FREE SOFTWARE MOVEMENT KARNATAKA

www.fsmk.org

Computer Programming Laboratory

LAB MANUAL

Prabodh C P

Volunteer, FSMK

Contents

COMPUTER PROGRAMMING LABORATORY
Common to all Branches

Sub Code: 14CPL16/14CPL26	**IA Marks : 25**
Hrs/ Week: 03	**Exam Hours : 03**
Total Hrs.: 42	**Exam Marks : 50**

Practical Examination Procedure:

- **Part - A** experiment is a demo experiment only and shall not be included in practical exam.
- **Part B:** All experiments are to be included for practical examination.
- Students are allowed to pick one experiment from the lot.
- Strictly follow the instructions as printed on the answer script for breakup of marks.
- **Change of experiment is allowed only once and 15% Marks should be deducted from the procedure part.**

Part I

PART A

PART A : Demonstration of Personal Computer and its Accessories

Demonstration and Explanation on Disassembly and Assembly of a Personal Computer by the faculty-in-charge. Students have to prepare a write-up on the same and include it in the Lab record and evaluated.

Laboratory Session-1: Write-up on Functional block diagram of Computer, CPU, Buses, Mother Board, Chip sets, Operating System & types of OS, Basics of Networking & Topology and NIC.

Laboratory Session-2: Write-up on RAM, SDRAM, FLASH memory, Hard disks, Optical media, CD-ROM/R/RW, DVDs, Flash drives, Keyboard, Mouse, Printers and Plotters.

Note: These TWO Laboratory sessions are used to fill the gap between theory classes and practical sessions.

Part II

PART B

PART B: Problem Solving in C

Implement the programs with GNU/LINUX platform using appropriate C compiler.

Chapter 1

Quadratic Equation

Design and develop a flowchart or an algorithm that takes three coefficients (a, b and c) of a Quadratic equation ($ax^2 + bx + c = 0$) as input and compute all possible roots. Implement a C program for the developed flowchart/algorithm and execute the same to output the possible roots for a given set of coefficients with appropriate messages.

C Code

```
/*****************************************************************************
*File         : 01Quadratic.c
*Description  : Program to find the roots of a Quadratic Equation
*Author       : Prabodh C P
*Compiler     : gcc 4.6.3 compiler, Ubuntu 14.04
*Date         : 26 June 2014
*****************************************************************************/

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

/*****************************************************************************
*Function         :        main
*Input parameters :        no parameters
*RETURNS          :        0 on success
*****************************************************************************/

int main(void)
{
        float fA,fB,fC,fDesc,fX1,fX2,fRealp,fImagp;

        printf("\n*****************************************************************");
        printf("\n*\tPROGRAM TO FIND ROOTS OF A QUADRATIC EQUATION\t    *\n");
        printf("*****************************************************************");


        printf("\nEnter the coefficients of a,b,c \n");
        scanf("%f%f%f",&fA,&fB,&fC);
        if(0 == fA)
        {
                printf("\nInvalid input, not a quadratic equation - try again\n");
                exit(0);
        }

        /*COMPUTE THE DESCRIMINANT*/
        fDesc=fB*fB-4*fA*fC;
```

```
        if(0 == fDesc)
        {
                fX1 = fX2 = -fB/(2*fA);

                printf("\nRoots are equal and the Roots are \n");
                printf("\nRoot1 = %g and Root2 = %g\n",fX1,fX2);
        }
        else if(fDesc > 0)
        {
                fX1 = (-fB+sqrt(fDesc))/(2*fA);
                fX2 = (-fB-sqrt(fDesc))/(2*fA);
                printf("\nThe Roots are Real and distinct, they are \n");
                printf("\nRoot1 = %g and Root2 = %g\n",fX1,fX2);
        }
        else
        {
                fRealp = -fB / (2*fA);
                fImagp = sqrt(fabs(fDesc))/(2*fA);
                printf("\nThe Roots are imaginary and they are\n");
                printf("\nRoot1 = %g+i%g\n",fRealp,fImagp);
                printf("\nRoot2 = %g-i%g\n",fRealp,fImagp);
        }

        return 0;
}
```

Output

Run the following commands in your terminal:
$ gcc 01Quadratic.c -lm
$./a.out

```
*************************************************************
*       PROGRAM TO FIND ROOTS OF A QUADRATIC EQUATION       *
*************************************************************
Enter the coefficients of a,b,c
0 1 2

Invalid input, not a quadratic equation - try again
```

$./a.out

```
*************************************************************
*       PROGRAM TO FIND ROOTS OF A QUADRATIC EQUATION       *
*************************************************************
Enter the coefficients of a,b,c
1 -5 6

The Roots are Real and distinct, they are

Root1 = 3 and Root2 = 2
```

$./a.out

```
*************************************************************
*       PROGRAM TO FIND ROOTS OF A QUADRATIC EQUATION       *
*************************************************************
Enter the coefficients of a,b,c
1 4 4

Roots are equal and the Roots are

Root1 = -2 and Root2 = -2
```

$./a.out

```
*************************************************************
*       PROGRAM TO FIND ROOTS OF A QUADRATIC EQUATION       *
*************************************************************
Enter the coefficients of a,b,c
1 3 3

The Roots are imaginary and they are

Root1 = -1.5+i0.866025
Root2 = -1.5-i0.866025
```

Chapter 2

Palindrome Check

Design and develop an algorithm to find the reverse of an integer number NUM and check whether it is PALINDROME or NOT. Implement a C program for the developed algorithm that takes an integer number as input and output the reverse of the same with suitable messages. Ex: Num: 2014, Reverse: 4102, Not a Palindrome

