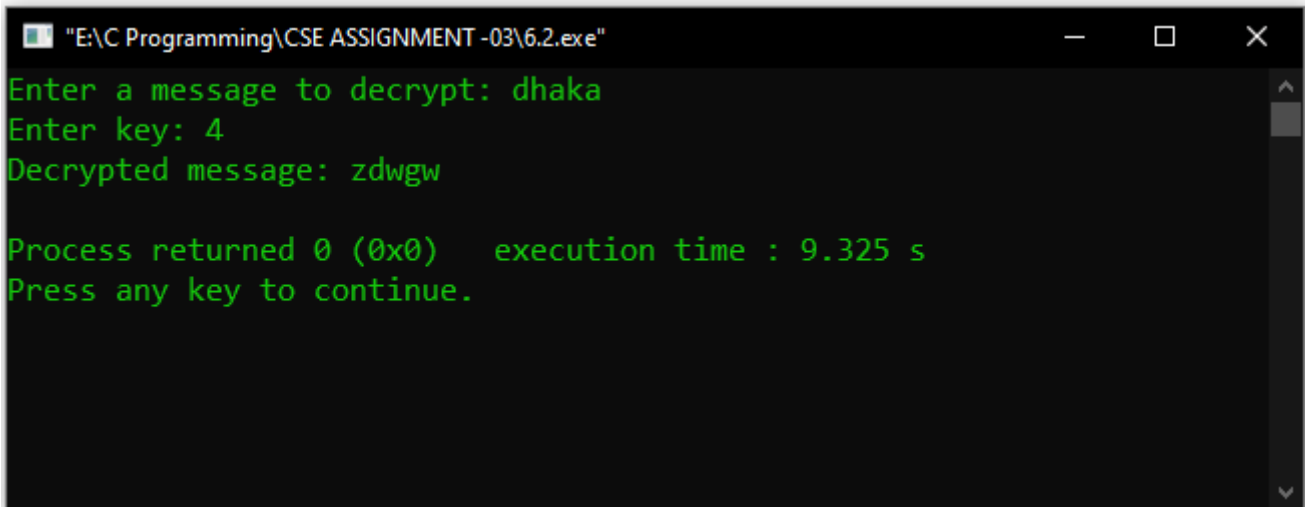
        if(ch >= 'a' && ch <= 'z'){
            ch = ch - key;

            if(ch < 'a'){
                ch = ch + 'z' - 'a' + 1;
            }
            message[i] = ch;
        }

        else if(ch >= 'A' && ch <= 'Z'){
            ch = ch - key;
```

```
    if(ch < 'A'){  
        ch = ch + 'Z' - 'A' + 1;  
    }  
    message[i] = ch;  
}  
}  
printf("Decrypted message: %s\n", message);  
return 0;  
}
```

RESULT:



```
"E:\C Programming\CSE ASSIGNMENT -03\6.2.exe"  
Enter a message to decrypt: dhaka  
Enter key: 4  
Decrypted message: zdwgw  
  
Process returned 0 (0x0)   execution time : 9.325 s  
Press any key to continue.
```

Problem- 7 :

- C Program Dynamic Memory Allocation, use of malloc(),calloc(), free(), realloc()

CODE :

1. malloc() :

The “malloc” or “memory allocation” method in C is used to dynamically allocate a single large block of memory with the specified size.

```
//malloc
#include <stdio.h>
#include <stdlib.h>
int main()
{
    int* ptr;
    int n, i;

    // Get the number of elements for the array
    printf("Enter number of elements:");
    scanf("%d",&n);
    printf("Entered number of elements: %d\n", n);

    // Dynamically allocate memory using malloc()
    ptr = (int*)malloc(n * sizeof(int));

    // Check if the memory has been successfully
    // allocated by malloc or not
    if (ptr == NULL) {
        printf("Memory not allocated.\n");
        exit(0);
    }
    else {

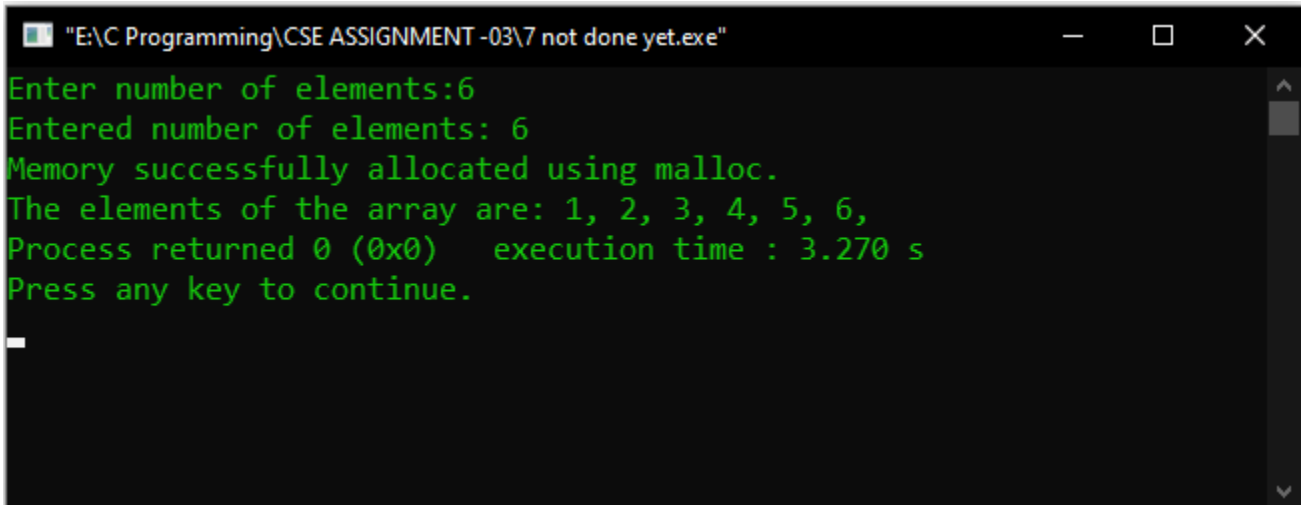
        // Memory has been successfully allocated
        printf("Memory successfully allocated using malloc.\n");

        // Get the elements of the array
        for (i = 0; i < n; ++i) {
            ptr[i] = i + 1;
        }
    }
}
```

```
// Print the elements of the array
printf("The elements of the array are: ");
for (i = 0; i < n; ++i) {
    printf("%d, ", ptr[i]);
}

return 0;
}
```

RESULT:



