

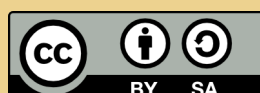
SIDDAGANGA INSTITUTE OF TECHNOLOGY

www.sit.ac.in

Foundations of Computer Programming Lab

LAB MANUAL

Prabodh C P
Asst Professor
Dept of CSE, SIT



This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

Contents

I PART A	4
1 Sales of a shop	5
2 Sum of last two digits	7
3 Boiling Points	9
4 Roots of a Quadratic Equation	12
5	15
5.1 Determine Quadrant	15
5.2 Sine Series	17
6 Print Calendar of a Month	19
7 Area under the curve $y=\log(x)$	21
8 Calculating nCr	23
9	25
9.1 Fibonacci Series	25
9.2 x raised to the power n	26
10 Bubble Sort and Binary Search	28
11 Selection Sort and Insertion Sort	31
II PART B	35
1	36
1.1 Set Difference	36
1.2 Duplicate element check in an array of elements	39
2 Matrix Construction	41
3 Matrix Multiplication	43
4 Swap smallest and largest elements in an array using pointers	47
5 Pass by Reference	49
6 Caesar Cipher	51
7 Substring Search	53
8 String Comparison	55
9 Bowler Information	57
10 Distance between two points	60
11 Bitwise Operations	63

FCP LABORATORY

Common to all Branches

Instructions

- All the C programs need to be executed on Codeblocks IDE using GCC Compiler.
- Algorithms and Flowcharts are compulsory for all the programs.
- Demo experiments are only for assignment purpose, not to be included in examinations.
- All experiments must be included in practical examinations.
- Students are allowed to pick one experiment each from Part A and Part B from the lots.

Demo Experiments

1. Demonstrate and explain the organization of a Personal Computer and its accessories.
2. Explanation of Concepts Computer Software, Computing Environment, Computer Languages, Creating and running programs

Students have to prepare a write-up for the above demonstrations and include it in the Lab record for evaluation.

References

Part A: Behrouz A. Forouzan , Richard F. Gilberg , Computer Science: **A Structured programming Approach Using C** - Cengage Learning; 3rd edition

For writing flowcharts refer to **Appendix C** of the above book.

Part I
PART A

Chapter 1

Sales of a shop

Write a program to calculate the total sales of a shop, given the unit price, quantity, discount rate and sales tax rate of a particular product. The sales tax is 8.5% which should be specified using a defined constant. The output should be displayed in proper form using appropriate format specifiers.

C Code

```
/******  
*File      : A01ShopSales.c  
*Descriptio : program to automate shop sales  
*Author    : Prabodh C P  
*Compiler  : gcc compiler, Ubuntu 14.04  
*Date     : 18 Novemeber 2014  
*****/  
#include <stdio.h>  
#include <stdlib.h>  
  
#define SALES_TAX 8.5  
  
int main()  
{  
    int iQuantity;  
    float fDiscRate, fDiscAmount, fUnitPrice, fSubTotal;  
    float fSubTaxable, fTaxAm, fTotal;  
    printf("\nEnter the number of items sold...\t:");  
    scanf("%d",&iQuantity);  
  
    printf("\nEnter the unit price... \t\t:");  
    scanf("%f",&fUnitPrice);  
  
    printf("\nEnter the discount rate (%)... \t\t:");  
    scanf("%f",&fDiscRate);  
  
    fSubTotal = iQuantity * fUnitPrice;  
    fDiscAmount = fSubTotal * fDiscRate / 100.0;  
    fSubTaxable = fSubTotal - fDiscAmount;  
    fTaxAm = fSubTaxable * SALES_TAX / 100.0;  
    fTotal = fSubTaxable + fTaxAm;  
  
    printf("\nQuantity sold:      \t%3d\n", iQuantity);  
    printf("Unit Price of items: %9.2f\n",fUnitPrice);  
    printf("-----\n");  
  
    printf("SubTotal :          %9.2f\n", fSubTotal);  
    printf("Discount :          -%8.2f\n", fDiscAmount);  
    printf("Discounted total :  %9.2f\n", fSubTaxable);
```

```
    printf("Sales Tax :          +%8.2f\n", fTaxAm);
    printf("Total sales :        %9.2f\n", fTotal);

    return 0;
}
```

Output

```
Enter the number of items sold...      :35
Enter the unit price...                 :15.55
Enter the discount rate (%)...         :6

Quantity sold:                          35
Unit Price of items:                     15.55
-----
SubTotal :                               544.25
Discount :                               - 32.65
Discounted total :                       511.60
Sales Tax :                               + 43.49
Total sales :                             555.08
```

```
Enter the number of items sold...      :43
Enter the unit price...                 :22.75
Enter the discount rate (%)...         :8

Quantity sold:                          43
Unit Price of items:                     22.75
-----
SubTotal :                               978.25
Discount :                               - 78.26
Discounted total :                       899.99
Sales Tax :                               + 76.50
Total sales :                             976.49
```

Chapter 2

Sum of last two digits

Write a program to extract and add the two least significant digits of an integer using the following user defined functions.

- i Function that obtains the unit place digit of a given number
- ii Function that obtains the tenth place digit of a given number
- iii Function that adds unit and tenth place digits of a given number

The main function should call these functions.

C Code

```
/*
*****
*File           : A02SumTwoDigits.c
*Descriptio    : Program to add the last two digits of a given number
*Author        : Prabodh C P
*Compiler      : gcc compiler, Ubuntu 14.04
*Date          : 4 August 2014
*****
#include <stdio.h>
#include <stdlib.h>
int fnExtractUnit(int);
int fnExtractTen(int);
int fnAddUnitTen(int, int);

int main(void)
{
    int iNum,iTen,iUnit,iSum;

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

    iUnit = fnExtractUnit(iNum);
    iTen = fnExtractTen(iNum);
    iSum = fnAddUnitTen(iTen, iUnit);

    printf("\nThe unit digit in %d is %d\n",iNum, iUnit);
    printf("\nThe ten's digit in %d is %d\n",iNum, iTen);
    printf("\nThe sum of last two digits in %d is %d\n",iNum, iSum);

    return 0;
}

int fnExtractUnit(int iVal)
{
    int iU;
```

```
    iU = (iVal%10);
    return iU;
}

int fnExtractTen(int iVal)
{
    int iT;
    iT = (iVal%100)/10;
    return iT;
}

int fnAddUnitTen(int iVal1, int iVal2)
{
    int iS;
    iS = (iVal1 + iVal2);
    return iS;
}
```