C Code

```
/*****************************************************************************
*File         : 02Palindrome.c
*Description  : Program to check whether the given integer is a Palindrome or not
*Author       : Prabodh C P
*Compiler     : gcc 4.6.3 compiler, Ubuntu 14.04
*Date         : 26 June 2014
*****************************************************************************/

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

/*****************************************************************************
*Function         :        main
*Input parameters :        no parameters
*RETURNS          :        0 on success
*****************************************************************************/

int main(void)
{
        int iNum,iRev = 0,iTemp,iDig;

        printf("\n*****************************************************************************");
        printf("\n*\tPROGRAM TO CHECK WHETHER AN INTEGER IS A PALINDROME OR NOT\t *\n");
        printf("*****************************************************************************");

        printf("\nEnter a number\n");
        scanf("%d",&iNum);

        iTemp = iNum;

        while(iNum!=0)
        {
                iDig = iNum % 10;
                iRev = iRev * 10 + iDig;
                iNum = iNum/10;
        }
        printf("\nReversed number is %d",iRev);
```

```
        if(iRev == iTemp)
                printf("\nNumber %d is a palindrome\n",iTemp);
        else
                printf("\nNumber %d is not a palindrome\n",iTemp);

        return 0;
}
```

Output

```
$ gcc 02Palindrome.c
$ ./a.out

*********************************************************************
        PROGRAM TO CHECK WHETHER AN INTEGER IS A PALINDROME OR NOT
*********************************************************************
Enter a number
2014

Reversed number is 4102
Number 2014 is not a palindrome

$ ./a.out

*********************************************************************
        PROGRAM TO CHECK WHETHER AN INTEGER IS A PALINDROME OR NOT
*********************************************************************
Enter a number
2002

Reversed number is 2002
Number 2002 is a palindrome
```

Chapter 3

3.1 Calculate Square Root

Design and develop a flowchart to find the square root of a given number N. Implement a C program for the same and execute for all possible inputs with appropriate messages. ***Note: Do not use library function sqrt(n).***

C Code

```
/*****************************************************************************
*File         : 03aSquareRoot.c
*Description  : Program to find the square root of a given number
*Author       : Prabodh C P
*Compiler     : gcc 4.6.3 compiler, Ubuntu 14.04
*Date         : 26 June 2014
*****************************************************************************/
#include <stdio.h>
#include <math.h>
#include <stdlib.h>

/*****************************************************************************
*Function         :        main
*Input parameters :        no parameters
*RETURNS          :        0 on success
*****************************************************************************/

int main()
{
    float fVal, fNextGuess, fLastGuess = 1.0f, fDiff ;
    printf("\nEnter a value whose square root has to be calculated\n");
    scanf("%f", &fVal);
    do
    {
        fNextGuess = 0.5 * (fLastGuess + (fVal/fLastGuess));
        fDiff = fabs(fNextGuess - fLastGuess);
        fLastGuess = fNextGuess;
    }while (fDiff > 0.0001);

    printf("\nSquare root of %g = %g\n", fVal, fNextGuess);
    return 0;
}
```

Output

```
$ gcc 03aSquareRoot.c

$ ./a.out

Enter a value whose square root has to be calculated
8.7
```

```
Square root of 8.7 = 2.94958

$ ./a.out

Enter a value whose square root has to be calculated
9

Square root of 9 = 3

$ ./a.out

Enter a value whose square root has to be calculated
4.3

Square root of 4.3 = 2.07364
```

3.2 Leap Year Check

Design and develop a C program to read a year as an input and find whether it is leap year or not. Also consider end of the centuries.

C Code

```
/*****************************************************************************
*File         : 03bCheckLeapYear.c
*Description  : Program to check whether a given year is leap year or not
*Author       : Prabodh C P
*Compiler     : gcc 4.6.3 compiler, Ubuntu 14.04
*Date         : 26 June 2014
*****************************************************************************/

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

/*****************************************************************************
*Function         :        main
*Input parameters :        no parameters
*RETURNS          :        0 on success
*****************************************************************************/

int main()
{
    int iYear;

    printf("\n Enter a year\n");
    scanf("%d", &iYear);

    if(iYear % 4 == 0 && iYear % 100 != 0)
        printf("\n%d is a leap year\n",iYear);
    else if(iYear % 400 == 0)
        printf("\n%d is a leap year\n",iYear);
    else
        printf("\n%d is not a leap year\n",iYear);
    return 0;
}
```

Output

```
$ gcc 03bCheckLeapYear.c
$ ./a.out

 Enter a year        2014

2014 is not a leap year

$ ./a.out

 Enter a year        1988

1988 is a leap year

$ ./a.out

 Enter a year        2000

2000 is a leap year

$ ./a.out

 Enter a year        2100

2100 is not a leap year
```

Chapter 4

Horner's Method

Design and develop an algorithm for evaluating the polynomial $f(x) = a_4x^4+a_3x^3+a_2x^2+a_1x+a_0$ for a given value of x and its coefficients using Horners method. Implement a C program for the developed algorithm and execute for different sets of values of coefficients and x.

C Code

```
/*****************************************************************************
*File         : 04Horner.c
*Description  : Program to implement Horners method for Polynomial evaluation
*Author       : Prabodh C P
*Compiler     : gcc 4.6.3 compiler, Ubuntu 14.04
*Date         : 26 June 2014
*****************************************************************************/

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

/*****************************************************************************
*Function         :        main
*Input parameters :        no parameters
*RETURNS          :        0 on success
*****************************************************************************/

int main(void)
{

        int iDeg,i,iaCoeff[10];
        float fX,fSum=0;

        iDeg = 4;

        printf("\n*****************************************************************************");
        printf("\n*\tPROGRAM TO EVALUATE A POLYNOMIAL USING HORNERS METHOD\t    *\n");
        printf("*****************************************************************************");

        printf("\nEnter the value of x\n");
        scanf("%f",&fX);

        printf("\nEnter the coefficients \n");
        for(i=0;i<=iDeg;i++)
        {
                scanf("%d",&iaCoeff[i]);
        }
```

```
        for(i=0;i<iDeg;++i)
        {
                fSum = (fSum + iaCoeff[i])*fX;
        }

        fSum = fSum + iaCoeff[iDeg];


        printf("\nValue of polynomial after evaluation=%f\n",fSum);

        return 0;
}
```