```
"E:\C Programming\CSE ASSIGNMENT -03\7 not done yet.exe"
Enter number of elements:6
Entered number of elements: 6
Memory successfully allocated using malloc.
The elements of the array are: 1, 2, 3, 4, 5, 6,
Process returned 0 (0x0)   execution time : 3.270 s
Press any key to continue.
_
```

2. calloc() :

“calloc” or “contiguous allocation” method in C is used to dynamically allocate the specified number of blocks of memory of the specified type.

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

int main()
{

    // This pointer will hold the
    // base address of the block created
    int* ptr;
    int n, i;

    // Get the number of elements for the array
    n = 5;
    printf("Enter number of elements: %d\n", n);

    // Dynamically allocate memory using calloc()
    ptr = (int*)calloc(n, sizeof(int));

    // Check if the memory has been successfully
    // allocated by calloc or not
    if (ptr == NULL) {
        printf("Memory not allocated.\n");
        exit(0);
    }
    else {

        // Memory has been successfully allocated
        printf("Memory successfully allocated using calloc.\n");

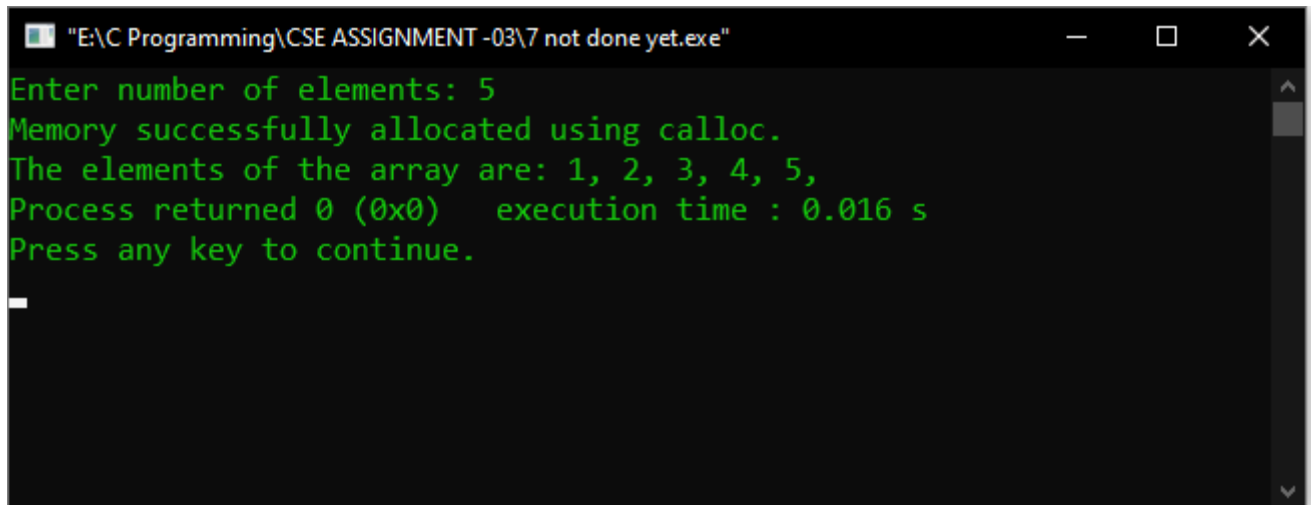
        // Get the elements of the array
        for (i = 0; i < n; ++i) {
            ptr[i] = i + 1;
        }

        // Print the elements of the array
        printf("The elements of the array are: ");
        for (i = 0; i < n; ++i) {
            printf("%d, ", ptr[i]);
        }
    }
}
```



```
    return 0;  
}
```

RESULT:



```
"E:\C Programming\CSE ASSIGNMENT -03\7 not done yet.exe"  
Enter number of elements: 5  
Memory successfully allocated using calloc.  
The elements of the array are: 1, 2, 3, 4, 5,  
Process returned 0 (0x0)    execution time : 0.016 s  
Press any key to continue.  
_
```

3. free() :

“free” method in C is used to dynamically de-allocate the memory. The memory allocated using functions malloc() and calloc() is not de-allocated on their own. Hence the free() method is used, whenever the dynamic memory allocation takes place. It helps to reduce wastage of memory by freeing it.

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

int main()
{

    // This pointer will hold the
    // base address of the block created
    int *ptr, *ptr1;
    int n, i;

    // Get the number of elements for the array
    n = 5;
    printf("Enter number of elements: %d\n", n);

    // Dynamically allocate memory using malloc()
    ptr = (int*)malloc(n * sizeof(int));

    // Dynamically allocate memory using calloc()
    ptr1 = (int*)calloc(n, sizeof(int));

    // Check if the memory has been successfully
    // allocated by malloc or not
    if (ptr == NULL || ptr1 == NULL) {
        printf("Memory not allocated.\n");
        exit(0);
    }
    else {

        // Memory has been successfully allocated
        printf("Memory successfully allocated using malloc.\n");

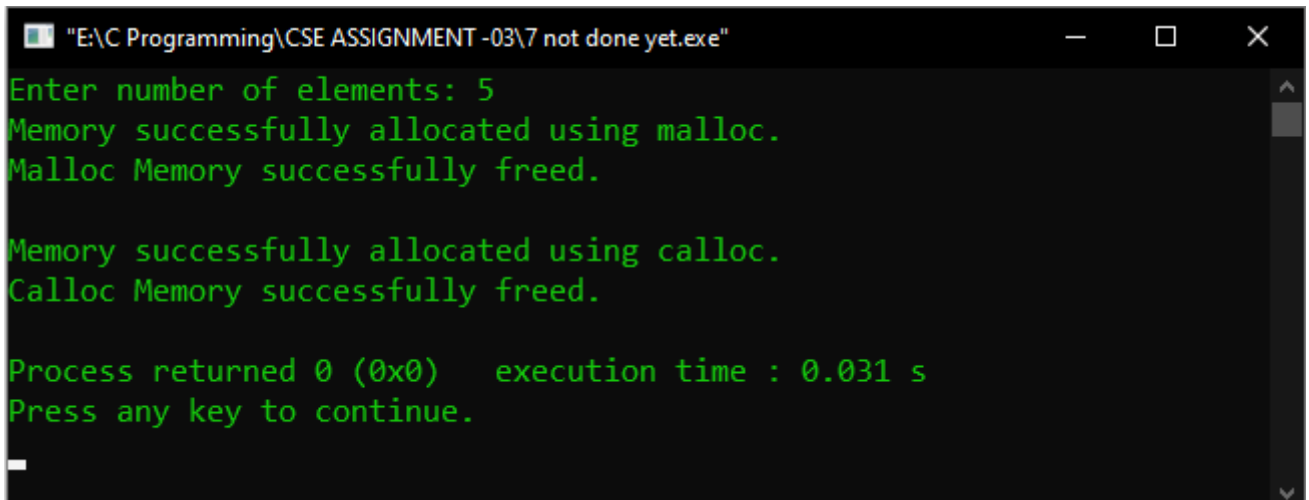
        // Free the memory
        free(ptr);
        printf("Malloc Memory successfully freed.\n");
    }
}
```

```
// Memory has been successfully allocated
printf("\nMemory successfully allocated using calloc.\n");

// Free the memory
free(ptr1);
printf("Calloc Memory successfully freed.\n");
}

return 0;
}
```

RESULT:



```
"E:\C Programming\CSE ASSIGNMENT -03\7 not done yet.exe"
Enter number of elements: 5
Memory successfully allocated using malloc.
Malloc Memory successfully freed.

Memory successfully allocated using calloc.
Calloc Memory successfully freed.

