Copy of
Computer Lab Assignment

NAME	ROLL NO
Dwip Shekhar Mondal	00221050303**7**
Tusher Mondal	00221050303**9**

Chosen Questions from Different Sets

Set - I:

Q1. *In a hotel, a professor lives in room no. X. The room numbers are sequentially numbered from 1 to n (n can be any integer). The sum of the room numbers left to X is equal to the sum of the room numbers to the right of X. Write a program to find X.*

Q2. *Write a program to convert an 8-digit number into words (consider both the Indian and International number system).*

Q3. *Write a program to merge two sorted lists of integers so that the resultant list remains sorted.*

Q4. *Write a program (WAP) to implement Pascal Triangle.*

Q5 . *Consider that M is a $n \times n$ square matrix whose each row contains real numbers or 0 such that the sum of each row is 1. If R is a n-dimensional column vector whose each component is 1/n. Use a random number generator to create the matrix M. Write a program to compute: $R = M$, where p should be taken as input.*

Set - II:

Q6. *Write a function named random partition() which will accept an 1-D array as input and randomly choose one of the array elements as X and partition the array into two parts where one part contains all elements less than X and another parts contain all elements greater than X. Do it without sorting the list.*

Q7. *Write a function to convert a decimal number to any other base given by the user.*

.....

Q8. *Write a program to store the country names and sort them in alphabetical order. Use an array of pointers to store the country names and pass the array to the function sort().*

Q9. *Write a recursive program to generate permutations of n numbers.*

Q10. *Write a function which will accept two strings and check whether the second string is present in the first one. If it is, it returns the starting position else returns 0. Write a program which dynamically allocates memory for two strings taken from the keyboard and uses the above function for searching one string into another.*

Set - III:

Q11. *Define a structure to store the following information of a student: name roll number, marks of five different subjects. Write a program to read information about more than 100 students, find the average marks and total marks of each of the students and sort the student names based on the total marks obtained.*

Q12. *Consider that a large binary matrix is stored in a file. Each line is a row of the matrix. The dimensions of the matrix are not known in advance. Write a program to read the matrix into a dynamic array, find its dimension, computer row-sums and create a new file to store row-no and the corresponding row sum.*

Q13. *Write a program to read a text file, convert each character to uppercase and write it to another file.*

Q14. *Write a menu driven program that depicts the working of a library. The menu options should be:*

1. *Add book information*
2. *Display book information*
3. *List all books of given author*

.....

4. *List the title of specified book*
5. *List the count of books in the library*
6. *List the books in the order of accession number*
7. *Exit*

Create a structure called 'library' to store accession number, title of the book, author name, price of the book, and flag indicating whether the book is issued or not. Consider that there are more than 1000 records.

Q15. *Write a program to add the contents of one file at the end of another file.*

Set - IV:

Q16. *Write a C++ program to create a class string, which stores string with constructor, displays the string and joins two strings with a join user defined function taking two arguments of string object.*

Q17. *Write a C++ program to demonstrate (A) Copy Constructor (B) Parameterized Constructor (C) Virtual destructor*

Q18. *Write a C++ program to overload operator '+' to concatenate two strings and hence reverse the concatenated string.*

Q19. *Write a C++ program to overload the following operators*

a) '>>' to accept time from user (in hours: mins:sec)
b) '+' to add two different user-given times.
c) '<<' to display the time in hours: mins: sec format.
d) '==' to check whether two user-given times are equal or not.

Q20. *Write a C++ program to add two complex numbers using the friend function.*

Code and Outputs

Q1. *In a hotel, a professor lives in room no. X. The room numbers are sequentially numbered from 1 to n (n can be any integer). The sum of the room numbers left to X is equal to the sum of the room numbers to the right of X. Write a program to find X.*

.....

CODE :

```
#include <stdio.h>
long myfunc(long n, long prefix, long suffix)
{
    if(n==1)
      {
            return n;
    }
    for(long i=1; i <= n; i++)
      {
            if(prefix==suffix)
            {
                  return i;
            }
        prefix += i;
            suffix -= (i + 1);

      }
    return -1;
}
long calcsum(long low, long up)
{
      long sum = 0;
      for(long i=low; i <= up; i++)
      {
            sum = sum + i;
      }
      return sum;
}
int main()
```

.....

```
{
      long n;
      long left = 0;
      printf("Enter total rooms:\n");
      scanf("%ld", &n);
      long right = calcsum(2, n);
    long hotel = myfunc(n, left, right);
    if(hotel == -1)
    {
        printf("Does not exist\n");
    }
    else
    {
        printf("Hotel room no: %ld\n", hotel);
    }
      return 0;
}
```

OUTPUT :

```
C:\MCA-Codes\C\A_39>q
Enter total rooms:
57121
Hotel room no: 40391


C:\MCA-Codes\C\A_39>q
Enter total rooms:
1681
Hotel room no: 1189


C:\MCA-Codes\C\A_39>q
Enter total rooms:
289
Does not exist
```

Q2. *Write a program to convert an 8-digit number into words (consider both the Indian and International number system).*

.....

CODE:

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
char *ones[20] = {"", "one ", "two ", "three ", "four ",
                  "five ","six ", "seven ","eight ",    "nine ",
                  "ten ",     "eleven ",  "twelve ",    "thirteen ",
"fourteen ",
                  "fifteen ", "sixteen ", "seventeen ", "eighteen ",
"nineteen " };


char *tens[10] = { "", "", "twenty ",  "thirty ", "forty ", "fifty ", "sixty ",
"seventy ", "eighty ", "ninety "};

char* num_to_word_two_digit(int);
char* num_to_word_three_digit(int);
void print_indian(int);
void print_international(int);
int main(){
    long n;
    char res[100];

    printf("\nEnter any number of 8 digits or less: \n");
    scanf("%ld", &n);

    print_indian(n);
    print_international(n);

    return 0;
}

char* num_to_word_two_digit(int n){
    char *str = (char*)malloc(sizeof(char)*100);
    char word1[100], word2[100];
    int q, r;
    if (n < 20){
```

.....

```
        return ones[n];
    }
    q = n / 10;
    r = n % 10;

    strcpy(word1, tens[q]);
    strcpy(word2, ones[r]);
    strcat(word1, word2);
    strcpy(str, word1);

    return str;
}

char* num_to_word_three_digit(int n){
    char *str = (char*)malloc(sizeof(char)*100);

    char word1[100], word2[100];
    int q, r;
    if (n < 100){
        return num_to_word_two_digit(n);
    }
    q = n / 100;
    r = n % 100;

    strcpy(word1, ones[q]);
    strcat(word1, "hundred ");
    strcpy(word2, num_to_word_two_digit(r));
    strcat(word1, word2);
    strcpy(str, word1);
    // printf("\nNumber in words: %s\n", word1);
    // strcat(res, word2);
    return str;
}

void print_indian(int n){
    char *result = (char*)malloc(sizeof(char)*255);
    strcpy(result, num_to_word_two_digit(n/10000000));
    if (num_to_word_two_digit(n/10000000) != ""){
        strcat(result, "crore ");
    }
    n = n % 10000000;
    //print lakhs
    strcat(result, num_to_word_two_digit(n/100000));
```

```
    if (num_to_word_two_digit(n/100000) != ""){
        strcat(result, "lakh ");
    }
    n = n % 100000;
    //print thousands
    strcat(result, num_to_word_two_digit(n/1000));
    if (num_to_word_two_digit(n/1000) != ""){
        strcat(result, "thousand ");
    }
    n = n % 1000;
    //print hundreds
    strcat(result, num_to_word_two_digit(n/100));
    if (num_to_word_two_digit(n/100) != ""){
        strcat(result, "hundred ");
    }
    n = n % 100;
    //print remainder
    strcat(result, num_to_word_two_digit(n));
    printf("\nNumber in words indian: %s\n", result);
}

void print_international(int n){
    char *result = (char*)malloc(sizeof(char)*255);
    //print million
    strcpy(result, num_to_word_two_digit(n/1000000));
    if (num_to_word_two_digit(n/10000000) != ""){
        strcat(result, "million ");
    }
    n = n % 1000000;
    //print thousands
    strcat(result, num_to_word_three_digit(n/1000));
    if (num_to_word_three_digit(n/1000) != ""){
        strcat(result, "thousand ");
    }
    n = n % 1000;
    //print remainder
    strcat(result, num_to_word_three_digit(n));
    printf("\nNumber in words international: %s\n", result);