Output

```
$ gcc 04Horner.c
$ ./a.out

*************************************************************************
*        PROGRAM TO EVALUATE A POLYNOMIAL USING HORNERS METHOD          *
*************************************************************************
Enter the degree of the polynomial and value of x
4
2

Enter the coefficients
5 4 3 2 1

Value of polynomial after evaluation=129.000000

$ ./a.out

*************************************************************************
*        PROGRAM TO EVALUATE A POLYNOMIAL USING HORNERS METHOD          *
*************************************************************************
Enter the degree of the polynomial and value of x
3
3

Enter the coefficients
1 2 3 4

Value of polynomial after evaluation=58.000000
```

Chapter 5

Taylor's Series

Write C Program to compute Sin(x) using Taylor series approximation given by $sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} \ldots\ldots\ldots$ Compare the result with the built- in Library function and print both the results.

C Code

```
/*****************************************************************************
*File         : 05SineAngle.c
*Description  : Program to calculate Sin(x) using Taylor series
*Author       : Prabodh C P
*Compiler     : gcc 4.6.3 compiler, Ubuntu 14.04
*Date         : 26 June 2014
*****************************************************************************/

#include <stdio.h>
#include <math.h>

#define PI 3.1416

/*****************************************************************************
*Function         :        main
*Input parameters :        no parameters
*RETURNS          :        0 on success
*****************************************************************************/

int main()
{
        float fAngD, fAngR;
        float fTerm, fNum, fDen, fVal;
        int i,iNum;

        printf("Enter the Angle ....\n");
        scanf("%f",&fAngD);

        printf("Angle = %f\n",fAngD);

        printf("Enter the Number of terms...\n");
        scanf("%d",&iNum);
        printf("No of terms = %d\n",iNum);

        fAngR= (fAngD*PI)/180 ;

        fNum=fAngR;
        fDen=1.0;
        fVal =0.0;
        fTerm=fNum/fDen;
        for(i=1;i<=iNum;i++)
```

```
        {
                fVal = fVal + fTerm;
                fNum = -fNum * fAngR * fAngR ;
                fDen = fDen * (2*i) * (2*i+1);
                fTerm = fNum/fDen;
        }
        printf(" Calculated value is :Sin( %f ) = %f\n",fAngD,fVal);
        printf("Built In function value is :Sin( %f ) = %f\n",
                                        fAngD, sin(fAngR));
        return 0;
}
```

Output

```
$ gcc 05SineAngle.c -lm

$ ./a.out
Enter the Angle ....
60
Angle = 60.000000
Enter the Number of terms...
10
No of terms = 10
 Calculated value is :Sin( 60.000000 ) = 0.866027
Built In function value is :Sin( 60.000000 ) = 0.866027

$ ./a.out
Enter the Angle ....
30
Angle = 30.000000
Enter the Number of terms...
8
No of terms = 8
 Calculated value is :Sin( 30.000000 ) = 0.500001
Built In function value is :Sin( 30.000000 ) = 0.500001

$ ./a.out
Enter the Angle ....
30
Angle = 30.000000
Enter the Number of terms...
3
No of terms = 3
 Calculated value is :Sin( 30.000000 ) = 0.500003
Built In function value is :Sin( 30.000000 ) = 0.500001

$ ./a.out
Enter the Angle ....
30
Angle = 30.000000
Enter the Number of terms...
2
No of terms = 2
 Calculated value is :Sin( 30.000000 ) = 0.499675
Built In function value is :Sin( 30.000000 ) = 0.500001
```

Chapter 6

Bubble Sort & Binary Search

Develop, implement and execute a C program that reads N integer numbers and arrange them in ascending order using *Bubble Sort* technique. Extend the program to perform a search operation on these sorted numbers by accepting a key element from the user applying *Binary Search* method. Report the result SUCCESS or FAILURE as the case may be.

C Code

```
/*****************************************************************************
*File         : 06BubbleBinary.c
*Description  : Program to implement Bubble Sort and Binary Search
*Author       : Prabodh C P
*Compiler     : gcc 4.6.3 compiler, Ubuntu 14.04
*Date         : 26 June 2014
*****************************************************************************/

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

/*****************************************************************************
*Function         :        main
*Input parameters :        no parameters
*RETURNS          :        0 on success
*****************************************************************************/

int main(void)
{
        int iNum, i, j, k, iaArr[10], iTemp, iKey, iPos, iLow, iHigh, iMid, iFound;

        printf("\nEnter no of elements\n");
        scanf("%d",&iNum);

        printf("\nEnter the elements\n");
        for(i=0;i<iNum;i++)
                scanf("%d",&iaArr[i]);

        for(i=0;i<iNum-1;i++)
        {
                for(j=i+1;j<iNum;j++)
                {
                        if(iaArr[i] > iaArr[j])
                        {
                                iTemp = iaArr[i];
                                iaArr[i] = iaArr[j];
                                iaArr[j] = iTemp;
                        }
```

```
                }
        }

        printf("\nThe Sorted array is \n");

        for(i=0;i<iNum;i++)
                printf("%d\t",iaArr[i]);

        printf("\n");
        printf("\nEnter the Key element\n");
        scanf("%d",&iKey);

        iFound = 0;
        iLow = 0;
        iHigh = iNum-1;

        while(iLow <= iHigh)
        {
                iMid = (iLow + iHigh)/2;

                if(iKey == iaArr[iMid])        /*KEY ELEMENT FOUND*/
                {
                        iPos = iMid;
                        iFound = 1;
                        break;
                }
                else if(iKey < iaArr[iMid])        /*KEY ELEMENT IS IN 1ST HALF*/
                        iHigh = iMid - 1;
                else        /*KEY ELEMENT IS IN 2ND HALF*/
                        iLow = iMid +1;
        }


        if(iFound)
                printf("\nKey element %d found at position %d\n",iKey,iPos+1);
        else
                printf("\nKey element not found\n");

        return 0;
}
```