Output

Enter a number
4567

The unit digit in 4567 is 7

The ten's digit in 4567 is 6

The sum of last two digits in 4567 is 13

Enter a number
1209

The unit digit in 1209 is 9

The ten's digit in 1209 is 0

The sum of last two digits in 1209 is 9

Enter a number
76

The unit digit in 76 is 6

The ten's digit in 76 is 7

The sum of last two digits in 76 is 13

Chapter 3

Boiling Points

The following table shows the normal boiling points of several substances. Write a program that reads a boiling point of a substance and identifies the substance if the input boiling point is within 5% of the expected boiling point

Substance	Normal Boiling Point
Water	100
Mercury	357
Copper	1187
Silver	2193
Gold	2660

For example if the input is 104 it lies in the 5% range of 100 which is from 95 to 105. Hence the substance has to be water

C Code

```
/*
*****
*File      : A03BoilingPoint.c
*Description : Boiling Point
*Author    : Prabodh C P
*Compiler  : gcc compiler, Ubuntu 14.04
*Date     : 18 Novemeber 2014
*****
#include <stdio.h>
#include <stdlib.h>

int main(void)
{
    double dBoilPt;

    printf("\nEnter Boiling point of the substance\n");
    scanf("%lf",&dBoilPt);

    if (dBoilPt >= 0.95*100      &&      dBoilPt <= 1.05*100)
        printf("\nThe substance is Water\n");
    else if (dBoilPt >= 0.95*357      &&      dBoilPt <= 1.05*357)
        printf("\nThe substance is Mercury\n");
    else if (dBoilPt >= 0.95*1187      &&      dBoilPt <= 1.05*1187)
        printf("\nThe substance is Copper\n");
    else if (dBoilPt >= 0.95*2193      &&      dBoilPt <= 1.05*2193)
        printf("\nThe substance is Silver\n");
    else if (dBoilPt >= 0.95*2660      &&      dBoilPt <= 1.05*2660)
        printf("\nThe substance is Gold\n");
    else
        printf("\nInvalid Substance\n");
    return 0;
}
```


Output

Enter Boiling point of the substance
102

The substance is Water

Enter Boiling point of the substance
365

The substance is Mercury

Enter Boiling point of the substance
1200

The substance is Copper

Enter Boiling point of the substance
2205

The substance is Silver

Enter Boiling point of the substance
2670

The substance is Gold

Enter Boiling point of the substance
3333

Invalid Substance

Chapter 4

Roots of a Quadratic Equation

Write a program using switch statement, to find all the possible roots of Quadratic equation. Display suitable error messages for invalid inputs. Write a user defined function *fnCalcDiscriminant* to calculate the discriminant.

Summary

Any quadratic equation has two roots and the roots of the equation can be found using the formula

$$x = \frac{-b \pm \sqrt{\text{discriminant}}}{2 * a}$$

where $\text{discriminant} = b^2 - 4 * a * c$ and a,b and c are the coefficients of the quadratic equation $ax^2 + bx + c = 0$. Hence as per the equation, if discriminant is equal to 0, then the value of both the roots are equal and real. Also to find x, when discriminant is not 0, we need to find square root of disc. When discriminant value is less than 0, the square root of the discriminant is imaginary and hence the roots of the equation are supposed to be imaginary and distinct. When discriminant value is greater than 0, the square root of the discriminant is real and hence the roots of the equation are supposed to be real and distinct.

C Code

```
/******  
*File           : A04Quadratic.c  
*Description    : Program to find the roots of a Quadratic Equation  
*Author        : Prabodh C P  
*Compiler      : gcc compiler, Ubuntu 14.04  
*Date         : 3 August 2014  
*****/  
  
#include<stdio.h>  
#include<stdlib.h>  
#include<math.h>  
float fnCalcDiscriminant(float, float, float);  
/******  
*Function      :      main  
*Input parameters:      no parameters  
*RETURNS      :      0 on success  
*****/  
  
int main(void)  
{  
    float fA,fB,fC,fDesc,fX1,fX2,fRealp,fImagp;  
    int iState;  
  
    printf("\n*****");  
    printf("\n*\tPROGRAM TO FIND ROOTS OF A QUADRATIC EQUATION\t *");  
    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=fnCalcDiscriminant(fA,fB,fC);
((0 == fDesc) ? (iState = 1):((fDesc > 0) ? (iState = 2) : (iState = 3)));
switch(iState)
{
    case 1:
        fX1 = fX2 = -fB/(2*fA);
        printf("\nRoots are equal and the Roots are \n");
        printf("\nRoot1 = %g and Root2 = %g\n",fX1,fX2);
        break;
    case 2:
        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);
        break;
    case 3:
        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;
}

float fnCalcDiscriminant(float a, float b, float c)
{
    return (b*b-4*a*c);
}

```

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 5

5.1 Determine Quadrant

Write a program that takes the x and y coordinates of a point in the Cartesian plane and prints a message telling either the point is on the origin or on the axis on which the point lies or on the quadrant in which it is found.

Sample lines of output:

(-1.0, -2.5) is in quadrant III

(0.0, 4.8) is on the y-axis

C Code

```
/******  
*File           : A05aCheckQuadrant.c  
*Description    : Program to determine the quadrant in which a point lies  
*Author        : Prabodh C P  
*Compiler       : gcc compiler, Ubuntu 14.04  
*Date          : 18 Novemeber 2014  
*****/  
#include <stdio.h>  
#include <stdlib.h>  
  
int main(void)  
{  
    float fXcor,fYcor;  
    printf("\nEnter the coordinates\n");  
    scanf("%g%g",&fXcor,&fYcor);  
  
    if(0 == fXcor && 0 ==fYcor)  
    {  
        printf("\n(%g,%g) is at origin\n",fXcor,fYcor);  
    }  
    else if(0 == fXcor)  
    {  
        printf("\n(%g,%g) is on Y-axis\n",fXcor,fYcor);  
    }  
    else if(0 == fYcor)  
    {  
        printf("\n(%g,%g) is on X-axis\n",fXcor,fYcor);  
    }  
    else if(fXcor > 0 && fYcor > 0)  
    {  
        printf("\n(%g,%g) is in Quadrant I\n",fXcor,fYcor);  
    }  
    else if(fXcor < 0 && fYcor > 0)  
    {  
        printf("\n(%g,%g) is in Quadrant II\n",fXcor,fYcor);  
    }  
}
```

```
else if(fXcor < 0 && fYcor < 0)
{
    printf("\n(%g,%g) is in Quadrant III\n",fXcor,fYcor);
}
else
{
    printf("\n(%g,%g) is in Quadrant IV\n",fXcor,fYcor);
}

return 0;
}
```

Output

Enter the coordinates

1.2 3.4

(1.2,3.4) is in Quadrant I

Enter the coordinates

2.3 -4.5

(2.3,-4.5) is in Quadrant IV

Enter the coordinates

-9.9 -6.5

(-9.9,-6.5) is in Quadrant III

Enter the coordinates

-6.5 7.5

(-6.5,7.5) is in Quadrant II

Enter the coordinates

5 0

(5,0) is on X-axis

Enter the coordinates

0 9.6

(0,9.6) is on Y-axis

Enter the coordinates

0 0

(0,0) is at origin

5.2 Sine Series

Write a program to find Sine of an angle using the series

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} \dots\dots\dots$$

For given N terms using the for loop structure. Also print sin(x) using Library Function.