}
```

OUTPUT :

```
C:\MCA-Codes\C\A_39>q

Enter any number of 8 digits or less:
29569995

Number in words indian: two crore ninety five lakh
sixty nine thousand nine hundred ninety five

Number in words international: twenty nine million
five hundred sixty nine thousand nine hundred ninet
y five


C:\MCA-Codes\C\A_39>q

Enter any number of 8 digits or less:
42609

Number in words indian: forty two thousand six hundred
 nine

Number in words international: forty two thousand six
hundred nine


C:\MCA-Codes\C\A_39>q

Enter any number of 8 digits or less:
10

Number in words indian: ten

Number in words international: ten
```

Q3. *Write a program to merge two sorted lists of integers so that the resultant list remains sorted.*

CODE:

```
#include <stdio.h>
#include <math.h>
int main()
{
	int size_a, size_b;
	printf("Enter the size of the first sorted list :\n");
	scanf("%d", &size_a);
	int a[size_a];
	printf("\nEnter the elements :\n");
	for(int i=0; i<size_a; i++)
	{
		printf("Number %d : ", i+1);
		scanf("%d", &a[i]);
	}
	printf("Printing the first sorted list.\n");
	for(int i=0; i<size_a; i++)
	{
		printf("%d\t", a[i]);
	}
	printf("\n\nEnter the size of the second sorted list :\n");
	scanf("%d", &size_b);
	int b[size_b];
	printf("\nEnter the elements :\n");
	for(int i=0; i<size_b; i++)
	{
		printf("Number %d : ", i+1);
		scanf("%d", &b[i]);
	}
	printf("\nPrinting the second sorted list.\n");
	for(int i=0; i<size_b; i++)
	{
		printf("%d\t", b[i]);
	}
	printf("\n");
	int size_c;
	size_c = size_a+size_b;
	int c[size_c];
```

```
	int point_a = 0, point_b = 0, point_c=0;
	while(point_c < size_c)
	{
		if(point_a < size_a && point_b < size_b)
		{
			if(a[point_a] > b[point_b])
			{
				c[point_c] = b[point_b];
				point_b = point_b + 1;
			}
			else
			{
				c[point_c] = a[point_a];
				point_a = point_a + 1;
			}
		}
		else if(point_b < size_b)
		{
			c[point_c] = b[point_b];
			point_b = point_b + 1;
		}
		else if(point_a < size_a)
		{
			c[point_c] = a[point_a];
			point_a = point_a + 1;
		}

		point_c = point_c + 1;
	}
	printf("\nThe resultant sorted list is : \n");
	for(int i=0; i<size_c; i++)
	{
		printf("%d\t", c[i]);
	}
	printf("\n");
	return 0;
}
```

OUTPUT:

```
C:\MCA-Codes\C\A_39>q
Enter the size of the first sorted list :
4

Enter the elements :
Number 1 : 20
Number 2 : 40
Number 3 : 60
Number 4 : 80
Printing the first sorted list.
20      40      60      80

Enter the size of the second sorted list :
4

Enter the elements :
Number 1 : 10
Number 2 : 30
Number 3 : 50
Number 4 : 70

Printing the second sorted list.
10      30      50      70

The resultant sorted list is :
10      20      30      40      50      60      70      80

C:\MCA-Codes\C\A_39>q
Enter the size of the first sorted list :
3

Enter the elements :
Number 1 : 5
Number 2 : 10
Number 3 : 12
Printing the first sorted list.
5       10      12

Enter the size of the second sorted list :
2

Enter the elements :
Number 1 : 1
Number 2 : 13

Printing the second sorted list.
1       13

The resultant sorted list is :
1       5       10      12      13
```

Q4. *Write a program (WAP) to implement Pascal Triangle.*

CODE:

```c
#include <stdio.h>

typedef long long int ll;

ll binomialCoeff(int, int);
ll fact(int);

int main(){
    int n, i, j, k;
    ll ele;
    printf("\nEnter the number of rows to be printed:\n");
    scanf("%d", &n);

    for(i = 0; i < n; i++){
        //print spaces
        for(j = n - i; j > 0; j--){
            printf(" ");
        }
        // print i elements for ith line
        for (k = 0; k <= i; k++)
        {
            //choose from line i, k i.e ncr(i, k)
            ele = binomialCoeff(i , k);
            printf("%3lld", ele);
        }
        printf("\n");
    }

    return 0;
}


ll fact(int n)
{
    if (n == 0){
        return 1;
    }
    return n*fact(n - 1);
```

```
}

ll binomialCoeff(int n, int k)
{
    return fact(n) / (fact(n - k) * fact(k));
}
```

OUTPUT:

```
Microsoft Windows [Version 10.0.19044.2486]
(c) Microsoft Corporation. All rights reserved.

C:\MCA-Codes>.\pascal

Enter the number of rows to be printed:
6
        1
       1  1
      1  2  1
     1  3  3  1
    1  4  6  4  1
   1  5 10 10  5  1

C:\MCA-Codes>
```

```
C:\MCA-Codes>.\pascal

Enter the number of rows to be printed:
7
         1
        1  1
       1  2  1
      1  3  3  1
     1  4  6  4  1
    1  5 10 10  5  1
   1  6 15 20 15  6  1

C:\MCA-Codes>
```

Q5 . *Consider that M is a $n \times n$ square matrix whose each row contains real numbers or 0 such that the sum of each row is 1. If R is a n-dimensional column vector whose each component is 1/n. Use a random number generator to create the matrix M. Write a program to compute: $R = M$, where p should be taken as input.*

CODE:

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define m 100
int main()
{
    srand(time(NULL));
    int n, p, i, j, k, rem, r;
    float M[m][m], M_p[m][m], R[m][1], new_R[m][1];
    printf("Enter the dimension of square matrix : \n");
    scanf("%d", &n);
    for (i = 0; i < n; i++)
    {
        rem = 100;
        for (j = 0; j < n - 1; j++)
        {
            r = rand() % rem;
            M[i][j] = r * 0.01;
            rem -= r;
        }
        M[i][n - 1] = rem * 0.01;
    }

    printf("\nThe generated matrix M is : \n");
    for (i = 0; i < n; i++)
    {
        for (j = 0; j < n; j++)
        {
            printf("%.2f ", M[i][j]);
        }
        printf("\n");
    }
    for (i = 0; i < n; i++)
```

```
    {
        R[i][0] = 1.0 / n;
    }
    printf("\nThe calculated matrix R is : \n");
    for (i = 0; i < n; i++)
    {
        printf("%.2f\n", R[i][0]);
    }
    printf("\nEnter the value of p : ");
    scanf("%d", &p);
    while (p--)
    {
        for (i = 0; i < n; i++)
        {
            float temp = 0;
            for (j = 0; j < n; j++)
            {
                temp += (M[i][j] * R[j][0]);
            }
            new_R[i][0] = temp;
        }
        for (i = 0; i < n; i++)
        {
            R[i][0] = new_R[i][0];
        }
    }
    printf("\nThe matrix (M^p)*R is : \n");
    for (i = 0; i < n; i++)
    {
        printf("%.2f\n", R[i][0]);
    }
    printf("Hence proved that for any positive integer P, the statement R
=(M^P)*R holds.\n");
    return 0;
}
```

OUTPUT:

```
Enter the dimension of square matrix :
3

The generated matrix M is :
0.10 0.84 0.06
0.23 0.33 0.44
0.09 0.09 0.82

The calculated matrix R is :
0.33
0.33
0.33

Enter the value of p : 10

The matrix (M^p)*R is :
0.33
0.33
0.33
Hence proved that for any positive integer P, the statement R =(M^P)*R holds.

Enter the dimension of square matrix :
5

The generated matrix M is :
0.93 0.00 0.00 0.03 0.04
0.39 0.47 0.00 0.00 0.14
0.23 0.55 0.01 0.02 0.19
0.49 0.20 0.19 0.11 0.01
0.61 0.29 0.06 0.02 0.02

The calculated matrix R is :
0.20
0.20
0.20
0.20
0.20

Enter the value of p : 100

The matrix (M^p)*R is :
0.20
0.20
0.20
0.20
0.20
Hence proved that for any positive integer P, the statement R =(M^P)*R holds.
```