Output

```
$ gcc 06BubbleBinary.c

$ ./a.out

Enter no of elements
6

Enter the elements
1 3 5 2 4 6

The Sorted array is
1        2        3        4        5        6

Enter the Key element
4

Key element 4 found at position 4

$ ./a.out

Enter no of elements
7

Enter the elements
1 7 6 2 9 4 8

The Sorted array is
1        2        4        6        7        8        9

Enter the Key element
3

Key element not found
```

Chapter 7

Matrix Multiplication

Develop, implement and execute a C program that reads two matrices *A (m x n)* and *B (p x q)* and Compute the product A and B. Read matrix A in row major order and matrix B in column major order. Print both the input matrices and resultant matrix with suitable headings and in matrix format. Program must check the compatibility of orders of the matrices for multiplication. Report appropriate message in case of incompatibility.

C Code

```
/*****************************************************************************
*File         : 07MatrixMultiplication.c
*Description  : Program to implement Matrix Multiplication
*Author       : Prabodh C P
*Compiler     : gcc 4.6.3 compiler, Ubuntu 14.04
*Date         : 26 June 2014
*****************************************************************************/

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

/*****************************************************************************
*Function         :        main
*Input parameters :        no parameters
*RETURNS          :        0 on success
*****************************************************************************/

int main(void)
{
        int iM, iN, iP, iQ, i, j, k, iaMat1[10][10], iaMat2[10][10];
        int iaProd[10][10] = {0};   //Initialize the matrix with all zeros


        printf("\n************************************************************");
        printf("\n*\tPROGRAM TO IMPLEMENT MATRIX MULIPLICATION\t*\n");
        printf("************************************************************");

        printf("\nEnter the order of Matrix1\n");
        scanf("%d%d",&iM,&iN);

        printf("\nEnter the order of Matrix2\n");
        scanf("%d%d",&iP,&iQ);

        if( iN != iP)
        {
                printf("\nMatrix Multiplication not possible\n");
                exit(0);
        }
```

```
        printf("\nEnter the elements of Matrix 1 in row major order\n");
        for(i=0;i<iM;i++)
                for(j=0;j<iN;j++)
                        scanf("%d",&iaMat1[i][j]);

        printf("\nEnter the elements of Matrix 2 in column major order\n");
        for(i=0;i<iQ;i++)
                for(j=0;j<iP;j++)
                        scanf("%d",&iaMat2[j][i]);


        for(i=0;i<iM;i++)
        {
                for(j=0;j<iQ;j++)
                {
                        for(k=0;k<iN;k++)
                        {
                        iaProd[i][j] += iaMat1[i][k] * iaMat2[k][j];
                        }
                }
        }

/*****************************************************************************
                |*|                     |*|
a00  a01  a02   |*|    b00  b01  b02    |*|
                |*|                     |*|
a10  a11  a12   |*|    b10  b11  b12    |*|
                |*|                     |*|
a20  a21  a22   |*|    b20  b21  b22    |*|
                |*|                     |*|

(a00*b00+a01*b10+a02*b20) (a00*b01+a01*b11+a02*b21) (a00*b02+a01*b12+a02*b22)
(a10*b00+a11*b10+a12*b20) (a10*b01+a11*b11+a12*b21) (a10*b02+a11*b12+a12*b22)
(a20*b00+a21*b10+a22*b20) (a20*b01+a21*b11+a22*b21) (a20*b02+a21*b12+a22*b22)
*****************************************************************************/


        printf("\nMatrix 1\n");
        for(i=0;i<iM;i++)
        {
                for(j=0;j<iN;j++)
                {
                        printf("%d\t",iaMat1[i][j]);
                }
                printf("\n");
        }
        printf("\n");

        printf("\nMatrix 2\n");
        for(i=0;i<iP;i++)
        {
                for(j=0;j<iQ;j++)
                {
                        printf("%d\t",iaMat2[i][j]);
                }
                printf("\n");
        }
        printf("\n");

        printf("\nThe Product matrix is is \n");
```

```
        for(i=0;i<iM;i++)
        {
                for(j=0;j<iQ;j++)
                {
                        printf("%d\t",iaProd[i][j]);
                }
                printf("\n");
        }
        printf("\n");
        return 0;
}
```

Output

```
$ gcc 07MatrixMultiplication.c

$ ./a.out

*********************************************************
*        PROGRAM TO IMPLEMENT MATRIX MULIPLICATION        *
*********************************************************
Enter the order of Matrix1
2 2

Enter the order of Matrix2
3 2

Matrix Multiplication not possible

$ ./a.out

*********************************************************
*        PROGRAM TO IMPLEMENT MATRIX MULIPLICATION        *
*********************************************************
Enter the order of Matrix1
2 2

Enter the order of Matrix2
2 2

Enter the elements of Matrix 1 in row major order
1 2 3 4

Enter the elements of Matrix 2 in column major order
1 2 3 4

Matrix 1
1        2
3        4


Matrix 2
1        3
2        4


The Product matrix is is
5        11
11        25
```

Chapter 8

String Operations

Write and execute a C program that

1. Implements string copy operation *STRCOPY(str1,str2)* that copies a string str1 to another string str2 without using library function.
2. Reads a sentence and prints frequency of each of the vowels and total count of consonants.