C Code

```

/*****
*File           : A05bSineAngle.c
*Descriptio    : Program to calculate the sine of an angle
*Author        : Prabodh C P
*Compiler      : gcc compiler, Ubuntu 14.04
*Date          : 18 Novemeber 2014
*****/
#include <stdio.h>
#include <math.h>

#define PI 3.1416

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

    printf("Enter the Angle ....\n");
    scanf("%f",&fAngD);
    printf("Angle = %g\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(%g) = %g\n",fAngD,fVal);
    printf("Built In function value is :sin(%g) = %g\n",fAngD, sin(fAngR));
    return 0;
}

```

Output

Enter the Angle

60

Angle = 60

Enter the Number of terms...

3

No of terms = 3

Calculated value is : $\sin(60) = 0.866297$

Built In function value is : $\sin(60) = 0.866027$

Enter the Angle

60

Angle = 60

Enter the Number of terms...

8

No of terms = 8

Calculated value is : $\sin(60) = 0.866027$

Built In function value is : $\sin(60) = 0.866027$

Enter the Angle

90

Angle = 90

Enter the Number of terms...

8

No of terms = 8

Calculated value is : $\sin(90) = 1$

Built In function value is : $\sin(90) = 1$

Chapter 6

Print Calendar of a Month

Write a program to display the calendar of a month whose starting day of the week and number of days in the month are given as input. (0,1,2,3...6 represent Sunday, Monday.....Saturday respectively). Write a function *printMonth* which takes these values as parameters and prints the calendar of the month. Perform input validation as well.

C Code

```
/*
*****
*File      : A06PrintCalendar.c
*Description : Program to print a calendar of the month
*Author    : Prabodh C P
*Compiler  : gcc compiler, Ubuntu 14.04
*Date     : 18 Novemeber 2014
*****
#include <stdio.h>
#include <stdlib.h>
void printMonth(int, int);

int main(void)
{
    int iNumOfDays, iStartDay;
    printf("\nEnter the day of the week on which the month starts\n");
    scanf("%d", &iStartDay);
    printf("\nEnter the number of days in the month\n");
    scanf("%d", &iNumOfDays);

    if(iStartDay < 0 || iStartDay > 6 || iNumOfDays >31 || iNumOfDays <28)
    {
        printf("\nInvalid Input\n");
        exit(0);
    }
    printMonth(iStartDay, iNumOfDays);
    return 0;
}

void printMonth(int iStDay, int iNoDays)
{
    int i;
    printf("\nSun\tMon\tTue\tWed\tThu\tFri\tSat\n");
    for(i=0;i<iStDay;i++)
        printf("\t");
    i=1;
    while(i<=iNoDays)
    {
        printf("%3d\t",i);
        if(0 == (i+iStDay)%7)
            printf("\n");
    }
}
```

```

        i++;
    }
}

```

Output

Enter the day of the week on which the month starts

3

Enter the number of days in the month

31

Sun	Mon	Tue	Wed	Thu	Fri	Sat
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

Enter the day of the week on which the month starts

6

Enter the number of days in the month

30

Sun	Mon	Tue	Wed	Thu	Fri	Sat
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30						

Enter the day of the week on which the month starts

7

Enter the number of days in the month

28

Invalid Input

Chapter 7

Area under the curve $y=\log(x)$

Write a Program to calculate the area under the curve $y = \log(x)$ within a given left and right limit. Use trapezoidal approximation. Write an user defined function *fnTrapArea* to calculate the area of a trapezoid.

C Code

```
/******  
*File          : A07AreaLogX.c  
*Description   : Program to calculate the area under the log(x) curve  
*Author       : Prabodh C P  
*Compiler     : gcc compiler, Ubuntu 14.04  
*Date        : 18 Novemeber 2014  
*****/  
#include <stdio.h>  
#include <stdlib.h>  
#include <math.h>  
  
double fnAreaTrapezoid(double, double, double);  
  
int main(void)  
{  
    double i, dDx=0.1,dX1,dX2,dY1,dY2;  
    double dLeft, dRight, dArea=0.0, dTotArea=0.0;  
  
    printf("\nEnter the left and right limit\n");  
    scanf("%lf%lf", &dLeft, &dRight);  
  
    printf("\nEnter the increment dx\n");  
    scanf("%lf", &dDx);  
  
    for(i=dLeft;i<dRight;i+=dDx)  
    {  
        dX1 = i;  
        dX2 = i+dDx;  
        dY1 = log(dX1);  
        dY2 = log(dX2);  
  
        dArea = fnAreaTrapezoid(dY1,dY2,dDx);  
        dTotArea += dArea;  
    }  
  
    printf("\nArea under the curve is %lf units\n", dTotArea);  
    return 0;  
}  
  
double fnAreaTrapezoid(double dHeight1, double dHeight2, double dBase)  
{
```

```
double dArea;  
dArea = dBase * ((dHeight1 + dHeight2)/2);  
return dArea;  
}
```

Output

```
Enter the left and right limit  
2 5
```

```
Enter the increment dx  
0.001
```

```
Area under the curve is 3.660895 units
```

```
Enter the left and right limit  
1 2
```

```
Enter the increment dx  
0.001
```

```
Area under the curve is 0.386988 units
```

```
Enter the left and right limit  
6 9
```

```
Enter the increment dx  
0.01
```

```
Area under the curve is 6.046442 units
```

Chapter 8

Calculating nCr

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 program to compute the binomial coefficient nCr. Perform input validation as well.