Q6. *Write a function named random partition() which will accept an 1-D array as input and randomly choose one of the array elements as X and partition the array into two parts where one part contains all elements less than X and another parts contain all elements greater than X. Do it without sorting the list.*

CODE:

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
typedef long int big_int;

void partition(big_int*, int);
void display_arr(big_int*, int);
big_int swap(big_int*, big_int*);

int main(){
    srand(time(NULL));
    big_int* arr;
    int n, i;
    printf("\nEnter the size of the array: \n");
    scanf("%d", &n);
    arr = (big_int*)calloc(n, sizeof(big_int));

    printf("\nEnter the elements in the array\n");
    for (i = 0; i < n; i++){
        scanf("%ld", &arr[i]);
    }

    display_arr(arr, n);
    partition(arr, n);

    return 0;
}
void display_arr(big_int* arr, int n){
    printf("\n");
    int i = 0;
    for (i = 0; i < n; i++){
        printf("%ld  ", arr[i]);
    }

}
```

```
void partition(big_int *arr, int n){
    int rand_index = 0, size1, size2, i, counter = 0;
    big_int pivot;
    big_int *p1, *p2;
    rand_index = (rand() % ( (n - 1) - 0 + 1)) + 0;
    printf("\nRandom index: %d, Pivot element: %ld\n", rand_index,
arr[rand_index]);

    int low = 0, high = n - 1;
    swap(&arr[rand_index], &arr[0]);
    pivot = arr[0];

    while (low < high){
        while (arr[low] <= pivot){
            low++;
        }
        while (arr[high] > pivot){
            high--;
        }
        if (low < high){
            swap(&arr[low], &arr[high]);
        }
    }
    swap(&arr[0], &arr[high]);
    printf("\nPatition index: %d, element: %d\n", high, arr[high]);

    size1 = high;
    size2 = n - high - 1;

    p1 = (big_int*)calloc(n, sizeof(big_int));
    p2 = (big_int*)calloc(n, sizeof(big_int));

    printf("\nArray after partitioning: \n");
    display_arr(arr, n);
    printf("\nSize of the new parts: %d, %d\n", size1, size2);

    for (i = 0; i < high; i++){
        p1[i] = arr[i];
    }
    for (i = high + 1; i < n; i++){
        p2[counter++] = arr[i];
    }
    printf("\nElements in partition 1\n");
```

```
    if (size1 != 0){
        display_arr(p1, size1);
    }else{
        printf("\nNone\n");
    }
    printf("\nElements in partition 2\n");
    if(size2 != 0){
        display_arr(p2, size2);
    }else{
        printf("\nNone\n");
    }
}
big_int swap(big_int *a, big_int *b){
    big_int tmp = *a;
    *a = *b;
    *b = tmp;
}
```

OUTPUT:

```
C:\MCA-Codes>.\partition

Enter the size of the array:
6

Enter the elements in the array
4 2 1 0 6 8

4  2  1  0  6  8
Random index: 4, Pivot element: 6

Patition index: 4, element: 6

Array after partitioning:

4  2  1  0  6  8
Size of the new parts: 4, 1

Elements in partition 1

4  2  1  0
Elements in partition 2

8
C:\MCA-Codes>
```

Q7. *Write a function to convert a decimal number to any other base given by the user.*

CODE:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
typedef unsigned long long int big_int;

int dec_to_base_x(big_int, int, char*, int);

int main(){

    big_int a;
    int base, index;
    char res[1024];

    printf("\nEnter Number: \n");
    scanf("%lld", &a);

    printf("\nEnter base to convert: \n");
    scanf("%d", &base);

    if(a == 0){
        printf("\nThe number in converted base: %d \n", 0);
        exit(1);
    }

    index = dec_to_base_x(a, base, res, 0);
    res[index] = '\0';
    strrev(res);
    printf("\nThe number in converted base : %s \n", res);

    return 0;
}

int dec_to_base_x(big_int a, int base, char* res, int index){
    if (a == 0){
       //strcpy(res, "0");
       if(a > 9){
            //printf("%c", (55 + a));
```

```
            res[index] = (55 + a);
        }else{
            //printf("%c", (48 + a));
            res[index] = (48 + a);
        }
        return index;
    }else{
        int x;
        x = dec_to_base_x(a / base, base, res, index + 1);
        if(a % base > 9){
            //printf("%c", (55 + a % base));
            res[index] = (55 + a % base);
        }else{
            //printf("%c", (48 + a % base));
            res[index] = (48 + a % base);
        }
        return x;
    }
}
```

OUTPUT:

```
C:\MCA-Codes>.\d2any

Enter Number:
144

Enter base to convert:
16

The number in converted base : 90

C:\MCA-Codes>.\d2any

Enter Number:
144

Enter base to convert:
2

The number in converted base : 10010000

C:\MCA-Codes>
```

Q8. *Write a program to store the country names and sort them in alphabetical order. Use an array of pointers to store the country names and pass the array to the function sort().*

CODE:

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

void sort_str_lex(char**, int);
void display(char**, int);

int main(){
    int n, i;
    char *country[100], c;
    printf("\nEnter number of strings to input\n");
    scanf("%d", &n);
    scanf("%c", &c);
    printf("\nEnter strings --\n");
    for (i = 0; i < n; i++){
        country[i] = (char*)malloc(30 * sizeof(char));
        gets(country[i]);
    }
    sort_str_lex(country, n);
    printf("\n-- After sorting --\n");
    display(country, n);
}

void display(char** arr, int n){
    printf("\nThe strings are --\n");
    int i;
    for (i = 0; i < n; i++){
        puts(arr[i]);
    }
}

void sort_str_lex(char** country, int n){
    int i, j, swapped;
    char tmp[60];
    for (i = 0; i < n - 1; i++){
        swapped = 0;
        for (j = 0; j < n - i - 1; j++){
```

```
            if (strcmp(country[j], country[j + 1]) > 0){
                strcpy(tmp, country[j]);
                strcpy(country[j], country[j + 1]);
                strcpy(country[j + 1], tmp);
                swapped = 1;
            }
        }
        if(swapped == 0){
            break;
        }
    }
}
```

OUTPUT:

```
C:\MCA-Codes>.\sort_lex

Enter number of strings to input
5

Enter strings --
india
japan
indonesia
jamaica
usa

-- After sorting --

The strings are --
india
indonesia
jamaica
japan
usa

C:\MCA-Codes>
```

Q9. *Write a recursive program to generate permutations of n numbers.*

CODE:

```c
#include <stdio.h>
#include <stdlib.h>
void odolbodol(int*, int*);
void recper(int*, int, int);
void takeinp(int*, int);
int main()
{
  int n;
  printf("Enter number of elements : ");
  scanf("%d", &n);
  int arr[n];
  takeinp(&arr[0], n);
  recper(arr, 0, n - 1);
  return 0;
}
void takeinp(int* p, int n)
{
	int pointer = 0;
	while(pointer<n)
	{
		printf("Enter element %d: ", pointer+1);
		scanf("%d", p+pointer);
		pointer = pointer + 1;
	}
}
void odolbodol(int *a, int *b)
{
  int temp = *a;
  *a = *b;
  *b = temp;
}
void recper(int *arr, int start, int end)
{
  if (start == end)
  {
    for (int i = 0; i <= end; i++)
      {
      printf("%d ", arr[i]);
```

```
    }
    printf("\n");
  }
  else
  {
    for (int i = start; i <= end; i++)
      {
      odolbodol(&arr[start], &arr[i]);
      recper(arr, start + 1, end);
      odolbodol(&arr[start], &arr[i]);
    }
  }
}
```

OUTPUT:

```
C:\MCA-Codes\C\B_39>q
Enter number of elements : 3
Enter element 1: 1
Enter element 2: 3
Enter element 3: 5
1 3 5
1 5 3
3 1 5
3 5 1
5 3 1
5 1 3
C:\MCA-Codes\C\B_39>q
Enter number of elements : 2
Enter element 1: 1
Enter element 2: 0
1 0
0 1
```

Q10. *Write a function which will accept two strings and check whether the second string is present in the first one. If it is, it returns the starting position else returns 0. Write a program which dynamically allocates memory for two strings taken from the keyboard and uses the above function for searching one string into another.*

CODE:

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

int check_sub_str(char*, char*);


int main(){
    char *str1, *str2, c;
    int m, n, res;

    printf("\nEnter size of the first string: \n");
    scanf("%d", &m);
    printf("\nEnter size of the second string: \n");
    scanf("%d", &n);
    scanf("%c", &c);
    str1 = (char*)malloc(m * sizeof(char));
    str2 = (char*)malloc(n * sizeof(char));

    printf("\nEnter the first string: \n");
    gets(str1);
    printf("\nEnter the second string: \n");
    gets(str2);


    if (*str1 == '\0' || *str1 == '\n'){
        printf("source string is not given");
        exit(1);
    }

    if (*str2 == '\0' || *str2 == '\n'){
        printf("Substring is not given to check");
        exit(1);
```

.....

```
    }


    res = check_sub_str(str1, str2);

    if (res == 0){
        printf("\nSubstring is not present.\n");
    }else{
        printf("\nSubstring present starting at index %d\n", res);
    }

    return 0;
}

int check_sub_str(char *str1, char *str2){

    int size_1 = 0, size_2 = 0, i, j, flag;

    while (str1[size_1] != '\0'){
        size_1++;
    }
    while (str2[size_2] != '\0'){
        size_2++;
    }

    for (i = 0; i <= size_1 - size_2; i++){
        for (j = i; j < i + size_2; j++){
            flag = 1;
            if (str1[j] != str2[j - i]){
                flag = 0;
                break;
            }
        }
        if (flag == 1){
            return i;
        }
    }

    return 0;
}
```

OUTPUT:

```
C:\MCA-Codes>.\sub

Enter size of the first string:
50

Enter size of the second string:
50

Enter the first string:
Dwip Mondal

Enter the second string:
Mondal

Substring present starting at index 5

C:\MCA-Codes>
```

```
C:\MCA-Codes>.\sub

Enter size of the first string:
50

Enter size of the second string:
50

Enter the first string:
This is sample string

Enter the second string:
random string

Substring is not present.

C:\MCA-Codes>
```

Q11. *Define a structure to store the following information of a student: name roll number, marks of five different subjects. Write a program to read information about more than 100 students, find the average marks and total marks of each of the students and sort the student names based on the total marks obtained.*

CODE:

```
#include <stdio.h>
#include <stdlib.h>
#define maxn 50
#define maxs 5

typedef struct
{
    char fname[maxn], lname[maxn];
    int Roll_Number;
    float total, Sub[maxs], avg;
} Student;

void displayOrder(Student s[], int *roll, int n);
void odolbodol(float* total, int *roll, int i, int j);
void driverInp(Student s[], int *roll, float *total, int n);
void driverPrint(int *roll, float *total, int n);
void driverDisplay(Student s[], int n);
void takeinp(Student *s);
void disp(Student s);
void driverSort(int *roll, float *total, int n);

int main()
{
    int n, *roll;
    float *total;
    printf("Number of Students : ");
    scanf("%d", &n);
    roll = (int *)malloc(sizeof(int)*n);
    total = (float *)malloc(sizeof(float)*n);
    Student s[n];
    printf("Taking input of %d students one by one!\n", n);
    for(int i=0; i<n; i++)
    {
```

```
        printf("\nReceiving data of Student Number %d", i+1);
        takeinp(&s[i]);
    }
    driverInp(s, roll, total, n);
    printf("\nDisplaying output of %d students one by one!\n", n);
    driverDisplay(s, n);
    driverSort(roll, total, n);
    printf("\nDisplaying Students names considering their total marks in
descending order!\n");
    displayOrder(s, roll, n);
    return 0;
}

void displayOrder(Student s[], int *roll, int n)
{
    for(int i=0; i<n; i++)
    {
        for(int j=0; j<n; j++)
        {
            if(s[j].Roll_Number == *(roll+i))
            {
                printf("%s %s\t", s[j].fname, s[j].lname);
                break;
            }
        }
    }
    printf("\n");
}

void driverSort(int *roll, float *total, int n)
{
    for(int i=0; i<n; i++)
    {
        for(int j=i+1; j<n; j++)
        {
            if(*(total+i)<*(total+j))
            {
                odolbodol(total, roll, i, j);
            }
        }
    }
}
```

```
void odolbodol(float* total, int *roll, int i, int j)
{
    int temp1 = *(roll + i);
    *(roll + i) = *(roll + j);
    *(roll + j) = temp1;
    float temp2 = *(total + i);
    *(total + i) = *(total + j);
    *(total + j) = temp2;
}

void driverInp(Student s[], int *roll, float *total, int n)
{
    for(int i=0; i<n; i++)
    {
        *(roll+i) = s[i].Roll_Number;
        *(total+i) = s[i].total;
    }
}

void driverPrint(int *roll, float *total, int n)
{
    printf("\nRoll Numbers - ");
    for(int i=0; i<n; i++)
    {
        printf("%d ", *(roll + i));
    }
    printf("\nTotal Numbers - ");
    for(int i=0; i<n; i++)
    {
        printf("%f ", *(total + i));
    }
}

void driverDisplay(Student s[], int n)
{
    for(int i=0; i<n; i++)
    {
        printf("\nStudent Number %d - ", i+1);
        disp(s[i]);
    }
}

void disp(Student s)
```

```
{
    printf("\nFull Name : %s %s", s.fname, s.lname);
    printf("\nRoll : %d\n", s.Roll_Number);
    printf("Average marks : %f\n", s.avg);
    printf("Total marks : %f\n", s.total);
}

void takeinp(Student *s)
{
    s->total = 0;
    printf("\nEnter full name : ");
    scanf("%s %s", s->fname, s->lname);
    printf("Enter Roll Number : ");
    scanf("%d", &s->Roll_Number);
    for(int i=0; i<5; i++)
    {
        printf("Enter marks %i : ", i+1);
        scanf("%f", &s->Sub[i]);
        s->total = s->total + s->Sub[i];
    }
    s->avg = s->total/5.0;
}
```

OUTPUT:

```
C:\MCA-Codes\C\C_39>q
Number of Students : 3
Taking input of 3 students one by one!

Receiving data of Student Number 1
Enter full name : Sudip Ghosh
Enter Roll Number : 19
Enter marks 1 : 87
Enter marks 2 : 100
Enter marks 3 : 94
Enter marks 4 : 98
Enter marks 5 : 99

Receiving data of Student Number 2
Enter full name : Dwip Mondal
Enter Roll Number : 37
Enter marks 1 : 100
Enter marks 2 : 100
Enter marks 3 : 100
Enter marks 4 : 98
Enter marks 5 : 98

Receiving data of Student Number 3
Enter full name : Tusher Mondal
Enter Roll Number : 39
Enter marks 1 : 100
Enter marks 2 : 100
Enter marks 3 : 100
Enter marks 4 : 100
Enter marks 5 : 96

Displaying output of 3 students one by one!

Student Number 1 -
Full Name : Sudip Ghosh
Roll : 19
Average marks : 95.599998
Total marks : 478.000000

Student Number 2 -
Full Name : Dwip Mondal
Roll : 37
Average marks : 99.199997
Total marks : 496.000000

Student Number 3 -
Full Name : Tusher Mondal
Roll : 39
Average marks : 99.199997
Total marks : 496.000000

Displaying Students names considering their total marks in descending order!
Dwip Mondal     Tusher Mondal   Sudip Ghosh
```

Q12. *Consider that a large binary matrix is stored in a file. Each line is a row of the matrix. The dimensions of the matrix are not known in advance. Write a program to read the matrix into a dynamic array, find its dimension, computer row-sums and create a new file to store row-no and the corresponding row sum.*

CODE:

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

#define TRUE 1
#define FALSE 0
#define MAX_BUFF_SIZE 1024


int check_valid(FILE*, int*);
int count_element(char*);
void display_mat(int**, int, int);
void push_into_matrix(FILE*, int**, int, int);
void row_sum(FILE*, int**, int, int);

int main(){

    FILE* file = fopen("./matrix.txt", "r");
    FILE *file_ptr_2;

    int is_valid_matrix, i;
    int dims[2] = {-1, -1};
    int **bin_matrix;

    is_valid_matrix = check_valid(file, dims);

    if(!is_valid_matrix){
        printf("\nThe input matrix has invalid dimensions: ----\n");
        exit(1);
    }

    printf("\nMatrix dimensions are: %d x %d\n", dims[0], dims[1]);


    bin_matrix = (int**)malloc(sizeof(int*) * dims[0]);
```

```
    for (i = 0; i < dims[0]; i++){
        bin_matrix[i] = (int*)malloc(sizeof(int) * dims[1]);
    }

    file = fopen("./matrix.txt", "r");
    push_into_matrix(file, bin_matrix, dims[0], dims[1]);
    display_mat(bin_matrix, dims[0], dims[1]);

    file_ptr_2 = fopen("./matrix_row_sum.txt", "w");
    row_sum(file_ptr_2, bin_matrix, dims[0], dims[1]);



    return 0;
}

void row_sum(FILE *file_ptr, int **matrix, int rows, int cols){
    if (file_ptr == NULL){
        printf("\nError opening file\n");
        exit(1);
    }
    char ch, tmp[64], line[256], row_no[10];
    int i = 0, j = 0, sum = 0;

    for (i = 0; i < rows; i++){
        sum = 0;
        for (j = 0; j < cols; j++){
            sum += matrix[i][j];
        }
        sprintf(tmp, "%d", sum);
        sprintf(row_no, "%d", i + 1);
        strcpy(line, "Sum of Row ");
        strcat(line, row_no);
        strcat(line, ": ");
        strcat(line, tmp);
        strcat(line, "\n");
        printf("%s\n", line);
        fputs(line, file_ptr);
    }

    fclose(file_ptr);
}
```

```c
int check_valid(FILE* file_ptr, int *dims){

    if (file_ptr == NULL){
        return FALSE;
    }
    int ch_index = 0, curr_row_ele, line_no = 0;
    int row_elements;
    char ch, line[MAX_BUFF_SIZE];

    while (ch = fgetc(file_ptr)){
        if (ch_index >= MAX_BUFF_SIZE){
            printf("Error max line size reached!");
            return FALSE;
        }
        if (ch == EOF){
            line_no += 1;
            line[ch_index] = '\0';
            fprintf(stdout, "%s", line);
            curr_row_ele = count_element(line);
            printf("\trow items:  %d\n", curr_row_ele);

            if (row_elements != curr_row_ele){
                // printf("row legth mismatch!:  %s", line);
                return FALSE;
            }
            break;
        }else if (ch == '\n') {
            line_no += 1;
            line[ch_index] = '\0';
            ch_index = 0;
            fprintf(stdout, "%s", line);
            curr_row_ele = count_element(line);
            printf("\trow items:  %d\n", curr_row_ele);
            if (line_no == 1){
                row_elements = curr_row_ele;
            }
            if (row_elements != curr_row_ele){
                // printf("row legth mismatch!:  %s", line);
                return FALSE;
            }
            continue;
        }else{
```

```
            line[ch_index++] = ch;
        }
    }


    dims[0] = line_no;
    dims[1] = row_elements;
    fclose(file_ptr);
    return TRUE;

}

void push_into_matrix(FILE *file_ptr, int** matrix, int rows, int cols){
    if (file_ptr == NULL){
        printf("\nError opening file\n");
        exit(1);
    }
    char ch;
    int i = 0, j = 0;
    while ((ch = fgetc(file_ptr)) != EOF){
        printf("%c", ch);
        if (ch != '0' && ch != '1'){
            continue;
        }

        if (j == cols){
            i += 1;
            j = 0;
        }
        matrix[i][j] = ch - '0';
        j ++;
    }

    fclose(file_ptr);
}


void display_mat(int** matrix, int m, int n){
    int i , j;
    printf("\nElements in the matrix are ---->\n");
    for (i = 0; i < m; i++)
    {
        for (j = 0; j < n; j++)
```

```
        {
            printf("%d\t", matrix[i][j]);
        }
        printf("\n");
    }
}

int count_element(char *line){
    char *temp = line;
    int ele = 0;
    while (*temp != '\0'){
        if (*temp == '0' || *temp == '1'){
            ele ++;
        }
        temp++;
    }
    return ele;
}
```

OUTPUT:

```
C:\MCA-Codes>gcc matrix.c -o matrix_file

C:\MCA-Codes>.\matrix_file
1 0 1 1 0 1 1 1 1 0 1 1 0       row items:  13
0 0 0 1 1 0 0 0 0 1 1 0 1       row items:  13
0 0 0 0 1 0 0 1 1 0 1 1 1       row items:  13
1 1 1 1 0 0 1 1 0 1 1 1 0       row items:  13
1 1 1 0 0 1 0 0 0 0 1 1 1       row items:  13

Matrix dimentions are: 5 x 13
1 0 1 1 0 1 1 1 1 0 1 1 0
0 0 0 1 1 0 0 0 0 1 1 0 1
0 0 0 0 1 0 0 1 1 0 1 1 1
1 1 1 1 0 0 1 1 0 1 1 1 0
1 1 1 0 0 1 0 0 0 0 1 1 1
Elements in the matrix are ---->
1       0       1       1       0       1       1       1       1       0       1       1       0
0       0       0       1       1       0       0       0       0       1       1       0       1
0       0       0       0       1       0       0       1       1       0       1       1       1
1       1       1       1       0       0       1       1       0       1       1       1       0
1       1       1       0       0       1       0       0       0       0       1       1       1
Sum of Row 1: 9

Sum of Row 2: 5

Sum of Row 3: 6

Sum of Row 4: 9

Sum of Row 5: 7
```

matrix_row_sum.txt matrix.txt

matrix.txt

```
1 0 1 1 0 1 1 1 1 0 1 1 0
0 0 0 1 1 0 0 0 0 1 1 0 1
0 0 0 0 1 0 0 1 1 0 1 1 1
1 1 1 1 0 0 1 1 0 1 1 1 0
1 1 1 0 0 1 0 0 0 0 1 1 1
```

matrix_row_sum.txt matrix.txt

matrix_row_sum.txt

```
Sum of Row 1: 9
Sum of Row 2: 5
Sum of Row 3: 6
Sum of Row 4: 9
Sum of Row 5: 7
```

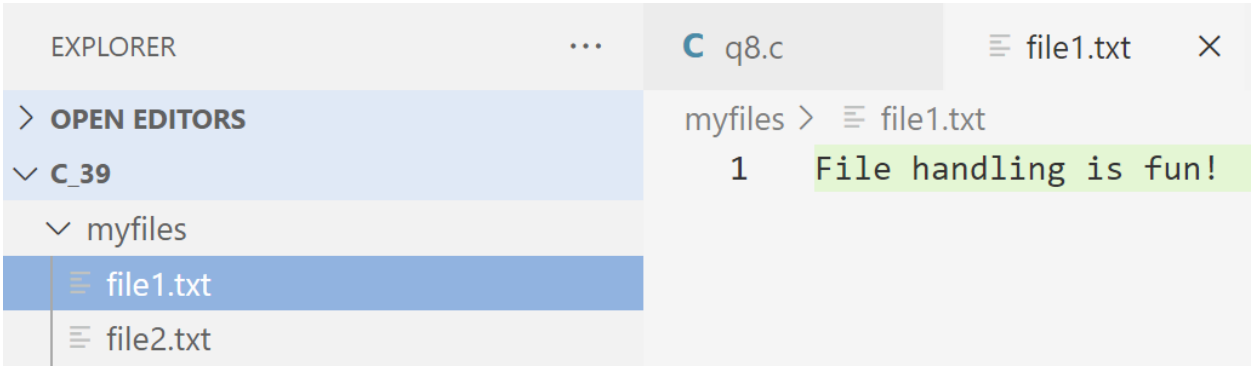
Q13. *Write a program to read a text file, convert each character to uppercase and write it to another file.*

CODE:

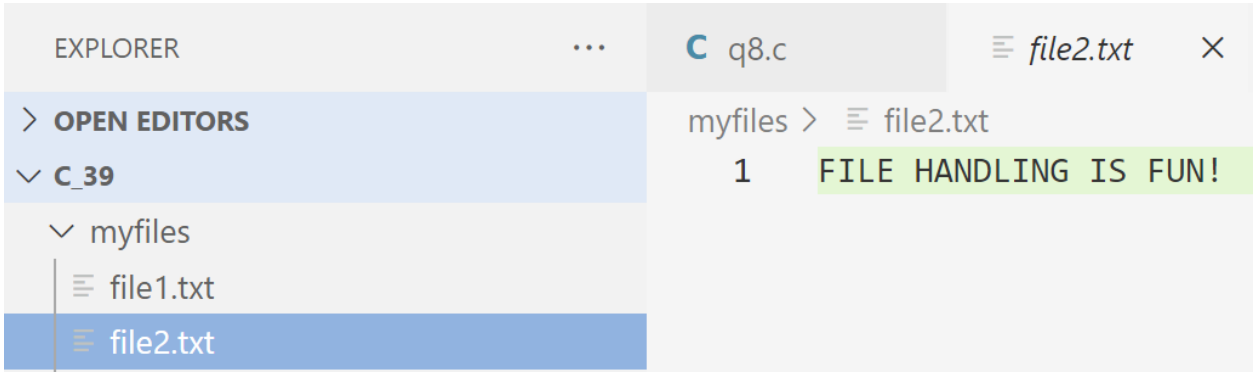
```c
#include <stdio.h>
#include <stdlib.h>
void copyUpper(FILE *t, FILE *d);
char convertUpper(char c);
int main()
{
    FILE *file1, *file2;
    file1 = fopen("myfiles/file1.txt", "r");
    file2 = fopen("myfiles/file2.txt", "w");
    copyUpper(file1, file2);
    printf("File saved successfully!");
    fclose(file1);
    fclose(file2);
    return 0;
}
void copyUpper(FILE *t, FILE *d)
{
    char temp;
    while(1)
    {
        temp = getc(t);
        if(temp != EOF)
        {
            if(temp >=97 && temp <=122)
            {
                putc(convertUpper(temp), d);
            }
            else
            {
                putc(temp, d);
            }
        }
        else
        {
            break;
        }
    }
```