C Code

```
/*****************************************************************************
*File         : 08String.c
*Description  : Program to perform operations on Strings
*Author       : Prabodh C P
*Compiler     : gcc compiler, Ubuntu 14.04
*Date         : 26 June 2014
*****************************************************************************/

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

void STRCOPY(char*, char*);

/*****************************************************************************
*Function         :        main
*Input parameters :        no parameters
*RETURNS          :        0 on success
*****************************************************************************/

int main()
{
    char acStr1[30],acStr2[30],acStr3[30], cChar;
    int i, iFreqA = 0, iFreqE = 0, iFreqI = 0, iFreqO = 0, iFreqU = 0, iFreqCons = 0;

    printf("\nEnter string to be copied\n");
    gets(acStr1);

    STRCOPY(acStr1,acStr2);

    printf("\nCopied String is\n");
    puts(acStr2);

    printf("\nEnter string to find letter frequency\n");
    gets(acStr3);

    for(i=0;acStr3[i]!='\0';i++)
    {
        cChar = tolower(acStr3[i]);
```

```
        switch(cChar)
        {
            case 'a' : iFreqA++;
                    break;
            case 'e' : iFreqE++;
                    break;
            case 'i' : iFreqI++;
                    break;
            case 'o' : iFreqO++;
                    break;
            case 'u' : iFreqU++;
                    break;
            case ' ' : break;
            default : iFreqCons++;
        }
    }
    printf("\nFrequency of vowel 'A' is : %d\n",iFreqA);
    printf("\nFrequency of vowel 'E' is : %d\n",iFreqE);
    printf("\nFrequency of vowel 'I' is : %d\n",iFreqI);
    printf("\nFrequency of vowel 'O' is : %d\n",iFreqO);
    printf("\nFrequency of vowel 'U' is : %d\n",iFreqU);
    printf("\nThe no of consonants is : %d\n",iFreqCons);


    return 0;
}

void STRCOPY(char *s1, char *s2)
{
    int i;
    for(i=0;s1[i]!='\0';i++)
        s2[i] = s1[i];
    s2[i] = '\0';
}
```

Output

```
$ gcc 08String.c
$ ./a.out

Enter string to be copied
Free Software is the future

Copied String is
Free Software is the future

Enter string to find letter frequency
The Future is ours

Frequency of vowel 'A' is : 0

Frequency of vowel 'E' is : 2

Frequency of vowel 'I' is : 1

Frequency of vowel 'O' is : 1

Frequency of vowel 'U' is : 3

The no of consonants is : 8
```

Chapter 9

9.1 Right Shift

Design and develop a C function *RightShift(x ,n)* that takes two integers x and n as input and returns value of the integer x rotated to the right by n positions. Assume the integers are unsigned. Write a C program that invokes this function with different values for x and n and tabulate the results with suitable headings

C Code

```
/*********************************************************************
*File         : 09ARightRotate.c
*Description  : Program to perform right rotate on a integer by n positions
*Author       : Prabodh C P
*Compiler     : gcc 4.6.3 compiler, Ubuntu 14.04
*Date         : 26 June 2014
*********************************************************************/

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

int fnRightShift(unsigned int , unsigned int);

/*********************************************************************
*Function         :        main
*Input parameters :        no parameters
*RETURNS          :        0 on success
*********************************************************************/

int main(void)
{
        unsigned int iVal, iNewVal, iNum;
        int iChoice;

        printf("\n*************************************************************");
        printf("\n*\tPROGRAM TO IMPLEMENT CIRCULAR RIGHT SHIFT\t*\n");
        printf("*************************************************************");


        do
        {
                printf("\nEnter the value to be rotated\n");
                scanf("%u",&iVal);

                printf("\nEnter the number of positions by which the value has to be rotated\n");
                scanf("%u",&iNum);

                iNewVal = fnRightShift(iVal, iNum);

                printf("\nThe value %u after right rotation by %u bits = %u\n",iVal,iNum,iNewVal);
```

```
                printf("\nPress 1 to continue or any other key to exit....\n");
                scanf("%d",&iChoice);
        }while(1 == iChoice);

        return 0;
}

/*****************************************************************************
*Function                : fnRightRot
*Description                : Function to perform Right Circular shift by the number of bits specified
*Input parameters        :
*        unsigned int iX - value to be rotated
*        int iN                - no of positions by which rotation has to be performed
*RETURNS                : resultant value after rotation has been performed
*****************************************************************************/


int fnRightShift(unsigned int iX , unsigned int iN)
{
        unsigned int iShift;
        int i;
        iShift = 1 << 31;



        for(i = 0; i< iN ;i++)
        {
                /*IF iX HAS AN 1 AT THE RIGHTMOST BIT IT HAS TO APPEAR AT THE LEFT END AFTER ROTATION*/

                if( iX % 2 )
                {
                        iX = iX >> 1;
                        iX |= iShift;
                }
                else        /*OTHERWISE SIMPLY APPLY RIGHT SHIFT*/
                {
                        iX = iX >> 1;
                }
        }
        return iX;
}
```

Output

```
$ gcc 09ARightRotate.c
$ ./a.out

*********************************************************
*        PROGRAM TO IMPLEMENT CIRCULAR RIGHT SHIFT        *
*********************************************************
Enter the value to be rotated
1234

Enter the number of positions by which the value has to be rotated
2

The value 1234 after right rotation by 2 bits = 2147483956

Press 1 to continue or any other key to exit....
1

Enter the value to be rotated
1234

Enter the number of positions by which the value has to be rotated
1

The value 1234 after right rotation by 1 bits = 617

Press 1 to continue or any other key to exit....
1

Enter the value to be rotated
5

Enter the number of positions by which the value has to be rotated
1

The value 5 after right rotation by 1 bits = 2147483650

Press 1 to continue or any other key to exit....
2
```

9.2 Primality Check

Design and develop a C function *isprime(num)* that accepts an integer argument and returns 1 if the argument is prime, or 0 otherwise. Write a C program that invokes this function to generate prime numbers between the given range.