Process returned 0 (0x0)   execution time : 0.031 s
Press any key to continue.
```

4. realloc() :

“realloc” or “re-allocation” method in C is used to dynamically change the memory allocation of a previously allocated memory.

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

int main()
{

    // This pointer will hold the
    // base address of the block created
    int* ptr;
    int n, i;

    // Get the number of elements for the array
    n = 5;
    printf("Enter number of elements: %d\n", n);

    // Dynamically allocate memory using calloc()
    ptr = (int*)calloc(n, sizeof(int));

    // Check if the memory has been successfully
    // allocated by malloc or not
    if (ptr == NULL) {
        printf("Memory not allocated.\n");
        exit(0);
    }
    else {

        // Memory has been successfully allocated
        printf("Memory successfully allocated using calloc.\n");

        // Get the elements of the array
        for (i = 0; i < n; ++i) {
            ptr[i] = i + 1;
        }

        // Print the elements of the array
        printf("The elements of the array are: ");
        for (i = 0; i < n; ++i) {
            printf("%d, ", ptr[i]);
        }
    }
}
```

```
// Get the new size for the array
n = 10;
printf("\n\nEnter the new size of the array: %d\n", n);

// Dynamically re-allocate memory using realloc()
ptr = realloc(ptr, n * sizeof(int));

// Memory has been successfully allocated
printf("Memory successfully re-allocated using realloc.\n");

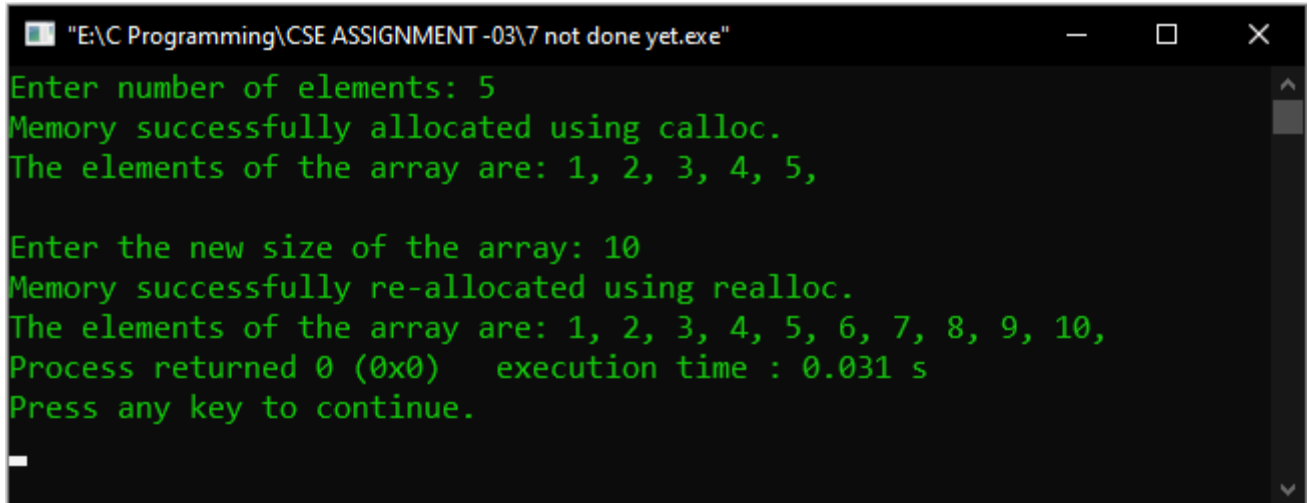
// Get the new elements of the array
for (i = 5; i < n; ++i) {
    ptr[i] = i + 1;
}

// Print the elements of the array
printf("The elements of the array are: ");
for (i = 0; i < n; ++i) {
    printf("%d, ", ptr[i]);
}

free(ptr);
}

return 0;
}
```

RESULT:



```
"E:\C Programming\CSE ASSIGNMENT -03\7 not done yet.exe"
Enter number of elements: 5
Memory successfully allocated using calloc.
The elements of the array are: 1, 2, 3, 4, 5,

Enter the new size of the array: 10
Memory successfully re-allocated using realloc.
The elements of the array are: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
Process returned 0 (0x0)   execution time : 0.031 s
Press any key to continue.
_
```

Problem- 8 :

➤ C Program find maximum or minimum value using pointer

CODE :

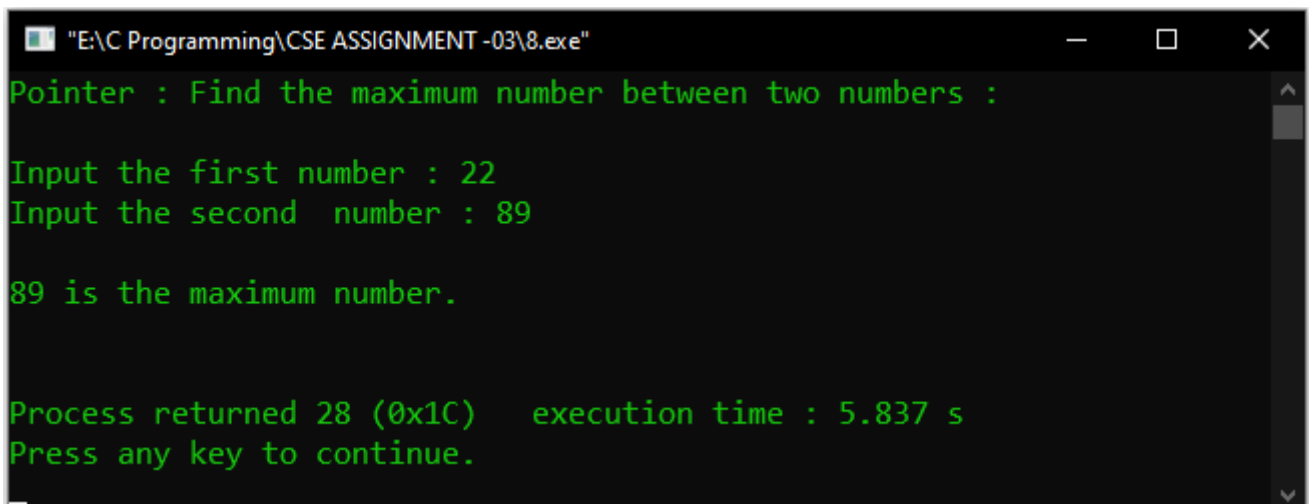
```
#include <stdio.h>
#include <stdlib.h>
void main()
{
    int fno,sno,*ptr1=&fno,*ptr2=&sno;

    printf("Pointer : Find the maximum number between two numbers
:\n");

    printf("\nInput the first number : ");
    scanf("%d", ptr1);
    printf("Input the second  number : ");
    scanf("%d", ptr2);

    if(*ptr1>*ptr2)
    {
        printf("\n%d is the maximum number.\n\n",*ptr1);
    }
    else
    {
        printf("\n%d is the maximum number.\n\n",*ptr2);
    }
}
```

RESULT:



```
"E:\C Programming\CSE ASSIGNMENT -03\8.exe"
Pointer : Find the maximum number between two numbers :

Input the first number : 22
Input the second  number : 89

89 is the maximum number.

Process returned 28 (0x1C)    execution time : 5.837 s
Press any key to continue.
```

Problem- 9 :

➤ C Program calculate sum of an array using a pointer of that array.