```
}
char convertUpper(char c)
{
    return (c-32);
}
```

OUTPUT:




```
C:\MCA-Codes\C\C_39>q
File saved successfully!
```



Q14. *Write a menu driven program that depicts the working of a library. The menu options should be:*

8. *Add book information*
9. *Display book information*
10. *List all books of given author*
11. *List the title of specified book*
12. *List the count of books in the library*
13. *List the books in the order of accession number*
14. *Exit*

Create a structure called 'library' to store accession number, title of the book, author name, price of the book, and flag indicating whether the book is issued or not. Consider that there are more than 1000 records.

CODE:

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

#define MAX_BOOKS 512

typedef struct{
  int accession_number;
  char *title;
  char *author;
  float price;
  int issued;
}Library;

int book_count = 0;

void add_book_info(Library**);
void display_book_information(Library**);
void list_books_by_author(Library **);
void list_given_book_title(Library **);
void list_all_by_order(Library **);
void swap_elements(Library*, Library*);

int main(){
    int i, choice;
```

.....

```
    char hack;
    Library **books;
    books = (Library**)malloc(sizeof(Library*) * MAX_BOOKS);
    for (i = 0; i < MAX_BOOKS; i++){
        books[i] = (Library*)malloc(sizeof(Library));
    }

    while (1){
        printf("\npress 1 Add book information");
        printf("\npress 2 Display book information");
        printf("\npress 3 List all books of given author");
        printf("\npress 4 List the title of specified book");
        printf("\npress 5 List the count of books in the library");
        printf("\npress 6 List the books in the order of accession number");
        printf("\npress 7 or any other key to exit ...\n");
        scanf("%d", &choice);
        switch (choice)
        {
            case 1:
                add_book_info(books);
                scanf("%c", &hack);
                break;
            case 2:
                display_book_information(books);
                scanf("%c", &hack);
                break;
            case 3:
                list_books_by_author(books);
                // scanf("%c", &hack);
                break;
            case 4:
                list_given_book_title(books);
                break;
            case 5:
                printf("Total book count: %d", book_count);
                break;
            case 6:
                list_all_by_order(books);
                break;
            default:
                printf("\nExiting ....\n");
                exit(1);
        }
```

```
    }

    return 0;
}

void add_book_info(Library **book){
    char *hack;
    if (book_count == MAX_BOOKS) {
        printf("Error: Library is full.\n");
        return;
    }

    printf("Enter accession number: ");
    scanf("%d", &book[book_count]->accession_number);
    scanf("%c", &hack);

    book[book_count]->title = (char*)malloc(sizeof(char)*50);
    printf("Enter title: ");
    gets(book[book_count]->title);

    book[book_count]->author = (char*)malloc(sizeof(char)*50);
    printf("Enter author: ");
    gets(book[book_count]->author);


    printf("Enter price: ");
    scanf("%f", &book[book_count]->price);

    book[book_count]->issued = 0;
    book_count++;
    printf("Book added successfully.\n");
}


void display_book_information(Library **book) {
  int accession_number;
  printf("Enter accession number: ");
  scanf("%d", &accession_number);

  for (int i = 0; i < book_count; i++) {
        if (book[i]->accession_number == accession_number) {
            printf("Accession number: %d\n", book[i]->accession_number);
            printf("Title: %s\n", book[i]->title);
```

```
            printf("Author: %s\n", book[i]->author);
            printf("Price: %.2f\n", book[i]->price);
            printf("Issued: %s\n", book[i]->issued ? "Yes" : "No");
            return;
        }
  }

  printf("Book not found.\n");
}

void list_books_by_author(Library **book)
{
    char author[50], *hack;
    scanf("%c", &hack);
    printf("Enter author: ");
    gets(author);

    int found = 0;
    for (int i = 0; i < book_count; i++)
    {
        if (strcmp(book[i]->author, author) == 0)
        {
            printf("\n----- Book found of given auther -----\n");
            printf("Accession number: %d\n", book[i]->accession_number);
            printf("Title: %s\n", book[i]->title);
            printf("Author: %s\n", book[i]->author);
            printf("Price: %.2f\n", book[i]->price);
            printf("Issued: %s\n", book[i]->issued ? "Yes" : "No");
            found = 1;
        }
    }

    if (!found)
    {
        printf("No books found by that author.\n");
    }
}

void list_given_book_title(Library **book){
    int accession_number;
    printf("Enter accession number: ");
    scanf("%d", &accession_number);
    for (int i = 0; i < book_count; i++) {
```

```
        if (book[i]->accession_number == accession_number) {
            printf("\nBook found having assession no %d \n", accession_number);
            printf("Title: %s\n", book[i]->title);
            return;
        }
    }
    printf("Book doesnt exist.\n");
}

void list_all_by_order(Library **book){
    int i, j;
    for(i = 0; i < book_count - 1; i++){
        for( j = 0; j < book_count - i - 1; j++){
            if (book[j]->accession_number > book[j + 1]->accession_number){
                swap_elements(book[j], book[j + 1]);
            }
        }
    }
    printf("\nBook listing by order of the accession no -----\n");
    for (i = 0; i < book_count; i++)
    {
        printf("Accession number: %d\n", book[i]->accession_number);
        printf("Title: %s\n", book[i]->title);
        printf("Author: %s\n", book[i]->author);
        printf("Price: %.2f\n", book[i]->price);
        printf("Issued: %s\n", book[i]->issued ? "Yes" : "No");
    }

}

void swap_elements(Library *book1, Library *book2){
    int accession_number = book1->accession_number;
    char *title = book1->title;
    char *author = book1->author;
    float price = book1->price;
    int issued = book1->issued;

    book1->accession_number = book2->accession_number;
    book1->title = book2->title;
    book1->author = book2->author;
    book1->price = book2->price;
    book1->issued = book2->issued;
```

```
    book2->accession_number = accession_number;
    book2->title = title;
    book2->author = author;
    book2->price = price;
    book2->issued = issued;
}
```