C Code

```
/******  
*File           : A08RecNCR.c  
*Description    : Program to calculate NCR recursively  
*Author        : Prabodh C P  
*Compiler      : gcc compiler, Ubuntu 14.04  
*Date         : 18 Novemeber 2014  
*****/  
#include <stdio.h>  
#include <stdlib.h>  
  
int fnFactorial(int);  
  
int main(void)  
{  
    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

Enter the value of n
3 6

Enter the value of r

nCr does not exist

Enter the value of n
5 2

Enter the value of r

For n = 5 and r = 2, $5C2 = 10$

Enter the value of n
7 5

Enter the value of r

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

Chapter 9

9.1 Fibonacci Series

Write a Program to generate Fibonacci series of N numbers using recursion.

C Code

```
/******  
*File      : A09aFibonacci.c  
*Descriptio : Program to print Fibonacci Series  
*Author    : Prabodh C P  
*Compiler  : gcc compiler, Ubuntu 14.04  
*Date     : 18 Novemeber 2014  
*****/  
#include <stdio.h>  
#include <stdlib.h>  
  
int fnFibonacci(int);  
  
int main(void)  
{  
    int iTerm, i, iNum;  
  
    printf("Enter the no of terms\n");  
    scanf("%d", &iNum);  
  
    printf("\nThe terms in the Fibonacci Sequence:\n");  
    for(i=0;i<iNum;++i)  
    {  
        iTerm = fnFibonacci(i);  
        printf("%d\t", iTerm);  
    }  
    return 0;  
}  
  
int fnFibonacci(int n)  
{  
    if(0 == n || 1 == n)  
        return n;  
    else  
    {  
        return fnFibonacci(n-1) + fnFibonacci(n-2);  
    }  
}
```

Output

Enter the no of terms

5

The terms in the Fibonacci Sequence:

0 1 1 2 3

9.2 x raised to the power n

Write a Program in C to find the value of 'x' raised to the power 'n' using recursion.

C Code

```

/*****
*File      : A09bfnRecPow.c
*Description : Program to calculate x raised to n recursively
*Author    : Prabodh C P
*Compiler  : gcc compiler, Ubuntu 14.04
*Date     : 18 Novemeber 2014
*****/
#include <stdio.h>
#include <stdlib.h>

int fnRecPow(int, int);
int main(void)
{
    int iX, iN;
    float fRes;
    printf("\n Enter value of x and n\n");
    scanf("%d%d", &iX, &iN);

    if(iN<0)
    {
        fRes = 1.0/fnRecPow(iX,-iN);
    }
    else
    {
        fRes = fnRecPow(iX,iN);
    }
    printf("\n%d raised to the power %d is %d\n",iX, iN, fRes);
    return 0;
}

int fnRecPow(int iX, int iN)
{
    if(0==iN)
        return 1;
    else if(1 == iN)
        return iX;
    else
        return (iX * fnRecPow(iX, iN-1));
}

```

Output

putta@putta-Lenovo-G580:~/Latex/Aug2nd/progs

```
Enter value of x and n
```

```
4 5
```

```
4 raised to the power 5 is 1024
```

```
putta@putta-Lenovo-G580:~/Latex/Aug2nd/progs
```

```
Enter value of x and n
```

```
9 9
```

```
9 raised to the power 9 is 387420489
```

Chapter 10

Bubble Sort and Binary Search

Write a 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. Write user defined functions for the following

- i reads N integer numbers
- ii sorts N integer numbers in ascending order using bubble sort
- iii display N integers
- iv search for a key element using Binary Search

C Code

```
/******  
*File           : A10BubbleBinary.c  
*Description    : Program to sort and search an array using Bubble sort  
*                and Binary search  
*Author        : Prabodh C P  
*Compiler       : gcc compiler, Ubuntu 14.04  
*Date          : 18 Novemeber 2014  
*****/  
#include <stdio.h>  
#include <stdlib.h>  
  
void fnReadArray(int [], int);  
void fnSortArray(int [], int);  
void fnDispArray(int [], int);  
int fnSearchArray(int [], int, int);  
  
int main(void)  
{  
    int iaList[100], iKey, iNum, iPos;  
  
    printf("\nEnter the no. of elements\n");  
    scanf("%d", &iNum);  
  
    printf("\nEnter %d elements\n", iNum);  
    fnReadArray(iaList, iNum);  
    printf("\nEntered values are :\n");  
    fnDispArray(iaList, iNum);  
  
    fnSortArray(iaList, iNum);  
  
    printf("\nAfter Sorting :\n");  
    fnDispArray(iaList, iNum);  
}
```

```

printf("\nEnter the key element\n");
scanf("%d", &iKey);
iPos = fnSearchArray(iaList, iKey, iNum);

if(-1 == iPos)
{
    printf("\nElement not found\n");
}
else
{
    printf("\nElement %d found at position %d\n", iKey, iPos+1);
}
return 0;
}

void fnReadArray(int iaArr[], int iNum)
{
    int i;
    for(i=0;i<iNum;i++)
        scanf("%d", &iaArr[i]);
}

void fnDispArray(int iaArr[], int iNum)
{
    int i;
    for(i=0;i<iNum;i++)
        printf("\t%d", iaArr[i]);
    printf("\n");
}

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

int fnSearchArray(int iaArr[], int iKey, int iNum)
{
    int iProbe, iLow, iHigh, iMid;
    iLow = 0;
    iHigh = iNum - 1;
    iProbe = 0;
    while(iLow <= iHigh)
    {
        iProbe++;
        iMid = (iLow + iHigh)/2;
        //printf("\nMid element = %d\n",iaArr[iMid]);
        if(iKey == iaArr[iMid])
        {
            printf("\nTotal number of probes required is %d\n", iProbe);

```

```
        return iMid;
    }
    else if(iKey < iaArr[iMid])
        iHigh = iMid-1;
    else
        iLow = iMid +1;
}
printf("\nTotal number of probes required is %d\n", iProbe);
return -1;
}
```

Output

Enter the number of elements

5

Enter 5 elements

1 7 4 2 9

Sorted Input Array

1 2 4 7 9

Enter the key element

4

Element 4 found at 3 position

Total number of probes required is 1

Enter the number of elements

3

Enter 3 elements

9 7 5

Sorted Input Array

5 7 9

Enter the key element

8

Element not found

Total number of probes required is 2

Chapter 11

Selection Sort and Insertion Sort

Write a program to implement sorting with the following user defined functions.

- i function to generate n random numbers in the range 0 to 999 and store it in an array.
- ii function to sort the array using Selection sort.
- iii function to sort the array using Insertion sort.
- iv function to display the elements of an array.

C Code

```
/******  
*File           : A11InsertSelectSort.c  
*Description    : Program to sort an array using Insertion Sort  
*Author        : Prabodh C P  
*Compiler      : gcc compiler 4.6.3, Ubuntu 12.04  
*Date         : 22 Nov 2013  
*****/  
  
#include <stdio.h>  
#include <stdlib.h>  
#include <time.h>  
  
void fnGenRandInput(int [], int);  
void fnDispArray( int [], int);  
void fnInsertionSort(int [], int);  
void fnSelectSort(int [], int);  
  