CODE :

```
#include<stdio.h>
int main()
{
    int array[5];
    int i, sum=0;
    int *ptr;

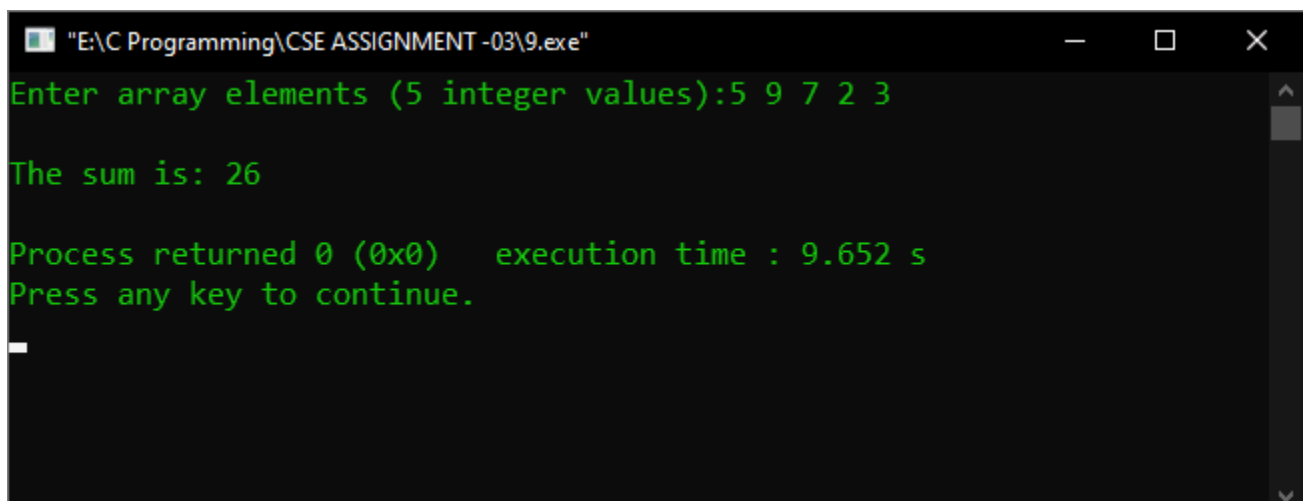
    printf("Enter array elements (5 integer values):");
    for(i=0;i<5;i++)
        scanf("%d",&array[i]);

    ptr = array;

    for(i=0;i<5;i++)
    {
        sum = sum + *ptr;
        ptr++;
    }

    printf("\nThe sum is: %d\n",sum);
}
```

RESULT :

A screenshot of a Windows command prompt window titled "E:\C Programming\CSE ASSIGNMENT -03\9.exe". The window has standard Windows window controls (minimize, maximize, close) in the top right corner. The text inside the window is green on a black background. It shows the prompt "Enter array elements (5 integer values):" followed by the input "5 9 7 2 3". Below that, it says "The sum is: 26". At the bottom, it displays "Process returned 0 (0x0) execution time : 9.652 s" and "Press any key to continue." with a cursor on a new line.

```
"E:\C Programming\CSE ASSIGNMENT -03\9.exe"
Enter array elements (5 integer values):5 9 7 2 3

The sum is: 26

Process returned 0 (0x0) execution time : 9.652 s
Press any key to continue.
_
```


Problem- 10 :

➤ How to initialize/declare structure, taking input and output.

1. Declaration of structure variable

Let us declare a student structure containing three fields i.e. name, roll and marks.

```
struct student
{
    char name[100];
    int roll;
    float marks;
};
```

2. initialize a structure variable

C language supports multiple ways to initialize a structure variable. You can use any of the initialization method to initialize your structure.

- Initialize using dot operator
- Value initialized structure variable
- Variant of value initialized structure variable

Initialize using dot operator :

```
// Declare structure variable
struct student stu1;
```

```
// Initialize structure members
stu1.name = "Pankaj";
stu1.roll = 12;
stu1.marks = 79.5f;
```

Value initialized structure variable :

```
// Declare and initialize structure variable
struct student stu1 = { "Pankaj", 12, 79.5f };
```

Variant of value initialized structure variable :

```
// Declare and initialize structure variable
struct student stu1 = {
.roll = 12,
.name = "Pankaj",
.marks = 79.5f
};
```

CODE :

```
//How to declare, initialize and access structures in C language
#include <stdio.h>
```

```
// Macro for default student structure initialization
#define NEW_STUDENT { "", 0, 0.0f }
```

```
// Student structure type declaration
struct student
{
    char   name[100];
    int    roll;
    float  marks;
};
```

```
int main()
{
    // Declare structure variable with default initialization
    struct student stu1 = NEW_STUDENT;

    // Read student details from user
    printf("Enter student name: ");
    gets(stu1.name);

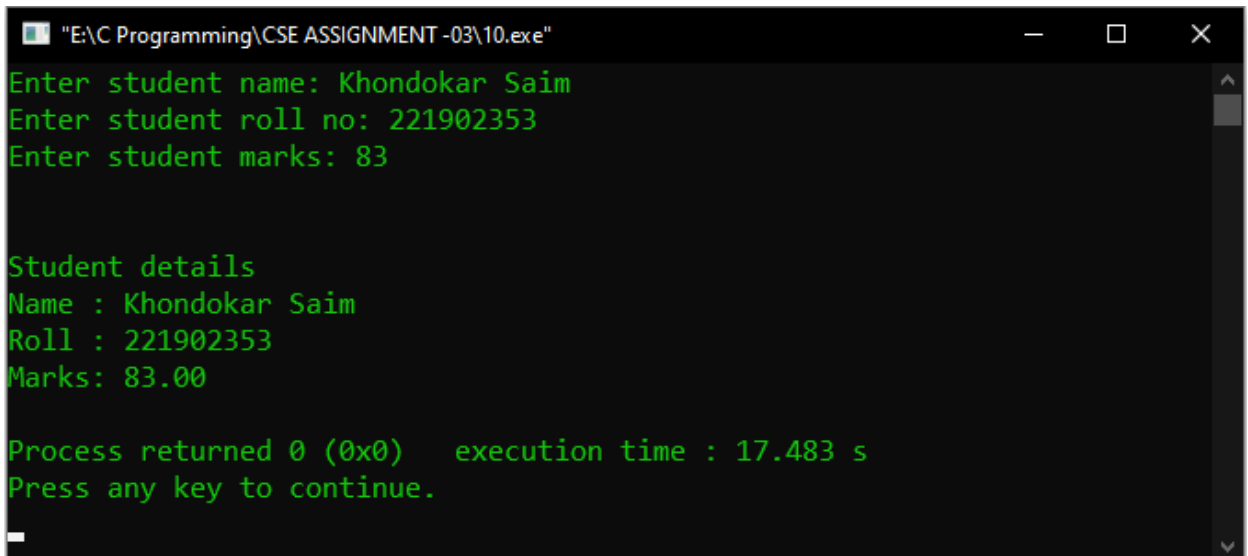
    printf("Enter student roll no: ");
    scanf("%d", &stu1.roll);

    printf("Enter student marks: ");
    scanf("%f", &stu1.marks);
```

```
// Print student details
printf("\n\nStudent details\n");
printf("Name : %s\n", stu1.name);
printf("Roll : %d\n", stu1.roll);
printf("Marks: %.2f\n", stu1.marks);

return 0;
}
```

RESULT:



```
"E:\C Programming\CSE ASSIGNMENT -03\10.exe"
Enter student name: Khondokar Saim
Enter student roll no: 221902353
Enter student marks: 83

Student details
Name : Khondokar Saim
Roll : 221902353
Marks: 83.00

Process returned 0 (0x0)   execution time : 17.483 s
Press any key to continue.
```

Problem- 11 :