OUTPUT:

```
C:\MCA-Codes>.\lib

press 1 Add book information
press 2 Display book information
press 3 List all books of given author
press 4 List the title of specified book
press 5 List the count of books in the library
press 6 List the books in the order of accession number
press 7 or any other key to exit ...
1
Enter accession number: 1011
Enter title: The C programming language
Enter author: Dennis Rechie
Enter price: 1000
Book added successfully.

press 1 Add book information
press 2 Display book information
press 3 List all books of given author
press 4 List the title of specified book
press 5 List the count of books in the library
press 6 List the books in the order of accession number
press 7 or any other key to exit ...
1
Enter accession number: 1012
Enter title: Let us C
Enter author: Yashavant Kanetkar
Enter price: 560
Book added successfully.

press 1 Add book information
press 2 Display book information
press 3 List all books of given author
press 4 List the title of specified book
press 5 List the count of books in the library
press 6 List the books in the order of accession number
press 7 or any other key to exit ...
2
Enter accession number: 1011
Accession number: 1011
Title: The C programming language
Author: Dennis Rechie
Price: 1000.00
Issued: No
```

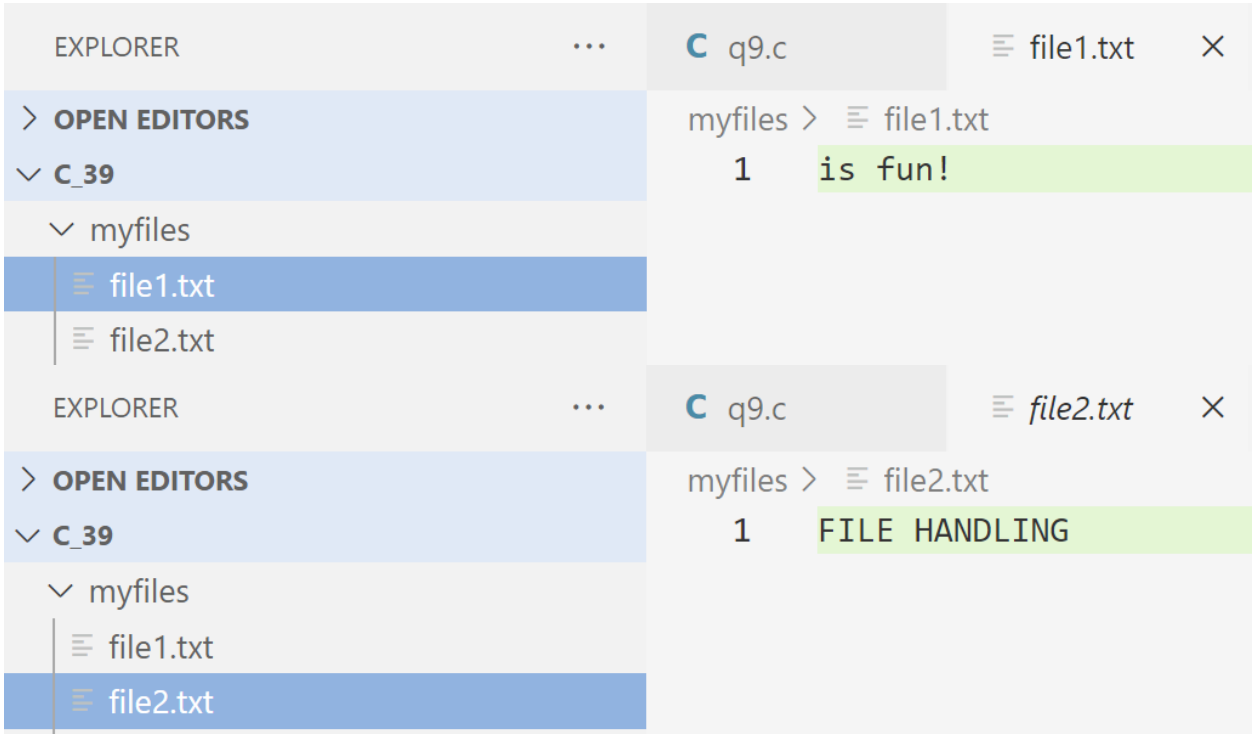
```
press 1 Add book information
press 2 Display book information
press 3 List all books of given author
press 4 List the title of specified book
press 5 List the count of books in the library
press 6 List the books in the order of accession number
press 7 or any other key to exit ...
5
Total book count: 2
press 1 Add book information
press 2 Display book information
press 3 List all books of given author
press 4 List the title of specified book
press 5 List the count of books in the library
press 6 List the books in the order of accession number
press 7 or any other key to exit ...
4
Enter accession number: 1012

Book found having assession no 1012
Title: Let us C

press 1 Add book information
press 2 Display book information
press 3 List all books of given author
press 4 List the title of specified book
press 5 List the count of books in the library
press 6 List the books in the order of accession number
press 7 or any other key to exit ...
6

Book listing by order of the accession no -----
Accession number: 1011
Title: The C programming language
Author: Dennis Rechie
Price: 1000.00
Issued: No
Accession number: 1012
Title: Let us C
Author: Yashavant Kanetkar
Price: 560.00
Issued: No

press 1 Add book information
press 2 Display book information
press 3 List all books of given author
press 4 List the title of specified book
press 5 List the count of books in the library
press 6 List the books in the order of accession number
press 7 or any other key to exit ...
```

Q15. *Write a program to add the contents of one file at the end of another file.*

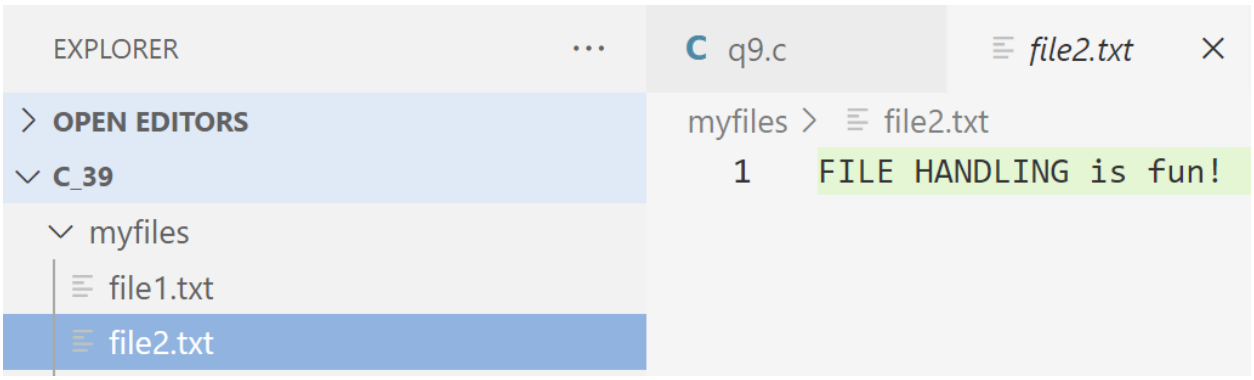
CODE:

```c
#include <stdio.h>
#include <stdlib.h>
void copy(FILE *t, FILE *d);
int main()
{
    FILE *file1, *file2;
    file1 = fopen("myfiles/file1.txt", "r");
    file2 = fopen("myfiles/file2.txt", "a");
    copy(file1, file2);
    fclose(file1);
    fclose(file2);
    return 0;
}
void copy(FILE *t, FILE *d)
{
    char temp;
    while(1)
    {
        temp = getc(t);
        if(temp != EOF)
        {
            putc(temp, d);
        }
        else
        {
            break;
        }
    }
}
```

OUTPUT:




```
C:\MCA-Codes\C\C_39>q
File modified successfully!
```



Q16. *Write a C++ program to create a class string, which stores string with constructor, displays the string and joins two strings with a join user defined function taking two arguments of string object.*

.....