C Code

```
/*****************************************************************************
*File         : 09BisPrimefn.c
*Description  : Program to check for primality
*Author       : Prabodh C P
*Compiler     : gcc 4.6.3 compiler, Ubuntu 14.04
*Date         : 26 June 2014
*****************************************************************************/

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

int fnCheckPrime(int);

/*****************************************************************************
*Function         :        main
*Input parameters :        no parameters
*RETURNS          :        0 on success
*****************************************************************************/

int main(void)
{
        int iVal1, iVal2, i;

        printf("\nEnter the range\n");
        scanf("%d%d",&iVal1, &iVal2);

        printf("\nPrime numbers in the range %d to %d is\n", iVal1, iVal2);
        for(i=iVal1;i<=iVal2;i++)
        {
        if(fnCheckPrime(i))        //arguments
            printf("%d\t", i);
        }

        return 0;
}

int fnCheckPrime(int iX)                //parameters
{
        int i,isPrime = 1;
        for(i=2;i<iX;i++)
        {
                if(iX%i == 0)
                {
                        isPrime = 0;
                        break;
                }
        }

        if(isPrime)
        return 1;
        else
        return 0;
}
```

Output

```
$ gcc 09BisPrimefn.c
$ ./a.out

Enter the range
40 80

Prime numbers in the range 40 to 80 is
41        43        47        53        59        61        67        71        73        79

$ ./a.out

Enter the range
20 45

Prime numbers in the range 20 to 45 is
23        29        31        37        41        43
```

Chapter 10

String Matching

Develop a function in C called MatchAny(s1,s2) that takes two string arguments and does the following task:

1. if s1 is substring of s2, Returns 1 along with the first location in the string s2
2. if s1 is equal to s2 , returns 0
3. otherwise, returns -1.

Write a C program that invokes *MatchAny(s1,s2)* for different input strings and output both the strings s1 & s2 with the return value in tabular form. Note: Do not use the standard library functions.

C Code

```
#include <stdio.h>
#include <stdlib.h>
int fnMatchAny(char*, char*);

int main()
{
    char acStr1[50], acStr2[50];
    int iRes;

    printf("\nEnter two strings\n");
    gets(acStr1);
    gets(acStr2);

    iRes = fnMatchAny(acStr1, acStr2);

    if(iRes == 0)
        printf("\nStrings are same\n");
    else if(iRes == -1)
        printf("\nString1 is not a substring of String2\n");
    else
        printf("\nString1 is a substring of String2 and match occured at position %d\n",iRes);

    return 0;
}

int fnMatchAny(char *s1, char *s2)
{
    int iL1, iL2, i, j, iCnt;
    for(iL1=0;s1[iL1]!='\0';iL1++);
    for(iL2=0;s2[iL2]!='\0';iL2++);

    if(iL1 > iL2)
        return -1;
    else if(iL1 == iL2)
    {
```

```
        iCnt = 0;
        for(i=0;i<iL1;i++)
        {
            if(s1[i]==s2[i])
                iCnt++;
        }
        if(iCnt == iL1)
            return 0;
        else
            return -1;
    }
    else
    {
            for(i=0;i<iL2-iL1;i++)
            {
                iCnt = 0;
                for(j=0;j<iL1;j++)
                {
                    if(s1[j]==s2[i+j])
                        iCnt++;
                }
                if(iCnt == iL1)
                    return i+1;
            }
            return -1;
    }
}
```

Output

```
$ gcc 10MatchAny.c

$ ./a.out

Enter two strings
Mother
GrandMothers

String1 is a substring of String2 and match occured at position 6

$ ./a.out

Enter two strings
Father
Mother

String1 is not a substring of String2

$ ./a.out

Enter two strings
Brother
Brother

Strings are same
```

Chapter 11

NcR

Draw the flowchart and write a recursive C function to find the factorial of a number, n!, defined by *fact(n)=1*, if *n=0*. Otherwise *fact(n)=n*fact(n-1)*. Using this function, write a C program to compute the binomial coefficient nCr. Tabulate the results for different values of n and r with suitable messages.

C Code

```
/*****************************************************************************
*File         : 11NcR.c
*Description  : Program to calculate nCr
*Author       : Prabodh C P
*Compiler     : gcc 4.6.3 compiler, Ubuntu 14.04
*Date         : 26 June 2014
*****************************************************************************/

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

int fnFactorial(int);

/*****************************************************************************
*Function         :        main
*Input parameters :        no parameters
*RETURNS          :        0 on success
*****************************************************************************/

int main()
{
    int iN,iR,iNCR;
    printf("\nEnter the value of n\n");
    scanf("%d", &iN);
    printf("\nEnter the value of r\n");
    scanf("%d", &iR);

    if(iN < iR)
    {
        printf("\nnCr does not exist\n");
        exit(0);
    }

    iNCR = (fnFactorial(iN)/(fnFactorial(iR)*fnFactorial(iN-iR)));

    printf("\nFor n = %d and r = %d, %dC%d = %d\n", iN ,iR, iN, iR, iNCR);

    return 0;
}
```

```
int fnFactorial(int iVal)
{
    if(0 == iVal)
        return 1;
    else
        return (iVal * fnFactorial(iVal - 1));
}
```

Output

```
$ gcc 11NcR.c

$ ./a.out

Enter the value of n
2

Enter the value of r
7

nCr does not exist

$ ./a.out

Enter the value of n
7

Enter the value of r
5

For n = 7 and r = 5, 7C5 = 21

$ ./a.out

Enter the value of n
3

Enter the value of r
0

For n = 3 and r = 0, 3C0 = 1

$ ./a.out

Enter the value of n
6

Enter the value of r
6

For n = 6 and r = 6, 6C6 = 1
```

Chapter 12

File Management

Given two text documentary files ”Ramayana.in” and ”Mahabharatha.in”. Write a C program to create a new file ”Karnataka.in” that appends the content of file ”Ramayana.in” to the file ”Mahabharatha.in”. Display the contents of output file ”Karnataka.in” on to screen. Also find number of words and newlines in the output file.

