LAB MANUAL
On
Programming in C (13MCA16)
For MCA 1st Semester of VTU

Instructions to the Readers:

- This document is a useful material for the 1st Semester MCA students (2013 Scheme) of Visvesvaraya Technological University, Belagavi, Karnatka.
- Though these programs are compiled and verified on Turbo C compiler, they can be executed on different C compilers with little/no modifications.
- Clarifications/suggestions are welcome!!

1. a) Convert degrees into Fahrenheit and vice versa.
b) Calculate the salary of an employee given his basic pay, HRA = 10% of basic pay, TA=5% of his basic pay and deductions IT = 2.5% of his basic pay.

```
//Program to convert degrees into Fahrenheit and vice versa
#include<stdio.h>
#include<conio.h>
void main()
{
        float c,f;
        clrscr();

        printf("Enter the temperature in centigrade: ");
        scanf("%f",&c);
        f = 9.0/5.0 *c+32;
        printf("\n%.2f Centigrade =%.2f Fahrenheit",c,f);

        printf("\n\n Enter the temperature in Fahrenheit");
        scanf("%f",&f);
        c=5.0/9.0 *(f-32);

        printf("\n%.2f Fahrenheit=%.2f Centigrade",f,c);
        getch();
}
```

Output:

```
Enter the temperature in centigrade: 47
47.00 Centigrade =116.60 Fahrenheit
```

```
Enter the temperature in Fahrenheit: 120
120.00 Fahrenheit=48.89 Centigrade
```

```
//Program to compute salary of an employee
#include<stdio.h>
#include<conio.h>

void main()
{
        float hra,ta,basic,it,netsal,gross;
        clrscr();

        printf("Enter the basic salary\n");
        scanf("%f",&basic);

        if(basic<0)
        {
                printf("Basic Salary must be greater than zero!!");
                getch();
                exit(0);
        }
        hra= 0.1*basic;
        ta= 0.05*basic;
        gross= basic+hra+ta;
        it= 0.025*basic;
        netsal = gross-it;

        printf("\nThe net salary = %f", netsal);
        getch();
}
```

Output:

```
        Enter the basic salary: 10000
        The net salary = 11250.000000
```

2. a) Check whether the given number is perfect number.
b) Solve quadratic equations for the given values of a, b, c.

```
//check for perfect number
#include<stdio.h>
#include<conio.h>
void main()
{
        int n,k=0,i;
```

```
        clrscr();

        printf("Enter the number\n");
        scanf("%d",&n);

        for(i=1;i<=n/2;i++)
                if((n%i)==0)
                        k+=i;

        if(k==n)
                printf("%d is perfect number\n",n);
        else
                printf("%d is not perfect number\n",n);

        getch();
}
```

Output:

```
Enter the number: 28
28 is perfect number

Enter the number:35
35 is not perfect number
```

//Quadratic equation

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
void main()
{
        float a, b, c, desc,r1,r2,real,img;
        clrscr();

        printf("Enter the values of a,b,c:\n");
        scanf("%f%f%f",&a,&b,&c);

        if(a==0)
        {
                printf("\nEquation is linear and has only one root. ");
                printf("\nAnd the root is %5.2f", -c/b);
                getch();
        }
        else
        {
                desc=b*b-4*a*c;
                if(desc==0)
                {
```

```
                r1=-b/(2*a);
                r2=-b/(2*a);
                printf("\nThe roots are real and equal");
                printf("\n r1 = %5.2f \t r2 = %5.2f",r1,r2);
                getch();
        }
        else if(desc>0)
        {
                r1=(-b+sqrt(desc))/(2*a);
                r2=(-b-sqrt(desc))/(2*a);
                printf("\nThe roots are real and distinct");
                printf("\n r1=%5.2f\t r2=%5.2f",r1,r2);
                getch();
        }
        else
        {
                real=-b/(2*a);
                img=sqrt(-desc)/(2*a);
                printf("\nThe roots are imaginary");
                printf("\n r1 =%.2f + %.2fi  \n r2 = %.2f %.2fi",real,img,real,img);
                getch();
        }
    }
}
```

Output:

```
Enter the values of a,b,c:
0       4       2
Equation is linear and has only one root.
And the root is -0.50

Enter the values of a,b,c:
1       4       4
The roots are real and equal
 r1 = -2.00     r2 = -2.00

Enter the values of a,b,c:
1       4       1
The roots are real and distinct
 r1=-0.27       r2=-3.73

Enter the values of a,b,c:
4       1       4
The roots are imaginary
 r1 = -0.12 +  0.99i
 r2 = -0.12 -  0.99i
```

3. a) Generate all Armstrong numbers up to n.
b) Convert a decimal number to a hexadecimal number.

```
//Armstrong number
#include <stdio.h>

void main()
{
	int r, n, i, sum=0, temp;

	clrscr();
	printf("Enter a number:\n");
	scanf("%d",&n);

	printf("Armstrong numbers from 1 to %d are:\n",n);

	for(i = 1; i<=n; i++)
	{
		temp = i;
		while( temp != 0 )
		{
			r = temp%10;
			sum = sum + r*r*r;
			temp = temp/10;
		}
		if (i == sum )
			printf("%d\n", i);

		sum = 0;
	}
	getch();
}
```

Output:

```
Enter a number: 1000
Armstrong numbers from 1 to 1000 are:
1
153
370
371
407
```

//Decimal to hexadecimal

```
#include<stdio.h>
#include<conio.h>
void main()
{
	int n, num, digit, i, j, temp;
	char hexa[100];
	clrscr();

	printf("Enter a decimal number:");
	scanf("%d", &num);

	n=num;

	i=0;
	while(n!=0)
	{
		temp=n%16;

		if(temp<10)
			temp= temp+48;
		else
			temp= temp+55;

		hexa[i++]=temp;

		n=n/16;
	}
	printf("\nThe decimal number %d in hexa = ",num);
	for(j=i-1;j>=0;j--)
		printf("%c", hexa[j]);

	getch();
}
```

Output:

```
	Enter a decimal number: 28
	The decimal number 28 in hexa = 1C
```

4. Write a menu driven C program to

a) Insert an element into an array

b) Delete an element from the array (first occurrence)

```
#include<stdio.h>
#incude<conio.h>
void main()
```

```
{
        int a[20], n, pos, i, item, opt;
        clrscr();

        printf("Enter number of elements:");
        scanf("%d", &n);
        printf("\nEnter array elements:");
        for(i=0;i<n;i++)
                scanf("%d", &a[i]);

        for(;;)
        {
                printf("\n 1. Insert item \n 2. Delete item");
                printf("\n 3. Display\n 4. Exit \n");
                printf("\nEnter your option:");
                scanf("%d",&opt);

                switch(opt)
                {
                        case 1: printf("\n Enter item to be inserted:");
                                scanf("%d", &item);
                                printf("\nEnter the position:");
                                scanf("%d", &pos);
                                for(i=n;i>=pos-1;i--)
                                        a[i]=a[i-1];
                                a[pos-1]=item;
                                n++;
                                break;
                        case 2: printf("\nEnter item to be deleted:");
                                scanf("%d",&item);
                                pos=0;
                                for(i=0;i<n;i++)
                                {
                                     if(a[i]==item)
                                     {
                                             pos=i+1;
                                             break;
                                     }
                                }
                                if(pos==0)
                                     printf("\nItem not found!!");
                                else
                                {
                                     for(i=pos-1;i<n;i++)
                                             a[i]=a[i+1];
                                     n--;
                                     printf("\nItem successfully deleted !!");
```

```
                        }
                        break;
                case 3: printf("\nThe array elements are:\n");
                        for(i=0;i<n;i++)
                              printf("%d\t",a[i]);
                        break;
                default: exit(0);
            }
        }
}
```

Output:

```
Enter number of elements:5
Enter array elements:
1       2       3       4       5

 1. Insert item
 2. Delete item
 3. Display
 4. Exit

Enter your option: 1
Enter item to be inserted:45
Enter the position:4

1. Insert item
2. Delete item
 3. Display
 4. Exit

Enter your option: 3
The array elements are:
1       2       3       45      4       5

1. Insert item
2. Delete item
3. Display
4. Exit
Enter your option: 2
Enter item to be deleted: 3
Item successfully deleted !!

1. Insert item
2. Delete item
3. Display
4. Exit
Enter your option: 3
```

```
The array elements are:
1       2       45      4       5

1. Insert item
2. Delete item
3. Display
4. Exit
Enter your option: 4
```

5. Write a Menu Driven C Program to

a) Accept a string from the user

b) Encode the string.

c) Decode the string

Apply the following procedure to encode it.

1. **Convert each character in a string to its ASCI value.**
2. **Add an integer value to it and display the encoded string**

Decode the string using reverse procedure and display.

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

void main()
{
        char str[100], str_en[100],str_de[100];
        int i, item, opt;
        clrscr();

        for(;;)
        {
                printf("\n 1. Read string \n 2. Encode \n 3. Decode\n 4.Exit \n");
                printf("Enter your option:");
                scanf("%d",&opt);
                fflush(stdin);

                switch(opt)
                {
                        case 1: printf("\nEnter a string:");
                                gets(str);
                                break;
                        case 2: printf("\nEnter an integer for encoding:");
                                scanf("%d", &item);
                                for(i=0;str[i]!='\0';i++)
                                      str[i]=str[i]+item;
                                printf("\nEncoded string is :");
                                puts(str);
```

```
                        break;
                case 3: printf("\nDecoded string is :");
                        for(i=0;str[i]!='\0'; i++)
                              str[i]=str[i]-item;
                        puts(str);
                        break;
                default: exit(0);
            }
        }
}
```

Output:

```
 1. Read string
 2. Encode
 3. Decode
 4.Exit
Enter your option:1
Enter a string: hello

 1. Read string
 2. Encode
 3. Decode
 4.Exit
Enter your option:2
Enter an integer for encoding:3
Encoded string is :khoor

 1. Read string
 2. Encode
 3. Decode
 4.Exit
Enter your option:3
Decoded string is :hello

 1. Read string
 2. Encode
 3. Decode
 4.Exit
Enter your option:4
```

6. Write a C program to multiply two matrices that satisfy the constraint of matrix multiplication.

```
//Matrix multiplication
#include<stdio.h>
```

```
#include<conio.h>

void read(int x[5][5],int m,int n)
{
	int i,j;

	for(i=0;i<m;i++)
		for(j=0;j<n;j++)
			scanf("%d",&x[i][j]);
}

void display(int x[5][5],int m,int n)
{
	int i,j;

	for(i=0;i<m;i++)
	{
		for(j=0;j<n;j++)
			printf("%d\t",x[i][j]);
		printf("\n");
	}
}

void multiply(int x[5][5],int y[5][5],int z[5][5],int m, int n, int q)
{
	int i,j,k;

	for(i=0;i<m;i++)
	{
		for(j=0;j<n;j++)
		{
			z[i][j]=0;
			for(k=0;k<q;k++)
				z[i][j]=z[i][j]+x[i][k]*y[k][j];
		}
	}
}

void main()
{
	int a[5][5],b[5][5],c[5][5];
	int m,n,p,q;
	clrscr();

	printf("Enter order of 1st matrix:");
	scanf("%d%d",&m,&n);
	printf("\nEnter order of 2nd matrix:");
```

```
        scanf("%d%d",&p,&q);

        if(n!=p)
        {
                printf("\nMultiplication is not possible");
                getch();
                exit(0);
        }
        else
        {
                printf("\nEnter the elements of first matrix:\n");
                read(a,m,n);
                printf("\nEnter the elements of second matrix:\n");
                read(b,p,q);

                multiply(a,b,c,m,n,q);
                printf("\nThe product matrix is:\n");
                display(c,m,q);
                getch();
        }
}
```