/******  
*Function      : main  
*Input parameters: no parameters  
*RETURNS      : 0 on success  
*****/  
int main(void)  
{  
  
    int iaArr[100],iNum;  
  
    printf("\nInsertion Sort\n");  
    printf("\nEnter the number of elements to sort\n");  
    scanf("%d",&iNum);  
    fnGenRandInput(iaArr,iNum);  
    printf("\nUnsorted Array\n");  
    fnDispArray(iaArr,iNum);  
    fnInsertionSort(iaArr,iNum);  
    printf("\nSorted Array\n");  
}
```

```

    fnDispArray(iaArr,iNum);

    printf("\nSelection Sort\n");
    printf("\nEnter the number of elements to sort\n");
    scanf("%d",&iNum);
    fnGenRandInput(iaArr,iNum);
    printf("\nUnsorted Array\n");
    fnDispArray(iaArr,iNum);
    fnSelectSort(iaArr,iNum);
    printf("\nSorted Array\n");
    fnDispArray(iaArr,iNum);

    return 0;
}

/*****
*Function      : fnSelectSort
*Description   : Function to sort elements in an iaArray using Quick Sort
*Input parameters:
*      int A[] - iaArray to hold integers
*      int n    - no of elements in the array
*RETURNS      : no value
*****/

void fnSelectSort(int A[], int n)
{
    int i,j,iMinPos,iTemp;
    for(i=0;i<n;i++)
    {
        iMinPos = i;
        for(j=i+1;j<n;j++)
        {
            if(A[j]<A[iMinPos])
                iMinPos = j;
        }
        iTemp = A[i];
        A[i] = A[iMinPos];
        A[iMinPos] = iTemp;
    }
}

/*****
*Function      : fnInsertionSort
*Description   : Function to sort an array using Insertion Sort
*Input parameters:
*      int A[] - iaArray to hold integers
*      int n    - no of elements in the array
*RETURNS      : no value
*****/

void fnInsertionSort(int A[], int n)
{
    int i, j, iKey;
    for(i=1;i<n;i++)
    {
        iKey = A[i];
        j = i-1;
        while(j>=0 && A[j] > iKey)
        {
            A[j+1] = A[j];
            j--;
        }
    }
}

```



```

        A[j+1] = iKey;
    }
}

/*****
*Function      : GenRandInput
*Description   : Function to generate a fixed number of random elements
*Input parameters:
*      int X[] - array to hold integers
*      int n   - no of elements in the array
*RETURNS      :no return value
*****/

void fnGenRandInput(int X[], int n)
{
    int i;
    srand(time(NULL));
    for(i=0;i<n;i++)
    {
        X[i] = rand()%1000;
    }
}

/*****
*Function      : DispArray
*Description   : Function to display elements of an array
*Input parameters:
*      int X[] - array to hold integers
*      int n   - no of elements in the array
*RETURNS      : no return value
*****/

void fnDispArray( int X[], int n)
{
    int i;
    for(i=0;i<n;i++)
        printf(" %4d \n",X[i]);
}

```

Output

Insertion Sort

Enter the number of elements to sort

5

Unsorted Array

845

154

879

668

214

Sorted Array

154

214

668

845

879

Selection Sort

Enter the number of elements to sort

5

Unsorted Array

925

281

200

162

484

Sorted Array

162

200

281

484

925

Part II

PART B

Chapter 1

1.1 Set Difference

Write a program to find the Difference of two sets(sets have distinct elements)

C Code

```
/******  
*File      : B01aArrayDiff.c  
*Description : Program to find the difference of two sets  
*Author    : Prabodh C P  
*Compiler  : gcc compiler, Ubuntu 14.04  
*Date     : 18 Novemeber 2014  
*****/  
  
#include <stdio.h>  
#include <stdlib.h>  
  
#define TRUE 1  
#define FALSE 0  
int main(void)  
{  
    int iaA[10], iaB[10], iaC[10], iNum1, iNum2;  
    int i, j, iNotOccur, iCount = 0;  
  
    printf("\nEnter no of elements in Array A\n");  
    scanf("%d",&iNum1);  
    printf("\nEnter no of elements in Array B\n");  
    scanf("%d",&iNum2);  
  
    printf("\nEnter %d elements in Array A\n",iNum1);  
    for(i=0;i<iNum1;++i)  
    {  
        scanf("%d",&iaA[i]);  
    }  
    printf("\nEnter %d elements in Array B\n",iNum2);  
    for(i=0;i<iNum2;++i)  
    {  
        scanf("%d",&iaB[i]);  
    }  
  
    /*Copy those elements of X that are not in Y into C*/  
    for(i=0;i<iNum1;i++)  
    {  
        iNotOccur = TRUE;  
        for(j=0;j<iNum2;j++)  
        {  
            if(iaA[i] == iaB[j])  
                iNotOccur = FALSE;  
        }  
    }  
}
```

```

    }
    if(iNotOccur == TRUE)
    {
        iaC[iCount++] = iaA[i];
    }
}
printf("\nElements of Array A:\n");
for(i=0;i<iNum1;++i)
{
    printf("%d\t",iaA[i]);
}

printf("\nElements of Array B:\n");
for(i=0;i<iNum2;++i)
{
    printf("%d\t",iaB[i]);
}

printf("\nDifference of A and B: \n");
for(i=0;i<iCount;++i)
{
    printf("%d\t",iaC[i]);
}
printf("\n");
return 0;
}

```

Output

Enter no of elements in Array A
4

Enter no of elements in Array B
5

Enter 4 elements in Array A
1 2 3 4

Enter 5 elements in Array B
3 4 5 6 7

Elements of Array A:
1 2 3 4
Elements of Array B:
3 4 5 6 7
Difference of A and B:
1 2

Enter no of elements in Array A
5

Enter no of elements in Array B
4

Enter 5 elements in Array A
3 4 5 6 7

Enter 4 elements in Array B
1 2 3 4

```
Elements of Array A:
3      4      5      6      7
Elements of Array B:
1      2      3      4
Difference of A and B:
5      6      7
```

```
Enter no of elements in Array A
5
```

```
Enter no of elements in Array B
4
```

```
Enter 5 elements in Array A
3 5 7 4 6
```

```
Enter 4 elements in Array B
4 1 2 3
```

```
Elements of Array A:
3      5      7      4      6
Elements of Array B:
4      1      2      3
Difference of A and B:
5      7      6
```

1.2 Duplicate element check in an array of elements

Write a program to find whether an array contains duplicate elements or not.

C Code

```

/*****
*File      : B01bDuplicate.c
*Description : Program to check whether an array contains duplicate elements
*Author    : Prabodh C P
*Compiler  : gcc compiler, Ubuntu 14.04
*Date     : 18 Novemeber 2014
*****/
#include <stdio.h>
#include <stdlib.h>

int main(void)
{
    int iaA[10], iNum;
    int i, j;

