

**Course Code : MCS-011**

**Course Title : Problem Solving and Programming**

**Assignment Number : MCA(1)/011/Assign/12**

**Maximum Marks : 100**

**Weightage : 25%**

**Last Dates for Submission : 15<sup>th</sup> October, 2012 (For July 2012 Session)**

**15<sup>th</sup> April, 2013 (For January 2013 Session)**

There are five questions in this assignment, which carries 80 marks. Rest 20 marks are for viva-voce. Answer all the questions. You may use illustrations and diagrams to enhance the explanations.

Please go through the guidelines regarding assignments given in the Programme Guide for the format of presentation.

**Question 1: Write a program to print the following pattern:**

a)	1	b)	1
	1 2		2 2
	1 2 3		3 3 3
	1 2 3 4		4 4 4 4
	1 2 3 4 5		5 5 5 5 5

**Solution (a) :**

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

```
#include<conio.h>
```

```
void main()
```

```
{
```

```
    int i,j,num_lines;
```

```
    clrscr();
```

```
    printf("\nEnter the number of lines to print: ");
```

```
    scanf("%d", &num_lines);
```

```
    for(i=1; i<=num_lines; i++)
```

```
    {
```

```
        for(j=1; j<=i; j++)
```

```
        {
```

```
            printf("%d ", j);
```

```
        }
```

```
        printf("\n");
```

```
    }
```

```
    getch();
```

```
}
```

**Solution 1 (b) :**

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

```
#include<conio.h>
```

```
void main()
```

```
{
```

```
    int i,j,num_lines;
```

```
    clrscr();
```

```
    printf("\nEnter the number of lines to print: ");
```

```
    scanf("%d", &num_lines);
```

```
    for(i=1; i<=num_lines; i++)
```

```
    {
```

```
        for(j=1; j<=i; j++)
```

```
        {
```

```
            printf("%d ", i);
```

```
        }
```

```
        printf("\n");
```

```
    }
```

```
    getch();
```

```
}
```