Output:

```
Enter order of 1st matrix: 2      3
Enter order of 2nd matrix: 2      4
Multiplication is not possible


Enter order of 1st matrix:2       3
Enter order of 2nd matrix:3       2

Enter the elements of first matrix:
1       2       3
4       5       6

Enter the elements of second matrix:
6       5
4       3
2       1

The product matrix is:
14      11
44      35
```

7. Write a C program to find the saddle point of a matrix.

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

```
#include<conio.h>
void main()
{
	int a[10][10],i,j,k,m,n,min,max,col;
	clrscr();

	printf("Enter order matrix\n");
	scanf("%d%d",&m,&n);
	printf("Enter matrix elements:\n");
	for(i=0;i<m;i++)
		for(j=0;j<n;j++)
			scanf("%d",&a[i][j]);

	for(i=0;i<m;i++)
	{
		min=a[i][0];
		for(j=0;j<n;j++)
		{
			if(a[i][j]<=min)
			{
				min=a[i][j];
				col=j;
			}
		}
		max=a[0][col];
		for(k=0;k<m;k++)
		{
			if(a[k][col]>=max)
				max=a[k][col];
		}

		if(max==min)
			printf("saddle point %d is at (%d,%d)",min, i+1,col+1);
	}
	getch();
}
```

```
Enter order matrix: 3	3
Enter matrix elements:
1	2	3
4	5	6
7	8	9
saddle point 7 is at (3,1)
```

8. Write a C program to implement a magic square of size n.

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

void main()
{
	int n, i, j, c, a[9][9] ;
	clrscr() ;

	printf("Enter the size of the magic square : ") ;
	scanf("%d", &n) ;
	if (n % 2 == 0)
	{
		printf("\nMagic square is not possible") ;
		return;
	}

	printf("\nThe magic square for %d x %d is :\n\n", n, n) ;
	j = (n + 1) / 2 ;
	i = 1 ;

	for(c = 1 ; c <= n * n ; c++)
	{
		a[i][j] = c ;
		if(c % n == 0)
		{
			i = (i + 1);
			continue;
		}
		if(i == 1)
			i = n ;
		else
			i = i - 1 ;

		if(j == n)
			j = 1;
		else
			j = j + 1 ;
	}

	for (i = 1 ; i <= n ; i++)
	{
		for (j = 1 ; j <= n ; j++)
			printf("%d\t", a[i][j]) ;
		printf("\n") ;
	}
	getch() ;
}
```

Output:

```
Enter the size of the magic square : 3
The magic square for 3 x 3 is :
8       1       6
3       5       7
4       9       2
```

9. Write a Menu driven C program to

a) Accept two numbers n and m
b) Sum of all integers ranging from n to m
c) Sum of all odd integers ranging from n to m.
d) Sum of all even integers ranging from n to m

Display an error message if n > m. Create functions for each of the options.

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

void read(int *, int *);
int sum_all(int*, int*);
int sum_odd(int*, int*);
int sum_even(int*, int*);


void main()
{
        int n, m, sAll, sOdd,sEven, opt;
        clrscr();

        for(;;)
        {
           printf("\n1. Read numbers \n 2. Sum of all elements \n 3. Sum of odd \n");
           printf("4. Sum of even numbers \n 5. Exit \n");
           printf("\nEnter your option:");
           scanf("%d", &opt);

           switch(opt)
           {
                case 1: read(&n, &m);
                          break;
                case 2: sAll=sum_all(&n, &m);
                        printf("\nSum of all numbers between %d & %d = %d", n, m, sAll);
                        break;
                case 3: sOdd = sum_odd(&n, &m);
```

```
                printf("\nSum of odd numbers between %d & %d = %d", n, m, sOdd);
                    break;
            case 4: sEven = sum_even(&n, &m);
              printf("\nSum of even integers between %d and %d = %d", n, m, sEven);
                    break;
            case 5:
            default:      exit(0);
        }
    }
}

void read(int *a, int *b)
{
      printf("Enter two integers:\n");
      scanf("%d%d", a, b);
      if(*a>*b)
      {
            printf("\nThe value n must be lesser than m");
            getch();
            exit(0);
      }
}

int sum_all(int *n, int *m)
{
      int i, sum=0;
      for(i=*n; i<=*m;i++)
            sum=sum+i;

      return sum;
}

int sum_odd(int *n, int *m)
{
      int i, sum=0;
      for(i=*n; i<=*m;i++)
            if(i%2!=0)
                  sum=sum+i;

      return sum;
}

int sum_even(int *n, int *m)
{
      int i, sum=0;
      for(i=*n; i<=*m;i++)
            if(i%2==0)
```

```
                sum=sum+i;

        return sum;
}
```

Output:

```
1. Read numbers
2. Sum of all elements
3. Sum of odd numbers
4. Sum of even numbers
5. Exit

Enter your option:1
Enter two integers:  5     25

1. Read numbers
2. Sum of all elements
3. Sum of odd numbers
4. Sum of even numbers
5. Exit

Enter your option:2
Sum of all integers between 5 and 25 = 315

1. Read numbers
2. Sum of all elements
3. Sum of odd numbers
4. Sum of even numbers
5. Exit

Enter your option:3
Sum of odd integers between 5 and 25 = 165

1. Read numbers
2. Sum of all elements
3. Sum of odd numbers
4. Sum of even numbers
5. Exit

Enter your option: 4
Sum of even integers between 5 and 25 = 150

1. Read numbers
2. Sum of all elements
3. Sum of odd numbers
4. Sum of even numbers
```

```
5. Exit

Enter your option:5
```

10. Write a Menu Driven C Program to implement the following using recursion.

a) Factorial of a number

b) Fibonacci series.

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
int fact(int n)
{
	if(n==0)
		return 1;
	return n*fact(n-1);
}

int fibo(int n)
{
	if(n==0)
		return 0;
	else if(n==1)
		return 1;
	return fibo(n-1)+fibo(n-2);
}

void main()
{
	int i, n, opt;
	clrscr();

	for(;;)
	{
		printf("\n1.Factorial \n2.Fibonacci \n3.Exit");
		printf("\nEnter your option:");
		scanf("%d",&opt);

		switch(opt)
		{
			case 1: printf("\nEnter n:");
				scanf("%d", &n);
				printf("\n %d! = %d", n, fact(n));
				break;
			case 2: printf("\nEnter n:");
				scanf("%d", &n);
```

```
                        printf("\nFibonacci sequence upto %d:", n);
                        for(i=0;i<=n;i++)
                                printf("\n%d", fibo(i));
                        break;
                case 3:
                default: exit(0);
            }
        }
}
```

Output:

```
1.Factorial
2.Fibonacci
3.Exit

Enter your option:1
Enter n:5
5! = 120

1.Factorial
2.Fibonacci
3.Exit

Enter your option:2
Enter n:10
Fibonacci sequence upto 10:
0     1     1     2     3     5     8     13    21    34    55