CODE:

```
#include <iostream>
#include <cstdlib>
#include <climits>
#include <cmath>
#include <cstring>
using namespace std;
class String
{
    private:
        int i, size;
        char *in;
    public:
        String()
        {
            i = 0;
        }
        String(const char in[])
        {
            i = 0;
            size = strlen(in);
            this->in = (char *)malloc(size*sizeof(char));
            while(i<size)
            {
                *(this->in + i) = *(in + i);
                i++;
            }
            *(this->in + i) = '\0';
        }
        void dispStr()
        {
            fputs(this->in, stdout);
        }
        void join(String a)
        {
            i = 0;
```

.....

```
            this->in = (char*)realloc(this->in, (a.size+size)*sizeof(char));
            while(i < a.size)
            {
                *(this->in + i + size) = *(a.in + i);
                i++;
            }
            *(this->in + i + size) = '\0';
        }
};
int main()
{
    String one("Hello "), two("World.");
    one.dispStr();
    cout << endl;
    two.dispStr();
    cout << endl;
    one.join(two);
    one.dispStr();
    cout << endl;
    return 0;
}
```

OUTPUT:

```
C:\MCA-Codes\C++\Set 4>q
Hello
World.
Hello World.
```

Q17. *Write a C++ program to demonstrate (A) Copy Constructor (B) Parameterized Constructor (C) Virtual destructor*

CODE:

```
#include <iostream>
using namespace std;

class Base {
    protected:
        int x;
    public:
        Base(){
            x = 0;
        }
        // Parameterized constructor
        Base(int x) {
            cout << "Base class parameterized constructor: " << x << endl;
            this->x = x;
        }

        // Copy constructor
        Base(const Base& obj) {
            cout << "Base class copy constructor called." << endl;
            x = obj.x;
        }

        // Virtual destructor
        virtual ~Base() {
            cout << "Base class destructor called." << endl;
        }

        void set_val(int x){
            this->x = x;
        }

        void display(){
            cout << "Value in Base class: " << x << endl;
        }
};

class Derived : public Base {
```

.....

```
    protected:
        int y;
    public:
        // Parameterized constructor
        Derived(int x, int y) : Base(x) {
            cout << "Derived class parameterized constructor: " << y << endl;
            this->y = y;
        }

        //derived destructor
        ~Derived() {
            cout << "Derived class destructor called." << endl;
        }
};

int main() {
    // Derived d1(10, 20);
    // cout << "Creating copy of d1" << endl;
    // Derived d2 = d1;

    Base b1(10);
    Base b2 = b1;


    b1.display();
    b1.set_val(1);
    b1.display();
    b2.display();


    // polymorphism situation
    // Not using virtual destructor, will not call the destructor of derived
class
    // using the Virtual destructor, calls virtual destructor, therefore no
memory leaks

    Base *b3 = new Derived(10, 12);
    delete b3;

    return 0;
}
```

OUTPUT:

```
C:\MCA-Codes>.\virtual
Base class parameterized constructor: 10
Base class copy constructor called.
Value in Base class: 10
Value in Base class: 1
Value in Base class: 10
Base class parameterized constructor: 10
Derived class parameterized constructor: 12
Derived class destructor called.
Base class destructor called.
Base class destructor called.
Base class destructor called.

C:\MCA-Codes>
```

Q18. *Write a C++ program to overload operator '+' to concatenate two strings and hence reverse the concatenated string.*

CODE:

```
#include <iostream>
using namespace std;
class MyString
{
    private:
        char *core;
    public:
        MyString()
        {

        }
        MyString(const char *f)
        {
            core = (char *)malloc(len(f)*sizeof(char));
            store(core, f);
        }
        MyString operator+(MyString addent)
        {
            int cursor = 0;
            MyString result;
            result.core = (char
*)malloc((len(this->core)+len(addent.core))*sizeof(char));
            store((result.core+cursor), this->core);
            cursor = len(this->core);
            store((result.core + cursor), addent.core);
            rev(result.core);
            return result.core;
        }
        void disp()
        {
            cout << core << endl;
        }
        void store(char *subject, const char *object)
        {
            int cursor = 0;
            while(*(object + cursor) != '\0')
            {
```

```
                *(subject + cursor) = *(object + cursor);
                cursor++;
            }
            *(subject+cursor) = '\0';
        }
        int len(const char *subject)
        {
            int count = 0;
            while(*(subject + count) != '\0')
            {
                count++;
            }
            return count;
        }
        void rev(char *subject)
        {
            char *temp;
            int cursorleft = 0, cursorright = len(subject);
            int whilec = cursorright;
            temp = (char *)malloc(cursorright*sizeof(char));
            while(cursorleft < whilec)
            {
                *(temp + cursorleft) = *(subject + cursorright-1);
                cursorleft++;
                cursorright--;
            }
            *(temp + cursorleft) = '\0';
            store(subject, temp);
        }
};
int main()
{
    MyString s1("C++ "), s2("Assignment");
    s1.disp();
    s2.disp();
    MyString s3 = s1 + s2;
    s3.disp();
    return 0;
}
```

OUTPUT:

```
C:\MCA-Codes\C++\Set 4>q
C++
Assignment
tnemngissA ++C
```

Q19. *Write a C++ program to overload the following operators*

a) '>>' to accept time from user (in hours: mins:sec)
b) '+' to add two different user-given times.
c) '<<' to display the time in hours: mins: sec format.
d) '==' to check whether two user-given times are equal or not.

CODE:

```
#include <iostream>
using namespace std;

class Time {
    private:
        int hours;
        int minutes;
        int seconds;
    public:
        // Overloaded '>>' operator to accept time from user
        friend istream& operator>>(istream& input, Time& t) {
            input >> t.hours >> t.minutes >> t.seconds;
            return input;
        }
        // Overloaded '+' operator to add two different user-given times
        Time operator+(const Time& t) {
            Time temp;
            temp.seconds = seconds + t.seconds;
            temp.minutes = minutes + t.minutes + temp.seconds / 60;
            temp.hours = hours + t.hours + temp.minutes / 60;
            temp.seconds %= 60;
            temp.minutes %= 60;
            return temp;
        }
        // Overloaded '<<' operator to display time in hours:mins:sec format
        friend ostream& operator<<(ostream& output, Time& t) {
            output << t.hours << ":" << t.minutes << ":" << t.seconds;
            return output;
        }
        // Overloaded '==' operator to check whether two user-given times are
equal or not
        bool operator==(const Time& t) {
            return (hours == t.hours && minutes == t.minutes && seconds ==
t.seconds);
        }
```

```
};
int main() {
    Time t1, t2, t3;
    cout << "Enter time 1 (hours:mins:secs): ";
    cin >> t1;
    cout << "Enter time 2 (hours:mins:secs): ";
    cin >> t2;

    t3 = t1 + t2; // Add t1 and t2 using overloaded '+' operator

    cout << "Time 1: " << t1 << endl;
    cout << "Time 2: " << t2 << endl;
    cout << "Time 1 + Time 2: " << t3 << endl;

    if (t1 == t2) // Compare t1 and t2 using overloaded '==' operator
        cout << "Time 1 and Time 2 are equal." << endl;
    else
        cout << "Time 1 and Time 2 are not equal." << endl;

    return 0;
}
```

OUTPUT:

```
C:\MCA-Codes>g++ time.cpp -o time

C:\MCA-Codes>.\time
Enter time 1 (hours:mins:secs): 12 12 24
Enter time 2 (hours:mins:secs): 15 45 23
Time 1: 12:12:24
Time 2: 15:45:23
Time 1 + Time 2: 27:57:47
Time 1 and Time 2 are not equal.

C:\MCA-Codes>.\time
Enter time 1 (hours:mins:secs): 12 15 23
Enter time 2 (hours:mins:secs): 12 15 23
Time 1: 12:15:23
Time 2: 12:15:23
Time 1 + Time 2: 24:30:46
Time 1 and Time 2 are equal.

C:\MCA-Codes>
```

Q20. *Write a C++ program to add two complex numbers using the friend function.*

CODE:

```
#include <iostream>
#include <cstdlib>
#include <climits>
#include <cmath>
using namespace std;
class Complex
{
    private:
        int real;
        int imaginerycross;
    public:
        Complex()
        {
            real = 0;
            imaginerycross = 0;
        }
        Complex(int a)
        {
            real = a;
            imaginerycross = 0;
        }
        Complex(int a, int b)
        {
            real = a;
            imaginerycross = b;
        }
        void show()
        {
            cout << real << "+" << imaginerycross << "i" << endl;
        }
        friend void sum(Complex, Complex, Complex*);
};
void sum(Complex a, Complex b, Complex *c)
{
    c->real = a.real + b.real;
    c->imaginerycross = a.imaginerycross + b.imaginerycross;
}
int main()
```

.....

```
{

    Complex A(5,6), B(6), C;
    A.show();
    B.show();
    C.show();
    sum(A, B, &C);
    C.show();
    return 0;
}
```

OUTPUT:

```
C:\MCA-Codes\C++\Set 4>q
5+6i
6+0i
0+0i
11+6i
```