**Question 2: Declare two arrays A and B, find**

**(i) A intersection B and (ii) A union B.**

**Solution :**

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

```
#include<conio.h>
```

```
#define SIZE 5
```

```
void get_data(int arr[]);
```

```
void print_data(int arr[], int n);
```

```
void bubble_sort(int arr[]);
```

```
int find_intersection(int array_1[], int array_2[], int intersect_result[]);
```

```
int find_union(int array_1[], int array_2[], int union_result[]);
```

```
void main()
```

```
{
```

```
    int array_1[SIZE], array_2[SIZE], intersect_result[SIZE], union_result[SIZE*2];
```

```
    int num_elements;
```

```
    clrscr();
```

```
    //Get the elements of Array1
```

```
    printf("\nEnter the elements of Array 1: \n");
```

```
    get_data(array_1);
```

```
    printf("\n\nElements of Array 1: ");
```

```
    print_data(array_1, SIZE);
```

```
//Sort array 1
```

```
bubble_sort(array_1);
```

```
printf("\n\nSorted elements of Array 1: ");
```

```
print_data(array_1, SIZE);
```

```
//Get the elements of Array2
```

```
printf("\n\nEnter the elements of Array 2: \n");
```

```
get_data(array_2);
```

```
printf("\n\nElements of Array 2: ");
```

```
print_data(array_2, SIZE);
```

```
//Sort array 2
```

```
bubble_sort(array_2);
```

```
printf("\n\nSorted elements of Array 2: ");
```

```
print_data(array_2, SIZE);
```

```
//Find Intersection and print the result
```

```
num_elements = find_intersection(array_1, array_2, intersect_result);
```

```
printf("\n\nIntersection is: ");
```

```
print_data(intersect_result, num_elements);
```

```
//Find Union
```

```
num_elements = find_union(array_1, array_2, union_result);
```

```
printf("\n\nUnion is: ");
```

```
    print_data(union_result, num_elements);

    getch();
}
```

```
void get_data(int arr[])
{
    int i,j;
    for(i=0; i<SIZE; i++)
    {
        j = i+1;
        printf("\nEnter element %d: ",j);
        scanf("%d", &arr[i]);
    }
}
```

```
void print_data(int arr[], int n)
{
    int i; printf("{ ");
    for(i=0; i<n; i++)
    {
        printf("%d ",arr[i]);
    }
    printf("}");
}
```

```
}
```

```
void bubble_sort(int arr[])
```

```
{
```

```
    int i,j,temp,swapped;
```

```
    for(i=1; i<SIZE; i++)
```

```
    {
```

```
        swapped = 0;
```

```
        for(j=0; j<SIZE-i; j++)
```

```
        {
```

```
            if(arr[j] > arr[j+1])
```

```
            {
```

```
                temp = arr[j];
```

```
                arr[j] = arr[j+1];
```

```
                arr[j+1] = temp;
```

```
                swapped = 1;
```

```
            }
```

```
        }
```

```
        if(swapped == 0)
```

```
        {
```

```
            break;
```

```
        }
```

```
    }
```

```
}
```

```
int find_intersection(int array_1[], int array_2[], int intersect_result[])
```

```
{
```

```
    int i = 0, j = 0, k = 0;
```

```
    while((i<SIZE) && (j<SIZE))
```

```
    {
```

```
        if(array_1[i] < array_2[j])
```

```
        {
```

```
            i++;
```

```
        }
```

```
        else if(array_1[i] > array_2[j])
```

```
        {
```

```
            j++;
```

```
        }
```

```
        else
```

```
        {
```

```
            intersect_result[k] = array_1[i];
```

```
            i++;
```

```
            j++;
```

```
            k++;
```

```
        }
```

```
    }
```

```
    return(k);
```



```
}
```

```
int find_union(int array_1[], int array_2[], int union_result[])
```

```
{
```

```
    int i = 0, j = 0, k = 0;
```

```
    while((i<SIZE) && (j<SIZE))
```

```
    {
```

```
        if(array_1[i] < array_2[j])
```

```
        {
```

```
            union_result[k] = array_1[i];
```

```
            i++;
```

```
            k++;
```

```
        }
```

```
        else if(array_1[i] > array_2[j])
```

```
        {
```

```
            union_result[k] = array_2[j];
```

```
            j++;
```

```
            k++;
```

```
        }
```

```
        else
```

```
        {
```

```
            union_result[k] = array_1[i];
```

```
            i++;
```

```
            j++;
```

```
            k++;
```

```
    }  
}  
  
if(i == SIZE)  
{  
    while(j<SIZE)  
    {  
        union_result[k] = array_2[j];  
        j++;  
        k++;  
    }  
}  
else  
{  
    while(i<SIZE)  
    {  
        union_result[k] = array_1[i];  
        i++;  
        k++;  
    }  
}  
  
return(k);  
}
```

**Question 3:** Write a program to crypt its input according to a specified transformation scheme. The transformation scheme will consist of two strings: a string of characters and then a string of replacement characters. The idea is that your program replaces every instance of the *i*th character in the initial string with the (*i*+2) character (of English alphabets) in the replacement string. When no substitution is defined for a character, the program just passes it through to the output unchanged. Blank spaces and the other symbols remains the same. The program should inform the user of any errors in the transformation scheme. Your program should display the phrase before and after the substitutions have been made.

**Example:**

**Original String:** I know C programming.

**String after the transformation:** K mpqy E rtqitcookpi.

**Solution :**

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

```
#include<conio.h>
```

```
void read_string(char str[]);
```

```
void crypt(char str[], int transform_scheme);
```

```
void main()
```

```
{
```

```
    int transform_scheme;
```

```
    char str[100];
```

```
    clrscr();
```

```
    printf("\nEnter a string (Not more than 100 chars): ");
```

```
    //Read data from the user
```

```
read_string(str);

printf("\nOriginal String is: %s\n", str);


//Read transformation scheme from the user

printf("\nEnter the transaformation scheme (In Integer): ");

scanf("%d", &transform_scheme);


//Logic to convert the input string to encrypted form

crypt(str, transform_scheme);

printf("\nConverted String is: %s\n", str);

getch();
}

void read_string(char str[])
{
    int i, temp;

    i = 0;
    while((temp = getchar()) != '\n')
    {
        str[i] = temp;
        i = i+1;
    }

    str[i] = '\0';
}
```

```
void crypt(char str[], int transform_scheme)
{
    int i,len;
    len = strlen(str);

    for(i=0; i<len; i++)
    {
        if(str[i] >= 'A' && str[i] <= 'Z')
        {
            str[i] = str[i] + transform_scheme;

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

    if(str[i] >= 'a' && str[i] <= 'z')
    {
        str[i] = str[i] + transform_scheme;
```