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

    printf("\nEnter %d elements in Array A\n",iNum);
    for(i=0;i<iNum;++i)
    {
        scanf("%d",&iaA[i]);
    }

    for(i=0;i<iNum;++i)
    {
        for(j=i+1;j<iNum;++j)
        {
            if(iaA[i] == iaA[j])
            {
                printf("\nArray contains duplicate elements\n");
                exit(0);
            }
        }
    }

    printf("\nArray does not contain duplicate elements\n");
    return 0;
}

```

Output

```
putta@putta-Lenovo-G580:~/Latex/Aug2nd/progs
```

```
Enter no of elements in Array
5
```

```
Enter 5 elements in Array A
1 2 3 1 2
```

```
Array contains duplicate elements
putta@putta-Lenovo-G580:~/Latex/Aug2nd/progs
```

```
Enter no of elements in Array
```

5

Enter 5 elements in Array A

2 4 6 8 9

Array does not contain duplicate elements

Chapter 2

Matrix Construction

Write a program that reads a matrix of order $M \times N$ and fills the matrix by following:

- i Lower triangle with '1' s
- ii Upper triangle with '-1' s
- iii Principal diagonal with zeros

Check the necessity condition needed and display the contents of the matrix in proper format.

C Code

```
/******  
*File          : B02TriMatrix.c  
*Description   : Program to construct a matrix  
*Author       : Prabodh C P  
*Compiler     : gcc compiler, Ubuntu 14.04  
*Date        : 18 Novemeber 2014  
*****/  
#include<stdio.h>  
#include<stdlib.h>  
  
int main(void)  
{  
    int iaMat[100][100],iRow,iCol,i,j;  
  
    printf("\nEnter the order of the matrix\n");  
    scanf("%d%d",&iRow,&iCol);  
  
    if(iRow != iCol)  
    {  
        printf("\nMatrix should be a Square Matrix\n");  
        exit(0);  
    }  
    for(i=0;i<iRow;i++)  
    {  
        for(j=0;j<iCol;j++)  
        {  
            if(i < j)  
                iaMat[i][j] = -1;  
            else if(i == j)  
                iaMat[i][j] = 0;  
            else  
                iaMat[i][j] = 1;  
        }  
    }  
    printf("\nMatrix \n");  
}
```

```
for(i=0;i<iRow;i++)
{
    for(j=0;j<iCol;j++)
    {
        printf("%2d\t",iaMat[i][j]);
    }
    printf("\n");
}
return 0;
}
```

Output

Enter the order of the matrix
4 5

Matrix should be a Square Matrix

Enter the order of the matrix
5 5

Matrix

0	-1	-1	-1	-1
1	0	-1	-1	-1
1	1	0	-1	-1
1	1	1	0	-1
1	1	1	1	0

Chapter 3

Matrix Multiplication

Write a program that reads two matrices A ($m \times n$) and B ($p \times q$) and Compute the product A X 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. Display the trace of the matrix if it exists. Program must check the compatibility of orders of the matrices for multiplication. Report appropriate message in case of incompatibility.

C Code

```
/******  
*File           : MatrixMul.c  
*Description    : Program to implement Matrix Multiplication  
*Author        : Prabodh C P  
*Compiler      : gcc 4.6.3 compiler, Ubuntu 10.04  
*Date         : Wednesday, July 16 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], iTrace = 0;  
    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 \n");

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

if(iM == iQ)
{
    for(i=0;i<iM;i++)
        iTrace += iaProd[i][i];
    printf("\nTrace of the product matrix is : %d\n", iTrace);
}
else
{
    printf("\nTrace does not exist\n");
}
return 0;
}

```

Output

```

*****
*          PROGRAM TO IMPLEMENT MATRIX MULIPLICATION          *
*****
Enter the order of Matrix1
2 3

Enter the order of Matrix2
3 2

Enter the elements of Matrix 1 in row major order
1 2 3 4 5 6

Enter the elements of Matrix 2 in column major order
1 2 3 4 5 6

Matrix 1
1      2      3
4      5      6

Matrix 2
1      4
2      5
3      6

The Product matrix is is
14      32
32      77

```

Trace of the product matrix is : 91

```
*****  
*          PROGRAM TO IMPLEMENT MATRIX MULIPLICATION          *  
*****  
Enter the order of Matrix1  
3 4  
  
Enter the order of Matrix2  
3 4  
  
Matrix Multiplication not possible
```

Chapter 4

Swap smallest and largest elements in an array using pointers

Write a program to find the smallest and largest elements in an array using pointers and then swap these elements and display the resultant array.

C Code

```
/******  
*File      : B04LarSmallSwapPtr.c  
*Description : Program to swap smallest and largest elements in an array  
*              using pointers  
*Author     : Prabodh C P  
*Compiler   : gcc compiler, Ubuntu 14.04  
*Date      : 18 Novemeber 2014  
*****/  
#include <stdio.h>  
#include <stdlib.h>  
  
int main(void)  
{  
    int iaList[50], iNum, i, j, iTemp, iPos;  
    int *iPtr1,*iPtr2;  
  
    iPtr1 = iPtr2 = iaList;  
    printf("\nEnter the value of n : ");  
    scanf("%d", &iNum);  
  
    printf("\nEnter the elements: ");  
    for(i=0;i<iNum;++i)  
    {  
        scanf("%d", &iaList[i]);  
    }  
  
    for(i=1;i<iNum;++i)  
    {  
        if(iaList[i] < *iPtr1)  
        {  
            iPtr1 = &iaList[i];  
        }  
        if(iaList[i] > *iPtr2)  
        {  
            iPtr2 = &iaList[i];  
        }  
    }  
  
    printf("\nArray elements before swapping the largest and smallest elements\n");
```

```

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

printf("\nSmallest Element = %d\n",*iPtr1);
printf("\nLargest Element = %d\n",*iPtr2);

iTemp = *iPtr1;
*iPtr1 = *iPtr2;
*iPtr2 = iTemp;

printf("\nArray elements after swapping the largest and smallest elements\n");
for(i=0;i<iNum;i++)
{
    printf("%d\t", iaList[i]);
}
printf("\n");
return 0;
}

```

Output

Enter the value of n : 5

Enter the elements: 5 4 3 2 1

Array elements before swapping the largest and smallest elements

5 4 3 2 1

Smallest Element = 1

Largest Element = 5

Array elements after swapping the largest and smallest elements

1 4 3 2 5

Enter the value of n : 6

Enter the elements: 3 2 1 4 5 6

Array elements before swapping the largest and smallest elements

3 2 1 4 5 6

Smallest Element = 1

Largest Element = 6

Array elements after swapping the largest and smallest elements

3 2 6 4 5 1

Chapter 5

Pass by Reference

Write a program that reads the length, breadth and height of a cuboid. The program has to calculate the surface area and volume of the cuboid using the function *fnCalcVolSurfArea*. The results are then to be displayed by the main function. (Hint: use pass by reference)

C Code