1.Factorial
2.Fibonacci
3.Exit

Enter your option:3
```

11. Create a structure Complex Number having real and imaginary part as properties. Write functions to add and subtract the two complex numbers.

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

struct complex
{
        int real, img;
};
```

```
typedef struct complex COMP;

void read(COMP *c)
{
	printf("\nEnter real part:");
	scanf("%d", &c->real);
	printf("\nEnter imaginary part:");
	scanf("%d", &c->img);
}

COMP add(COMP c1, COMP c2)
{
	COMP c3;
	c3.real=c1.real + c2.real;
	c3.img=c1.img + c2.img;
	return c3;
}

COMP sub(COMP c1, COMP c2)
{
	COMP c3;
	c3.real=c1.real - c2.real;
	c3.img=c1.img - c2.img;
	return c3;
}


void main()
{
	COMP c1, c2, c3, c4;
	clrscr();

	printf("\nEnter first complex number:");
	read(&c1);
	printf("\nEnter second complex number:");
	read(&c2);

	c3=add(c1,c2);
	c4=sub(c1,c2);
	printf("\nSum = (%d, %d)", c3.real,c3.img);
	printf("\nDifference = (%d, %d)", c4.real, c4.img);

	getch();
}
```

Output:

```
Enter first complex number:
Enter real part:3
Enter imaginary part:4

Enter second complex number:
Enter real part:2
Enter imaginary part:-5

Sum = (5, -1)
Difference = (1, 9)
```

12. Define a structure called student having the properties of student_id, student name and branch of the student with a sub structure of marks of 3 subjects. Write a Menu Driven C Program to

a) Add new student detail
b) Delete a student detail
c) Display all student details
d) Display the name of the student with the best mark
e) Display the name of the student with the worst mark

Display the average marks scored by the students

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


struct marks
{
	int m1, m2, m3;
};

struct student
{
	int sID;
	char name[25];
	char branch[5];
	struct marks m;
	float avg;
};

void add(struct student s[], int *count)
{
	printf("\nEnter student id:");
	scanf("%d", &s[*count].sID);
	printf("\nEnter student name:");
	scanf("%s", s[*count].name);
```

```
        printf("\nEnter student branch:");
        scanf("%s", s[*count].branch);
        printf("\nEnter marks in 3 subjects");
        scanf("%d%d%d", &s[*count].m.m1, &s[*count].m.m2,  &s[*count].m.m3);
        (*count)++;
}

void del(struct student s[], int *count , int id)
{
        int i, flag=0, key;

        for(i=0;i<*count;i++)
        {
                if(id == s[i].sID)
                {
                        flag=1;
                        key=i;
                        break;
                }
        }

        if(flag==0)
        {
                printf("\nID not found");
                return;
        }
        else
        {
                printf("\nDeleted entry is:\n");
                printf("\%d\t%s\t%s\t%d\t%d\t%d", s[key].sID, s[key].name,
                                s[key].branch, s[key].m.m1, s[key].m.m2, s[key].m.m3);

                for(i=key;i<*count-1;i++)
                        s[i]=s[i+1];

                (*count)--;
        }
}

void Average(struct student s[], int *count)
{
        int i;
        for(i=0;i<*count;i++)
                s[i].avg=(s[i].m.m1 + s[i].m.m2 + s[i].m.m3)/3.0;
}

void best(struct student s[], int *count)
```

```
{
        int i, id;
        float big =s[0].avg;

        Average(s, count);

        for(i=0;i<*count;i++)
                if(big<s[i].avg)
                {
                        big=s[i].avg;
                        id =i;
                }
        printf("\n The topper is:\n");
        printf("\n%d %s  %f", s[id].sID, s[id].name, s[id].avg);
}

void worst(struct student s[], int *count)
{
        int i, id;
        float small =s[0].avg;