➤ structure using pointer. create and assign value using pointers

CODE :

```
#include <stdio.h>

int main(void)
{
    // student structure
    struct student
    {
        char id[15];
        char firstname[64];
        char lastname[64];
        float points;
    };

    // student structure variable
    struct student std;

    // student structure pointer variable
    struct student *ptr = NULL;

    // assign std to ptr
    ptr = &std;

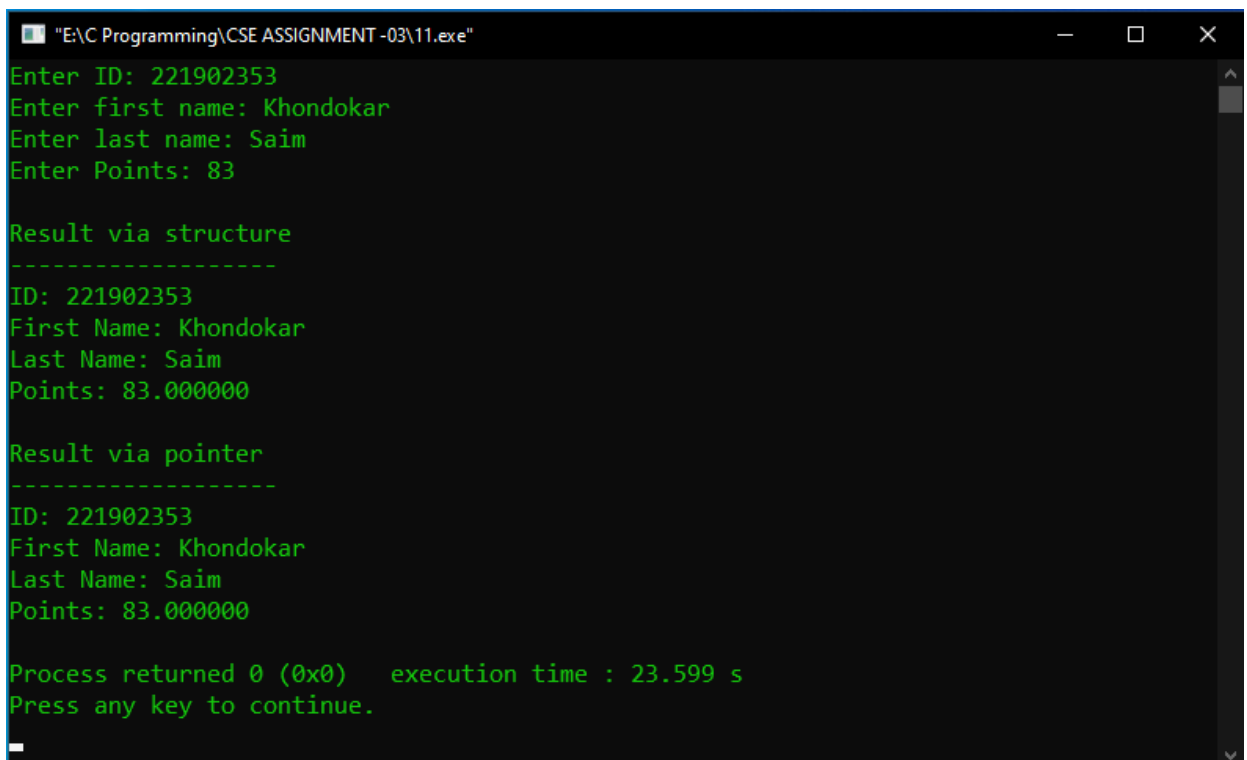
    // get student detail from user
    printf("Enter ID: ");
    scanf("%s", ptr->id);
    printf("Enter first name: ");
    scanf("%s", ptr->firstname);
    printf("Enter last name: ");
    scanf("%s", ptr->lastname);
    printf("Enter Points: ");
    scanf("%f", &ptr->points);

    // display result via structure variable
    printf("\nResult via structure");
    printf("\n-----\n");
    printf("ID: %s\n", std.id);
    printf("First Name: %s\n", std.firstname);
    printf("Last Name: %s\n", std.lastname);
    printf("Points: %f\n", std.points);
```

```
// display result via pointer variable
printf("\nResult via pointer");
printf("\n-----\n");
printf("ID: %s\n", ptr->id);
printf("First Name: %s\n", ptr->firstname);
printf("Last Name: %s\n", ptr->lastname);
printf("Points: %f\n", ptr->points);

return 0;
}
```

RESULT:



```
"E:\C Programming\CSE ASSIGNMENT -03\11.exe"
Enter ID: 221902353
Enter first name: Khondokar
Enter last name: Saim
Enter Points: 83

Result via structure
-----
ID: 221902353
First Name: Khondokar
Last Name: Saim
Points: 83.000000

Result via pointer
-----
ID: 221902353
First Name: Khondokar
Last Name: Saim
Points: 83.000000

Process returned 0 (0x0)   execution time : 23.599 s
Press any key to continue.
```

Problem- 12 :

- create a student information management system using structure where you can add info, show info, update info, delete info

CODE :

```
#include<stdlib.h>
#include<string.h>
#include<stdio.h>
struct Student
{
    int rollnumber;
    char name[100];
    char phone[100];
    float percentage;
    struct Student *next;
}* head;
void insert(int rollnumber, char* name, char* phone, float
percentage)
{
    struct Student * student = (struct Student *) malloc(sizeof(struct
Student));
    student->rollnumber = rollnumber;
    strcpy(student->name, name);
    strcpy(student->phone, phone);
    student->percentage = percentage;
    student->next = NULL;

    if(head==NULL){
        head = student;
    }
    else{
        student->next = head;
        head = student;
    }
}
```

```

void search(int rollnumber)
{
    struct Student * temp = head;
    while(temp!=NULL){
        if(temp->rollnumber==rollnumber){
            printf("Roll Number: %d\n", temp->rollnumber);
            printf("Name: %s\n", temp->name);
            printf("Phone: %s\n", temp->phone);
            printf("Percentage: %0.4f\n", temp->percentage);
            return;
        }
        temp = temp->next;
    }
    printf("Student with roll number %d is not found !!!\n", rollnumber);
}

void update(int rollnumber)
{
    struct Student * temp = head;
    while(temp!=NULL){
        if(temp->rollnumber==rollnumber){
            printf("Record with roll number %d Found !!!\n", rollnumber);
            printf("Enter new name: ");
            scanf("%s", temp->name);
            printf("Enter new phone number: ");
            scanf("%s", temp->phone);
            printf("Enter new percentage: ");
            scanf("%f",&temp->percentage);
            printf("Updation Successful!!!\n");
            return;
        }
        temp = temp->next;
    }
    printf("Student with roll number %d is not found !!!\n", rollnumber);
}

void Delete(int rollnumber)
{
    struct Student * temp1 = head;
    struct Student * temp2 = head;
    while(temp1!=NULL)
    {

```

```

if(temp1->rollnumber==rollnumber){

    printf("Record with roll number %d Found !!!\n", rollnumber);

    if(temp1==temp2){

        head = head->next;
        free(temp1);
    }
    else{

        temp2->next = temp1->next;
        free(temp1);
    }

    printf("Record Successfully Deleted !!!\n");
    return;

}
temp2 = temp1;
temp1 = temp1->next;

}
printf("Student with roll number %d is not found !!!\n", rollnumber);

}
void display()
{
    struct Student * temp = head;
    while(temp!=NULL){

        printf("Roll Number: %d\n", temp->rollnumber);
        printf("Name: %s\n", temp->name);
        printf("Phone: %s\n", temp->phone);
        printf("Percentage: %0.4f\n\n", temp->percentage);
        temp = temp->next;

    }
}
int main()
{