C Code

```
/*****************************************************************************
*File         : 12FileAppend.c
*Description  : Program to append two files
*Author       : Prabodh C P
*Compiler     : gcc 4.6.3 compiler, Ubuntu 14.04
*Date         : 26 June 2014
*****************************************************************************/

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

/*****************************************************************************
*Function         :        main
*Input parameters :        no parameters
*RETURNS          :        0 on success
*****************************************************************************/

int main()
{
    FILE *fp1,*fp2, *fp3;
    int iWordCount = 0, iLineCount = 0;
    char cCharacter;

    fp1 = fopen("Ramayana.in","r");
    fp2 = fopen("Mahabharatha.in", "r");
    fp3 = fopen("Karnataka.in", "w+");

    while((cCharacter=fgetc(fp1))!=EOF)
    {
        fputc(cCharacter,fp3);
    }

    while((cCharacter=fgetc(fp2))!=EOF)
    {
        fputc(cCharacter,fp3);
    }

    rewind(fp3);
```

```
    while((cCharacter=fgetc(fp3))!=EOF)
    {
        if(cCharacter == '\n')
        {
            iLineCount++;
            iWordCount++;
        }
        else if(cCharacter == '\t' || cCharacter == ' ')
            iWordCount++;
       printf("%c",cCharacter);
    }
    fclose(fp1);
    fclose(fp2);
    fclose(fp3);

    printf("\n\nNumber of lines is : %d\n", iLineCount);

    printf("\n\nNumber of words is : %d\n", iWordCount);

    //Code for Verification (works only on GNU\Linux)
    printf("\nOutput of command : wc Karnataka.in -lw\n");
    system("wc Karnataka.in -lw");
    return 0;
}
```

Output

```
$./a.out
This file contains Ramayana.
It was written by Maharshi Valmiki.
This file contains Mahabharatha. It was written by Ved Vyas a fisherman who went on to become a saint.


Number of lines is : 3


Number of words is : 29

Output of command : wc Karnataka.in -lw
  3  29 Karnataka.in
```

Chapter 13

Student Records

Write a C program to maintain a record of "n" student details using an array of structures with four fields (Roll number, Name, Marks, and Grade). Each field is of an appropriate data type. Print the marks of the student given student name as input.

C Code

```
/******************************************************************************
*File         : 13StudMarks.c
*Description  : Program to maintain student records
*Author       : Prabodh C P
*Compiler     : gcc 4.6.3 compiler, Ubuntu 14.04
*Date         : 26 June 2014
******************************************************************************/

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

typedef struct
{
    int iRollNo;
    char acName[30];
    int iMarks;
    char cGrade;
}STUDENT;

/******************************************************************************
*Function         :        main
*Input parameters :        no parameters
*RETURNS          :        0 on success
******************************************************************************/

int main()
{
    STUDENT s[20];
    int i, iNum, iFound = 0;
    char acStudName[30];

    printf("\nEnter the number of students\n");
    scanf("%d", &iNum);
    getchar();

    printf("\nEnter Student Details\n");
    for(i=0;i<iNum;i++)
    {
        printf("\nStudent %d\nRollNo : ", i+1);
        scanf("%d", &s[i].iRollNo);
```

```
        getchar();
        printf("Name   : ");
        gets(s[i].acName);
        printf("Marks  : ");
        scanf("%d", &s[i].iMarks);
        getchar();
        printf("Grade  : ");
        scanf("%c", &s[i].cGrade);
    }

    getchar();
    printf("\nEnter the name to display marks\n");
    gets(acStudName);

    for(i=0;i<iNum;i++)
    {
        if(strcmp(acStudName, s[i].acName)==0)
        {
            iFound = 1;
            printf("\nMarks obtained by %s is %d\n", s[i].acName, s[i].iMarks);
            exit(0);
        }
    }
    if(iFound == 0)
        printf("\nNo student by name %s found\n", acStudName);
    return 0;
}
```

Output

```
$ gcc 13StudMarks.c
$ ./a.out

Enter the number of students
3
Enter Student Details

Student 1
RollNo : 123
Name   : Raj
Marks  : 96
Grade  : S

Student 2
RollNo : 124
Name   : Sam
Marks  : 76
Grade  : A

Student 3
RollNo : 125
Name   : Ali
Marks  : 56
Grade  : B

Enter the name to display marks
Ali

Marks obtained by Ali is 56

$ ./a.out

Enter the number of students
1
Enter Student Details

Student 1
RollNo : 126
Name   : Ram
Marks  : 55
Grade  : B

Enter the name to display marks
Bob

No student by name Bob found
```

Chapter 14

Pointers and Arrays

14.1 Sum of elements using pointers

Write a C program using pointers to compute the sum of all elements stored in an array A[n]. Where n is the number of elements.

C Code

```
/*****************************************************************************
*File         : 14aPointerArrSum.c
*Description  : Program to calculate sum of elements using pointers
*Author       : Prabodh C P
*Compiler     : gcc 4.6.3 compiler, Ubuntu 14.04
*Date         : 26 June 2014
*****************************************************************************/
#include<stdio.h>

/*****************************************************************************
*Function         :        main
*Input parameters :        no parameters
*RETURNS          :        0 on success
*****************************************************************************/

int main(void)
{
        int iaArr[100],iNum,i,iTotal;
        int *iPtr;

        iPtr = iaArr;
        printf("\nEnter the number of elements\n");
        scanf("%d",&iNum);

        printf("\nEnter the elements\n");
        for(i=0;i<iNum;i++)
                scanf("%d",iPtr+i);

        iTotal = 0;
        for(i=0;i<iNum;i++)
        {
                iTotal = iTotal + *(iPtr+i);
        }

        printf("\nSum of elements = %d\n", iTotal);
        return 0;
}
```

Output

```
$ gcc 14aPointerArrSum.c
$ ./a.out

Enter the number of elements
5

Enter the elements
1 3 5 7 9

Sum of elements = 25
$ ./a.out

Enter the number of elements
4

Enter the elements
2 4 6 8

Sum of elements = 20
```

14.2 Dynamic Memory Allocation

Write a C program to find sum of n elements entered by the user. Demonstrate the program using functions malloc() to allocate memory dynamically and free() to deallocate.