```
/******  
*File          : B05SurfAreaVolCuboid.c  
*Description   : Program to illustrate pass by reference  
*Author        : Prabodh C P  
*Compiler      : gcc compiler, Ubuntu 14.04  
*Date          : 18 Novemeber 2014  
*****/  
#include <stdio.h>  
#include <stdlib.h>  
  
void fnCalcVolSurfArea(int, int, int, int*, int*);  
  
int main(void)  
{  
    int iLength, iBreadth, iHeight, iSurfArea, iVolume;  
    printf("\nEnter the length, breadth and height of the cuboid\n");  
    scanf("%d%d%d", &iLength, &iBreadth, &iHeight);  
    fnCalcVolSurfArea(iLength, iBreadth, iHeight, &iSurfArea, &iVolume);  
    printf("\nSurface Area of cuboid is %d units\n", iSurfArea);  
    printf("\nVolume of cuboid is %d cubic units\n", iVolume);  
    return 0;  
}  
void fnCalcVolSurfArea(int iL, int iB, int iH, int *iA, int *iV)  
{  
    *iA = 2*iL*iB + 2*iB*iH + 2*iH*iL;  
    *iV = iL*iB*iH;  
}
```

Output

```
Enter the length, breadth and height of the cuboid  
3 4 5  
  
Surface Area of cuboid is 94 units  
  
Volume of cuboid is 60 cubic units
```

Enter the length, breadth and height of the cuboid

6 6 6

Surface Area of cuboid is 216 units

Volume of cuboid is 216 cubic units

Chapter 6

Caesar Cipher

Write a Program to implement Caesar Cipher. Input a message and then encodes it by replacing each character in the message by a character that is three positions ahead in the English alphabet sequence, wrap back to 'a' if the character is 'z'. Display the encoded message. Decode the message using the inverse procedure and display it.

C Code

```
/******  
*File      : B06Caesar.c  
*Description : program to implement Caesar Cipher  
*Author    : Prabodh C P  
*Compiler  : gcc compiler, Ubuntu 14.04  
*Date     : 18 Novemeber 2014  
*****/  
#include <stdio.h>  
#include <stdlib.h>  
#include <ctype.h>  
void fnEncrypt(char []);  
void fnDecrypt(char []);  
  
int main(void)  
{  
    char acMesg[50];  
  
    printf("\nEnter your plain text in lower case\n");  
    gets(acMesg);  
  
    fnEncrypt(acMesg);  
    printf("Encrypted Ciphertext\n");  
    puts(acMesg);  
  
    fnDecrypt(acMesg);  
    printf("Decrypted Plaintext\n");  
    puts(acMesg);  
  
    return 0;  
}  
  
void fnEncrypt(char acStr[50])  
{  
    int i;  
    for(i=0;acStr[i]!='\0';i++)  
    {  
        if(!isspace(acStr[i]))  
            acStr[i] = (acStr[i]-'a'+3+26)%26+'a';  
    }  
}
```

```
}  
void fnDecrypt(char acStr[50])  
{  
    int i;  
    for(i=0;acStr[i]!='\0';i++)  
    {  
        if(!isspace(acStr[i]))  
            acStr[i] = (acStr[i]-'a'-3+26)%26+'a';  
    }  
}
```

Output

```
Enter your plain text in lower case  
ramanu kaadige hodanu  
Encrypted Ciphertext  
udpdqx nddgljh krgdqx  
Decrypted Plaintext  
ramanu kaadige hodanu
```

```
Enter your plain text in lower case  
you are what you think you are  
Encrypted Ciphertext  
brx duh zkdw brx wklqn brx duh  
Decrypted Plaintext  
you are what you think you are
```

Chapter 7

Substring Search

Write a program to read a line of text from the keyboard and print the number of occurrences of a given substring using the builtin function strstr().

C Code

```
/******  
*File          : B07StrSearch.c  
*Description   : program to search for a sub string in a given string  
*Author       : Prabodh C P  
*Compiler     : gcc compiler, Ubuntu 14.04  
*Date        : 18 Novemeber 2014  
*****/  
#include <stdio.h>  
#include <stdlib.h>  
#include <string.h>  
  
int main()  
{  
    char acText[100], acPattern[10];  
    int iCount = 0 ;  
    char *pos = NULL, *start = NULL;  
    printf("\nEnter the text\n");  
    gets(acText);  
  
    printf("\nEnter the substring to search for\n");  
    gets(acPattern);  
  
    start = acText;  
    while((pos = strstr(start, acPattern)))  
    {  
        iCount++;  
        printf("\nSubstring found at position %ld\n", pos-acText+1);  
        start = pos+1;  
    }  
  
    printf("\nNo of occurences is %d\n", iCount);  
  
    return 0;  
}
```

Output

Enter the text
The cat with five legs ran down the hall

Enter the substring to search for
he

Substring found at position 2

Substring found at position 34

No of occurrences is 2

Enter the text
Freedom of the mind

Enter the substring to search for
faith

No of occurrences is 0

Chapter 8

String Comparison

Write a program to read two strings and then determine whether they are identical or not using a user defined function `fnMyStrCmp` which takes these strings as parameters. Display the result in the main function

C Code

```
/******  
*File          : B08MyStrCmp.c  
*Description   : program to perform string comparison  
*Author       : Prabodh C P  
*Compiler     : gcc compiler, Ubuntu 14.04  
*Date        : 18 Novemeber 2014  
*****/  
#include <stdio.h>  
#include <stdlib.h>  
#include <stdbool.h>  
  
bool fnMyStrCmp(char [], char []);  
  
int main()  
{  
    char acStr1[50],acStr2[50];  
    bool notIdentical;  
  
    printf("\nEnter string1\n");  
    gets(acStr1);  
  
    printf("\nEnter string2\n");  
    gets(acStr2);  
  
    notIdentical = fnMyStrCmp(acStr1, acStr2);  
  
    if(notIdentical)  
        printf("\nThe strings are not identical\n");  
    else  
        printf("\nThe strings are identical\n");  
  
    return 0;  
}  
  
bool fnMyStrCmp(char str1[], char str2[])  
{  
    int i,j;  
    bool notSame = false;  
    for(i=0; str1[i] != '\0';i++);  
    for(j=0; str2[j] != '\0';j++);
```

```
    if(i != j)
    {
        notSame = true;
        return notSame;
    }
    for(i=0; str1[i] != '\0';i++)
    {
        if(str1[i] != str2[i])
        {
            notSame = true;
            return notSame;
        }
    }
    return notSame;
}
```

Output