```



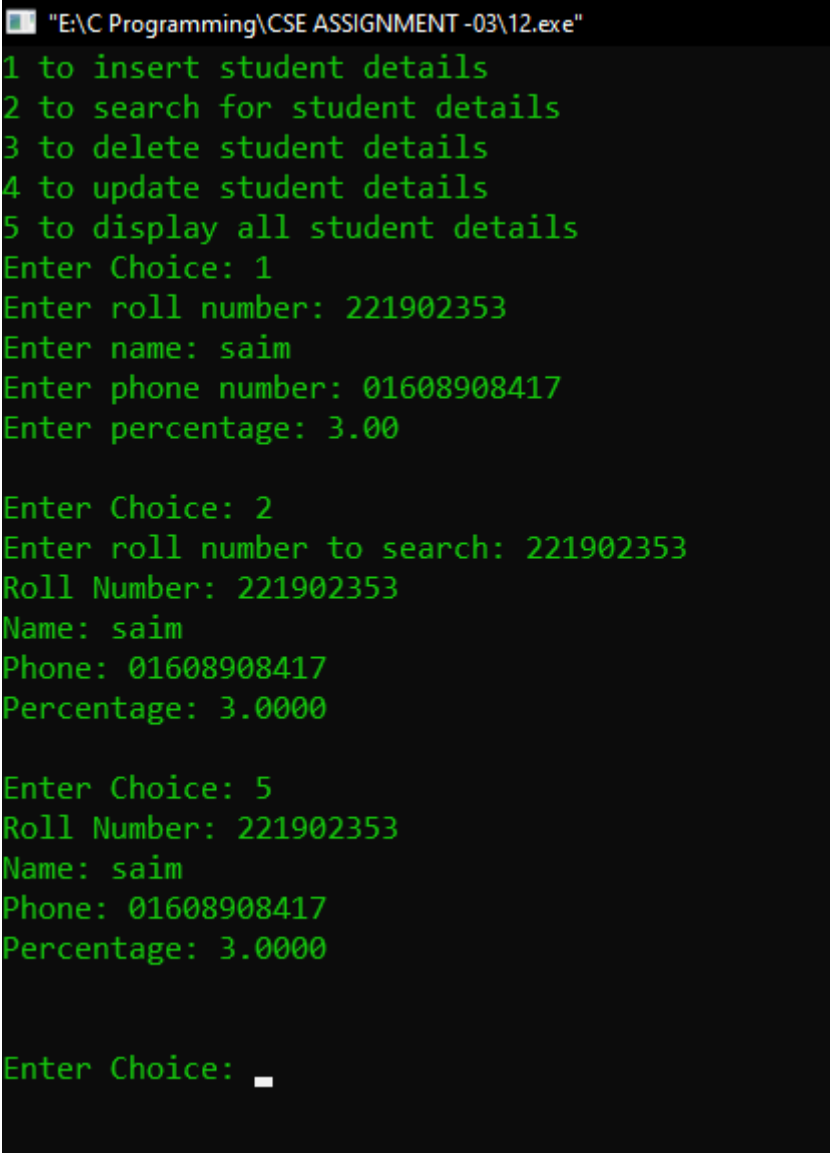
```

head = NULL;
int choice;
char name[100];
char phone[100];
int rollnumber;
float percentage;
printf("1 to insert student details\n2 to search for student
details\n3 to delete student details\n4 to update student details\n5 to
display all student details");
do
{
    printf("\nEnter Choice: ");
    scanf("%d", &choice);
    switch (choice)
    {
        case 1:
            printf("Enter roll number: ");
            scanf("%d", &rollnumber);
            printf("Enter name: ");
            scanf("%s", name);
            printf("Enter phone number: ");
            scanf("%s", phone);
            printf("Enter percentage: ");
            scanf("%f", &percentage);
            insert(rollnumber, name, phone, percentage);
            break;
        case 2:
            printf("Enter roll number to search: ");
            scanf("%d", &rollnumber);
            search(rollnumber);
            break;
        case 3:
            printf("Enter roll number to delete: ");
            scanf("%d", &rollnumber);
            Delete(rollnumber);
            break;
        case 4:
            printf("Enter roll number to update: ");
            scanf("%d", &rollnumber);
            update(rollnumber);
            break;
        case 5:
            display();
            break;
    }
}

```

```
} while (choice != 0);  
}
```

RESULT:



```
"E:\C Programming\CSE ASSIGNMENT -03\12.exe"  
1 to insert student details  
2 to search for student details  
3 to delete student details  
4 to update student details  
5 to display all student details  
Enter Choice: 1  
Enter roll number: 221902353  
Enter name: saim  
Enter phone number: 01608908417  
Enter percentage: 3.00  
  
Enter Choice: 2  
Enter roll number to search: 221902353  
Roll Number: 221902353  
Name: saim  
Phone: 01608908417  
Percentage: 3.0000  
  
Enter Choice: 5  
Roll Number: 221902353  
Name: saim  
Phone: 01608908417  
Percentage: 3.0000  
  
Enter Choice: _
```

Problem- 13 :

➤ C program for check a string palindrome or not

CODE :

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

int main(){
    char string1[20];
    int i, length;
    int flag = 0;

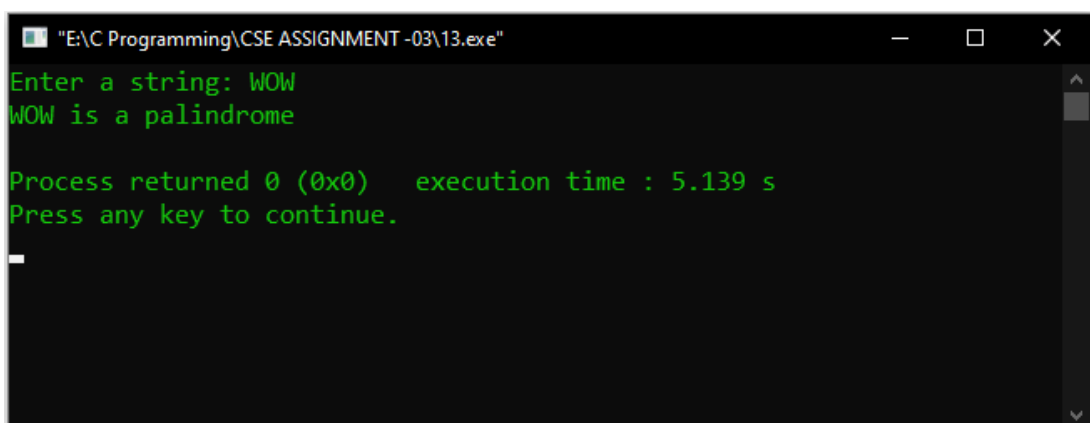
    printf("Enter a string:");
    scanf("%s", string1);

    length = strlen(string1);

    for(i=0;i < length ;i++){
        if(string1[i] != string1[length-i-1]){
            flag = 1;
            break;
        }
    }

    if (flag) {
        printf("%s is not a palindrome", string1);
    }
    else {
        printf("%s is a palindrome\n", string1);
    }
    return 0;
}
```

RESULT:

A screenshot of a Windows command prompt window titled "E:\C Programming\CSE ASSIGNMENT -03\13.exe". The window has a black background with green text. The first line shows the prompt "Enter a string:" followed by the input "WOW". The second line shows the output "WOW is a palindrome". The third line shows the status "Process returned 0 (0x0) execution time : 5.139 s". The fourth line shows the prompt "Press any key to continue." with a cursor on the next line.