C Code

```
/*****************************************************************************
*File         : 14bSumMalloc.c
*Description  : Program to demonstrate dynamic memory allocation
*Author       : Prabodh C P
*Compiler     : gcc 4.6.3 compiler, Ubuntu 14.04
*Date         : 26 June 2014
*****************************************************************************/

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

/*****************************************************************************
*Function         :        main
*Input parameters :        no parameters
*RETURNS          :        0 on success
*****************************************************************************/

int main()
{
        int *iPtr, iNum, i, iSum=0;

        printf("\nEnter number of elements : ");
        scanf("%d", &iNum);

        iPtr = (int*)malloc(iNum * sizeof(int));

        for(i=0;i<iNum;i++)
                scanf("%d", iPtr+i);

        for(i=0;i<iNum;i++)
                iSum += *(iPtr+i);

        printf("\nThe sum of elements is : %d\n", iSum);

        free(iPtr);
        return 0;
}
```

Output

```
$ gcc 14bSumMalloc.c
$ ./a.out

Enter number of elements : 4
12 23 34 45

The sum of elements is : 114

$ ./a.out

Enter number of elements : 6
98 87 76 65 54 45

The sum of elements is : 425
```

Chapter 15

Pointers and Arrays

15.1 Median of a list

Write a C program using pointers to find the median of a list of members.

C Code

```
/*****************************************************************************
*File         : 15aMedian.c
*Description  : Program to find the median of a list of members
*Author       : Prabodh C P
*Compiler     : gcc 4.6.3 compiler, Ubuntu 14.04
*Date         : 26 June 2014
*****************************************************************************/

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

/*****************************************************************************
*Function         :        main
*Input parameters :        no parameters
*RETURNS          :        0 on success
*****************************************************************************/

int main()
{
    int iNum, i, j;
    float faArr1[50], fTemp;
    float *fPtr;

    float fMedian;

    fPtr = faArr1;

    printf("\nEnter the no of elements in the array\n");
    scanf("%d",&iNum);

    printf("\nEnter elements of Array\n");
    for(i = 0; i < iNum; i++)
        scanf("%f",fPtr+i);

    printf("\nThe elements of Array\n");
    for(i = 0; i < iNum; i++)
        printf("%0.2f \t",*(fPtr+i));

    for(i = 0; i < iNum-1; i++)
    {
```

```
        for(j = i+1; j < iNum; j++)
        {
            if(faArr1[i] > faArr1[j])
            {
                fTemp = faArr1[i];
                faArr1[i] = faArr1[j];
                faArr1[j] = fTemp;
            }
        }
    }

    if(iNum%2)
        fMedian = *(fPtr+(iNum/2));
    else
        fMedian = (*(fPtr+(iNum/2)) + *(fPtr+(iNum/2)-1))/2.0f;


    printf("\nMedian = %g\n", fMedian);

    return 0;
}
```

Output

```
$ gcc 15aMedian.c
$ ./a.out

Enter the no of elements in the array
6

Enter elements of Array
1.2 2.3 3.4 4.5 5.6 6.7

The elements of Array
1.20         2.30         3.40         4.50         5.60         6.70

Median = 3.95

$ ./a.out

Enter the no of elements in the array
7

Enter elements of Array
1.2 2.3 3.4 4.5 5.6 6.7 7.6

The elements of Array
1.20         2.30         3.40         4.50         5.60         6.70         7.60

Median = 4.5
```

15.2 First Minimum

Write a C program using pointers to compute the sum of all elements stored in an array A[n]. Where n is the number of elements.

C Code

```
/*****************************************************************************
*File         : 15bSmallest.c
*Description  : Program to find first minimum element in an array
*Author       : Prabodh C P
*Compiler     : gcc 4.6.3 compiler, Ubuntu 14.04
*Date         : 26 June 2014
*****************************************************************************/

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

/*****************************************************************************
*Function         :        main
*Input parameters :        no parameters
*RETURNS          :        0 on success
*****************************************************************************/

int main()
{
    int iNum, i, j, iPos=1;
    float faArr1[50], fTemp;
    float *fPtr;

    float fSmall;

    fPtr = faArr1;

    printf("\nEnter the no of elements in the array\n");
    scanf("%d",&iNum);

    printf("\nEnter elements of Array\n");
    for(i = 0; i < iNum; i++)
        scanf("%f",fPtr+i);

    fSmall = *fPtr;
    for(i = 1; i < iNum; i++)
    {
        if(*(fPtr+i) < fSmall)
        {
            fSmall = *(fPtr+i);
            iPos = i+1;
        }
    }
    printf("\nThe elements of Array\n");
    for(i = 0; i < iNum; i++)
        printf("%0.2f \t",*(fPtr+i));

    printf("\nSmallest element is %0.2f and present at %d position\n", fSmall, iPos);
    return 0;
}
```

Output

```
$ gcc 15bSmallest.c
$ ./a.out

Enter the no of elements in the array
6

Enter elements of Array
4.6 1.8
23.9 -3.55 -5.4 9.7

The elements of Array
4.60         1.80         23.90         -3.55         -5.40         9.70
Smallest element is -5.40 and present at 5 position

$ ./a.out

Enter the no of elements in the array
4

Enter elements of Array
0 5.76 -3.7 9.3

The elements of Array
0.00         5.76         -3.70         9.30
Smallest element is -3.70 and present at 3 position
```