        Average(s, count);
        for(i=0;i<*count;i++)
                if(small>s[i].avg)
                {
                        small=s[i].avg;
                        id =i;
                }
        printf("\n The worst scorer is:\n");
        printf("\n%d %s  %f", s[id].sID, s[id].name, s[id].avg);
}

void main()
{
        struct student s[10];
        int i, count=0, opt, id, big, small;
        clrscr();

        for(;;)
        {
                printf("\n******* Structure Operations *********");
                printf("\n1. Add new student \n2. Delete a student \n3. Display");
                printf("\n4. Display topper \n5. Display least scorer ");
                printf("\n6. Display Average \n7. Exit");

                printf("\nEnter your option:");
                scanf("%d", &opt);
```

```
            switch(opt)
            {
                  case 1: add(s, &count);
                          break;
                  case 2: printf("\nEnter Student ID to be deleted:");
                          scanf("%d", &id);
                          del(s, &count, id);
                          break;
                  case 3: printf("\n ID \t NAME \t BRANCH \t M1 \t M2 \t M3");
                          for(i=0;i<count;i++)
                                printf("\n%d\t%s\t%s\t\t%d\t%d\t%d",s[i].sID,
                                        s[i].name, s[i].branch, s[i].m.m1,
                                        s[i].m.m2,s[i].m.m3);
                          break;
                  case 4: best(s, &count);
                          break;
                  case 5: worst(s, &count);
                          break;
                  case 6: Average(s, &count);
                          printf("\nAverage marks:\n");
                          for(i=0;i<count; i++)
                                printf("\n%d  %s  %f", s[i].sID, s[i].name, s[i].avg);
                          break;
                  case 7:
                  default: exit(0);
            }
      }
}
```

13. a) Write a C Program to remove all white spaces and newline characters from a file.

b) **Find whether a given word exists in the file. If it exists display the location of the word**

Program for removing white spaces and newline characters:

NOTE: The student should first create one file viz. *source.txt* in the edit window. Then execute the program. Again he/she should check the output file i.e. *dest.txt* at edit window itself.

```
#include<stdio.h>
#include<conio.h>
void main()
{
      FILE *fs,*fd;
      char ch;
```

```
        clrscr();

        fs=fopen("source.txt","r");
        if(fs==NULL)
        {
                printf("File can not be opened to read");
                exit(0);
        }

        fd=fopen("dest.txt","w");
        if(fd==NULL)
        {
                printf("\nFile can not be opened to write");
                exit(0);
        }

        while(1)
        {
                ch=fgetc(fs);
                if(ch==EOF)
                        break;

                if(ch==' ' || ch=='\n')
                        continue;
                else
                        fputc(ch,fd);
        }

        printf("File copied successfully!!!");
        getch();
}
```

//Search for word in a file

NOTE: The student should first create one file viz. *source.txt* in the edit window. Then execute the program.

```
#include <stdio.h>
#include <stdlib.h>
void main()
{
        FILE *fp;
        int line_num = 1, find_result = 0;
        char temp[512], str[10];
        clrscr();

        printf("Enter a String:");
```

```
        scanf("%s",str);
        fp=fopen("source.txt","r");

        while(fgets(temp, 512, fp) != NULL)
        {
                if((strstr(temp, str)) != NULL)
                {
                        printf("A match found on line: %d\n", line_num);
                        printf("\n%s\n", temp);
                        find_result++;
                }
                line_num++;
        }

        if(find_result == 0)
                        printf("\nSorry, couldn't find a match.\n");

        fclose(fp);
        getch();
}
```

14. Write a C program to copy one file content to another file without using inbuilt functions.

NOTE: The student should first create one file viz. *source.txt* in the edit window. Then execute the program. Again he/she should check the output file i.e. *dest.txt* at edit window itself.

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

void main()
{
        FILE *fs,*fd;
        char ch;
        clrscr();

        fs=fopen("source.txt","r");
        if(fs==NULL)
        {
                printf("File can not be opened to read");
                exit(0);
        }

        fd=fopen("dest.txt","w");
```

```
        if(fd==NULL)
        {
                printf("\nFile can not be opened to write");
                exit(0);
        }

        while(1)
        {
                ch=fgetc(fs);

                if(ch==EOF)
                        break;

                fputc(ch,fd);
        }

        printf("File copied successfully!!!");
        getch();
}
```