```
"E:\C Programming\CSE ASSIGNMENT -03\13.exe"
Enter a string: WOW
WOW is a palindrome

Process returned 0 (0x0)   execution time : 5.139 s
Press any key to continue.
_
```

Problem- 14 :

➤ C program for concatenate string without strcat().

CODE :

```
#include <stdio.h>
int main() {
    char s1[100] = "Green ", s2[] = "University";
    int length, j;

    // store length of s1 in the length variable
    length = 0;
    while (s1[length] != '\0') {
        ++length;
    }

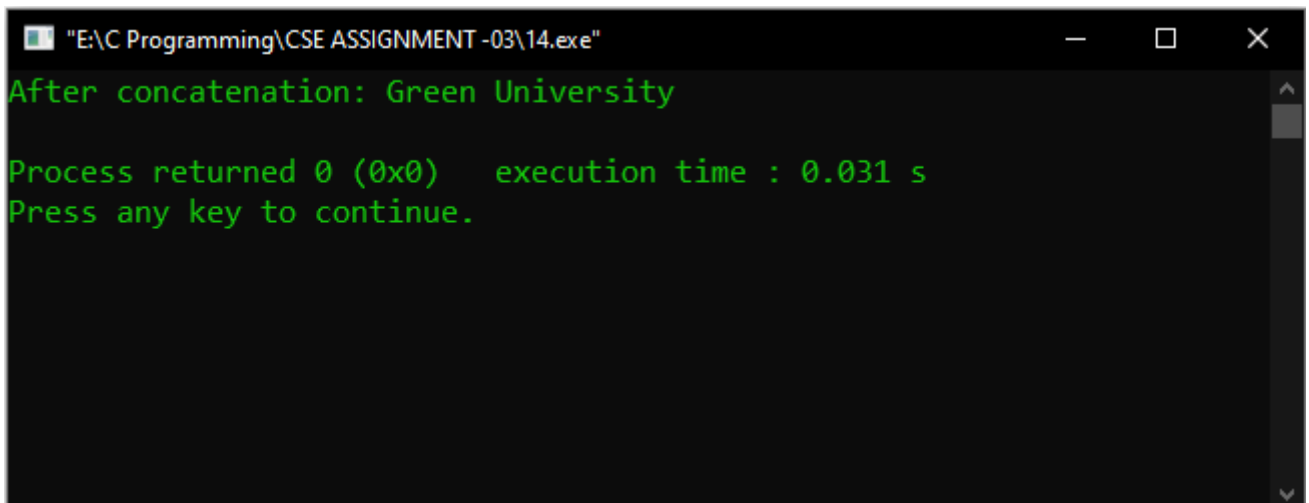
    // concatenate s2 to s1
    for (j = 0; s2[j] != '\0'; ++j, ++length) {
        s1[length] = s2[j];
    }

    s1[length] = '\0';

    printf("After concatenation: ");
    puts(s1);

    return 0;
}
```

RESULT :

A screenshot of a Windows command prompt window. The title bar at the top reads "E:\C Programming\CSE ASSIGNMENT -03\14.exe". The window has standard Windows window controls (minimize, maximize, close) on the right. The command prompt shows the output of the program: "After concatenation: Green University" in green text. Below this, it shows "Process returned 0 (0x0) execution time : 0.031 s" and "Press any key to continue." in green text. The background of the command prompt is black, and the text is green. There is a vertical scrollbar on the right side of the window.

```
"E:\C Programming\CSE ASSIGNMENT -03\14.exe"
After concatenation: Green University

Process returned 0 (0x0)   execution time : 0.031 s
Press any key to continue.
```

Problem- 15 :

➤ C program for concatenate string using strcat().

CODE :

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

int main()
{
    char a[100], b[100];

    printf("Enter the first string\n");
    gets(a);

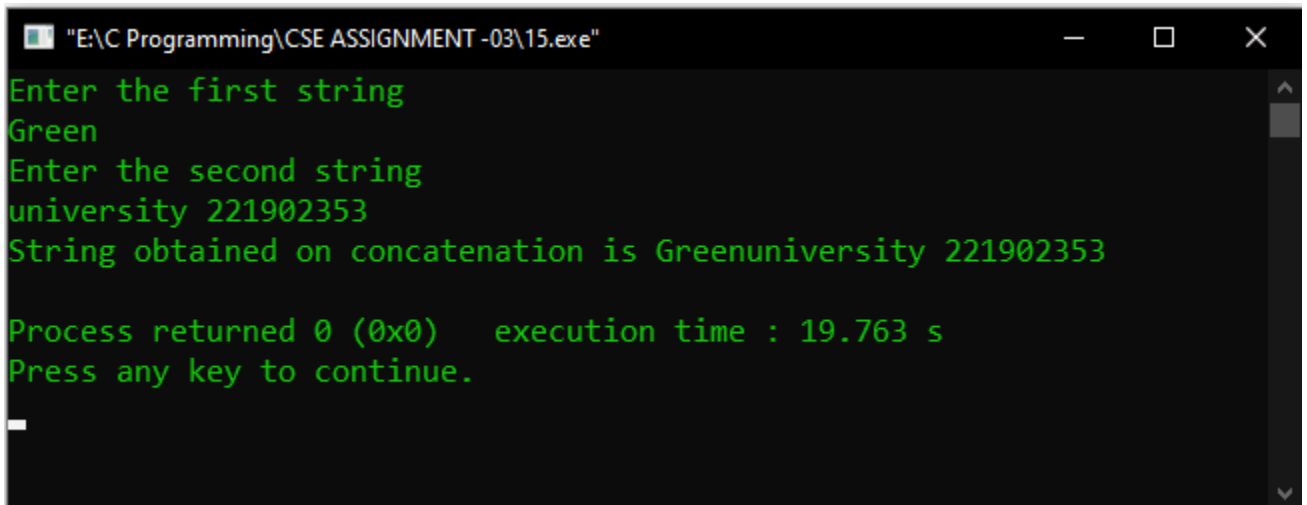
    printf("Enter the second string\n");
    gets(b);

    strcat(a,b);

    printf("String obtained on concatenation is %s\n",a);

    return 0;
}
```

RESULT :



```
"E:\C Programming\CSE ASSIGNMENT -03\15.exe"
Enter the first string
Green
Enter the second string
university 221902353
String obtained on concatenation is Greenuniversity 221902353

Process returned 0 (0x0)   execution time : 19.763 s
Press any key to continue.

```

Problem- 16 :

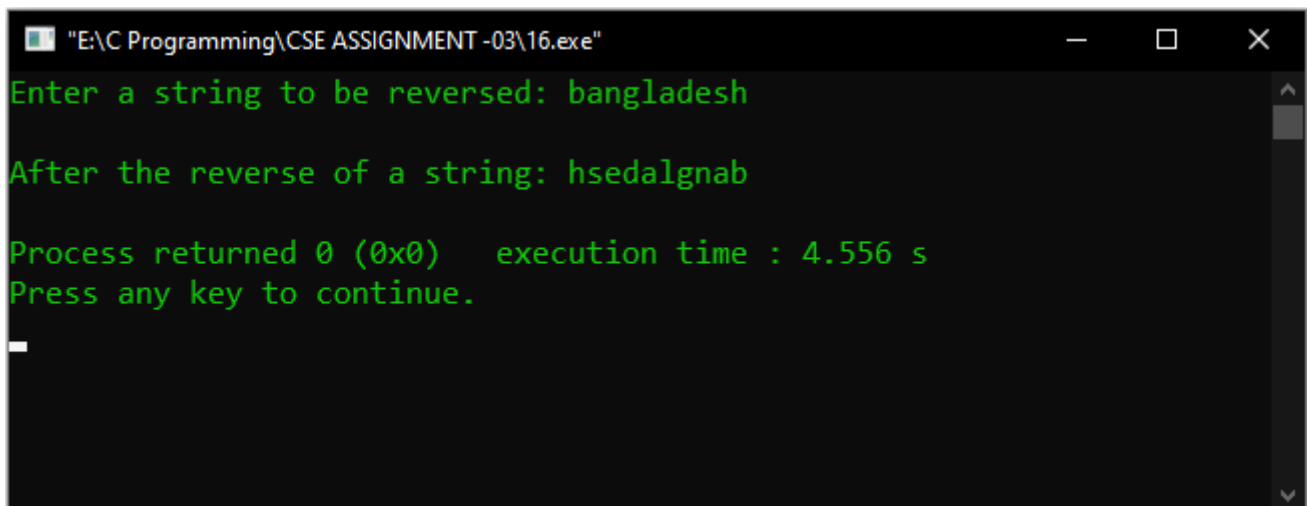
➤ C program for reverse a string

CODE :

```
#include <stdio.h>
#include <string.h>
int main()
{
    char str[40]; // declare the size of character string
    printf ("Enter a string to be reversed: ");
    scanf ("%s", str);

    // use strrev() function to reverse a string
    printf (" \nAfter the reverse of a string: %s \n", strrev(str));
    return 0;
}
```