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

        if(str[i] < 'a')
        {
            str[i] = 'z' - (('a' - str[i]) - 1);
        }
    }
}
```

**Question 4:** Write an interactive program called “DISTANCE CONVERTER” that accepts the distance in millimetres / feet / miles / yards / kilometres and displays its equivalent in metres.

**Solution :**

```
#include <stdio.h>
#include <conio.h>
void main ( )
{
    float meter, dis;
    int ch;

    clrscr ( );

    printf ("\n Press 1 to input the distance in millimeters.");

    printf ("\n Press 2 to input the distance in feet.");
```

```
printf ("\n Press 3 to input the distance in miles.");

printf ("\n Press 4 to input the distance in yards.");

printf ("\n Press 5 to input the distance in kilometers.");

printf ("\n Enter the choice..");

scanf ("%d",&ch);

switch ( ch )

{

    case 1: printf ("\n Input the distance in millimeters...");

            scanf ("%f",&dis);

            meter = dis/1000;

            printf ("\n Distance in meter =%f",meter);

            break;

    case 2: printf ("\n Input the distance in feet...");

            scanf ("%f",&dis);

            meter = dis*.3048;

            printf ("\n Distance in meter =%f",meter);

            break;

    case 3: printf ("\n Input the distance in miles...");

            scanf ("%f",&dis);

            meter = dis*1.60934*1000;

            printf ("\n Distance in meter =%f",meter);

            break;

    case 4: printf ("\n Input the distance in yards...");

            scanf ("%f",&dis);

            meter = dis*.9144;
```

```
        printf ("\n Distance in meter =%f",meter);

        break;

    case 5: printf ("\n Input the distance in kilometers...");

        scanf ("%f",&dis);

        meter = dis*1000;

        printf ("\n Distance in meter =%f",meter);

        break;

    default: printf("\n Wrong choice.");

}

getch ( );

}
```

**Question 5: Write an interactive program to generate progress reports for the students of class XII (Science group). Assumptions can be made wherever necessary.**

**Solution :**

```
#include <stdio.h>

#include <conio.h>

void main ( )

{

    float meter, dis;

    int ch;

    clrscr ( );

    printf ("\n Press 1 to input the distance in millimeters.");

    printf ("\n Press 2 to input the distance in feet.");

    printf ("\n Press 3 to input the distance in miles.");

    printf ("\n Press 4 to input the distance in yards.");
```



```
printf ("\n Press 5 to input the distance in kilometers.");

printf ("\n Enter the choice..");

scanf ("%d",&ch);

switch ( ch )

{

    case 1: printf ("\n Input the distance in millimeters..");

            scanf ("%f",&dis);

            meter = dis/1000;

            printf ("\n Distance in meter =%f",meter);

            break;

    case 2: printf ("\n Input the distance in feet..");

            scanf ("%f",&dis);

            meter = dis*.3048;

            printf ("\n Distance in meter =%f",meter);

            break;

    case 3: printf ("\n Input the distance in miles..");

            scanf ("%f",&dis);

            meter = dis*1.60934*1000;

            printf ("\n Distance in meter =%f",meter);

            break;

    case 4: printf ("\n Input the distance in yards..");

            scanf ("%f",&dis);

            meter = dis*.9144;

            printf ("\n Distance in meter =%f",meter);

            break;
```

```
case 5: printf ("\n Input the distance in kilometers..:");  
  
        scanf ("%f",&dis);  
  
        meter = dis*1000;  
  
        printf ("\n Distance in meter =%f",meter);  
  
        break;  
  
default: printf("\n Wrong choice.");  
  
}  
  
getch ( );  
  
}
```

VIJAY JOTANI(THANK TO ME ON 7383921223 BY SMS)