RESULT:

A screenshot of a Windows command prompt window with a black background and green text. The title bar at the top reads "E:\C Programming\CSE ASSIGNMENT -03\16.exe". The prompt shows the program asking for a string to be reversed, with "bangladesh" entered. The output shows the reversed string "hsedalgnab". Below this, it displays "Process returned 0 (0x0) execution time : 4.556 s" and "Press any key to continue." followed by a single line of input.

```
"E:\C Programming\CSE ASSIGNMENT -03\16.exe"
Enter a string to be reversed: bangladesh

After the reverse of a string: hsedalgnab

Process returned 0 (0x0)   execution time : 4.556 s
Press any key to continue.
_
```

Problem- 17 :

➤ C program for find the Length of a String

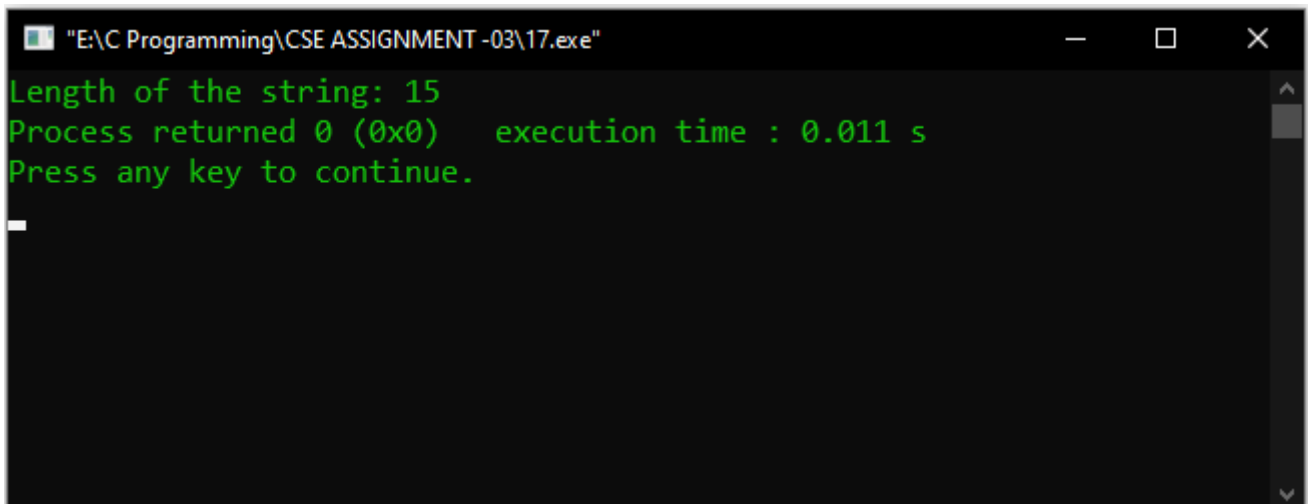
CODE :

```
#include <stdio.h>
int main() {
    char s[] = "Chittagong city";
    int i;

    for (i = 0; s[i] != '\0'; ++i);

    printf("Length of the string: %d", i);
    return 0;
}
```

RESULT :



```
"E:\C Programming\CSE ASSIGNMENT -03\17.exe"
Length of the string: 15
Process returned 0 (0x0) execution time : 0.011 s
Press any key to continue.
```

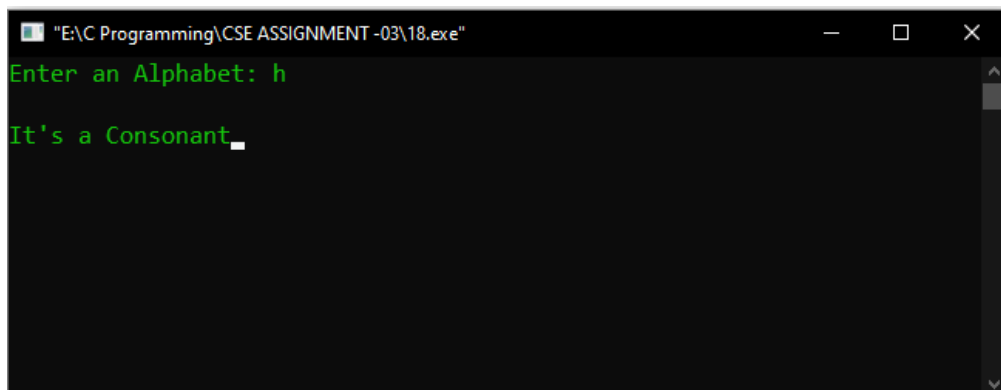
Problem- 18 :

➤ C program for check inputted character is vowel or consonant

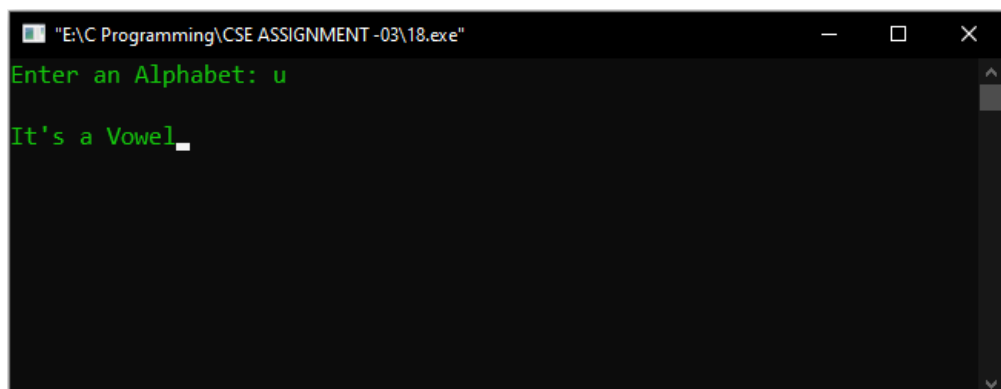
CODE :

```
#include<stdio.h>
#include<conio.h>
int main()
{
    char ch;
    printf("Enter an Alphabet: ");
    scanf("%c", &ch);
    if(ch=='a' || ch=='e' || ch=='i' || ch=='o' || ch=='u')
        printf("\nIt's a Vowel");
    else if(ch=='A' || ch=='E' || ch=='I' || ch=='O' || ch=='U')
        printf("\nIt's a Vowel");
    else
        printf("\nIt's a Consonant");
    getch();
    return 0;
}
```

RESULT :



```
"E:\C Programming\CSE ASSIGNMENT -03\18.exe"
Enter an Alphabet: h
It's a Consonant_
```



```
"E:\C Programming\CSE ASSIGNMENT -03\18.exe"
Enter an Alphabet: u
It's a Vowel_
```