```
Enter string1
Tumkur
```

```
Enter string2
Bangalore
```

The strings are not identical

```
Enter string1
Tumkur
```

```
Enter string2
TUMKUR
```

The strings are not identical

```
Enter string1
tumkur
```

```
Enter string2
tumkur
```

The strings are identical

```
Enter string1
Tumkur
```

```
Enter string2
Tumkur City
```

The strings are not identical

Chapter 9

Bowler Information

Write a program that stores the details of 5 bowlers given by the user. The following are the information stored for each bowler

- a Name
- b Nationality
- c No of matches played
- d No of Wickets taken
- e No of overs bowled
- f No of runs conceded

Then find and display details of

- i the most economic bowler
- ii the bowler with the best strike rate

C Code

```
/******  
*File          : B09Bowler.c  
*Description   : program to analyse bowler information  
*Author        : Prabodh C P  
*Compiler      : gcc compiler, Ubuntu 14.04  
*Date          : 18 Novemeber 2014  
*****/  
#include <stdio.h>  
#include <stdlib.h>  
  
typedef struct  
{  
    char acName[30];  
    char acNationality[30];  
    int iMatches, iWickets, iOvers, iRuns;  
}BOWLER;  
  
int main(void)  
{  
    BOWLER bowlers[5];  
    int i, iPosEconomy=0, iPosStrRate=0;  
    float fEconomy[5], fStrRate[5],fBestEconomy,fBestStrRate;  
    printf("\nEnter details\n");  
  
    for(i=0;i<5;i++)  
    {  
        printf("Name : ");
```

```

scanf("%s", bowlers[i].acName);
printf("Nationality : ");
scanf("%s", bowlers[i].acNationality);
printf("Matches Played : ");
scanf("%d", &bowlers[i].iMatches);
printf("Wickets taken : ");
scanf("%d", &bowlers[i].iWickets);
printf("Overs Bowled : ");
scanf("%d", &bowlers[i].iOvers);
printf("Runs Conceded : ");
scanf("%d", &bowlers[i].iRuns);
}

fBestEconomy = fEconomy[0] = (float)bowlers[0].iRuns/bowlers[0].iOvers;
fBestStrRate = fStrRate[0] = (float)bowlers[0].iOvers/bowlers[0].iWickets;
printf("Name\t\tNation\t\tMatches Wickets Overs Runs Economy SR\n");

for(i=0;i<5;i++)
{
    fEconomy[i] = (float)bowlers[i].iRuns/bowlers[i].iOvers;
    fStrRate[i] = (float)bowlers[i].iOvers/bowlers[i].iWickets;
    printf("%-16s", bowlers[i].acName);
    printf("%-10s\t", bowlers[i].acNationality);
    printf("%3d", bowlers[i].iMatches);
    printf("%9d", bowlers[i].iWickets);
    printf("%12d", bowlers[i].iOvers);
    printf("%6d", bowlers[i].iRuns);
    printf("%8.2f", fEconomy[i]);
    printf("%8.2f", fStrRate[i]);
    printf("\n");

    if(fBestEconomy > fEconomy[i])
        iPosEconomy = i;
    if(fBestStrRate > fStrRate[i])
        iPosStrRate = i;
}
printf("\nThe Bowler with the best economy is : %s\n", bowlers[iPosEconomy].acName);
printf("\nThe Bowler with the best Strike rate is : %s\n\n", bowlers[iPosStrRate].acName);
return 0;
}

```

Output

Enter details

Name : Ramesh
 Nationality : India
 Matches Played : 123
 Wickets taken : 99
 Overs Bowled : 1200
 Runs Conceded : 4567

Name : Mushtaq
 Nationality : Bangladesh
 Matches Played : 56
 Wickets taken : 50
 Overs Bowled : 510
 Runs Conceded : 2200

Name : Prakash
 Nationality : India
 Matches Played : 212
 Wickets taken : 176
 Overs Bowled : 2096
 Runs Conceded : 6036

Name : Stewart
 Nationality : England
 Matches Played : 98
 Wickets taken : 65
 Overs Bowled : 910
 Runs Conceded : 7122

Name : Chandana
 Nationality : Srilanka
 Matches Played : 154
 Wickets taken : 136
 Overs Bowled : 1321
 Runs Conceded : 9999

Name	Nation	Matches	Wickets	Overs	Runs	Economy	SR
Ramesh	India	123	99	1200	4567	3.81	12.12
Mushtaq	Bangladesh	56	50	510	2200	4.31	10.20
Prakash	India	212	176	2096	6036	2.88	11.91
Stewart	England	98	65	910	7122	7.83	14.00
Chandana	Srilanka	154	136	1321	9999	7.57	9.71

The Bowler with the best economy is : Prakash

The Bowler with the best Strike rate is : Chandana

Chapter 10

Distance between two points

Write a program that create a structure to represent a point in a cartesian plane. The program has to read coordinates of two points as input. Using a user defined function *fnCalcDistance* which takes these points as parameters calculate the distance between the two points. Display the result in the main function. (Hint : use Distance formula)

C Code

```
/******  
*File      : B10StructDist.c  
*Descriptio : program to find the distance between two cartesian points  
*Author    : Prabodh C P  
*Compiler  : gcc compiler, Ubuntu 14.04  
*Date     : 18 Novemeber 2014  
*****/  
#include <stdio.h>  
#include <stdlib.h>  
#include <math.h>  
  
typedef struct  
{  
    int iXcor, iYcor;  
}POINT;  
  
int main(void)  
{  
    POINT p1, p2;  
  
    float fDistance;  
  
    int iDiffX,iDiffY;  
  
    printf("\nEnter coordinates of the first point\n");  
    scanf("%d%d",&p1.iXcor,&p1.iYcor);  
  
    printf("\nEnter coordinates of the second point\n");  
    scanf("%d%d",&p2.iXcor,&p2.iYcor);  
  
    iDiffX = p2.iXcor - p1.iXcor;  
    iDiffY = p2.iYcor - p1.iYcor;  
  
    fDistance = sqrt(((iDiffX)*(iDiffX)) + ((iDiffY)*(iDiffY)));  
  
    printf("\nDistance between the two points is %g units\n",fDistance);  
  
    return 0;  
}
```

}

Output

Enter coordinates of the first point
0 0

Enter coordinates of the second point
5 0

Distance between the two points is 5 units

Enter coordinates of the first point
5 5

Enter coordinates of the second point
5 9

Distance between the two points is 4 units

Enter coordinates of the first point
1 1

Enter coordinates of the second point
6 7

Distance between the two points is 7.81025 units

Chapter 11

Bitwise Operations

Write a program that reads an unsigned integer and then perform the following operations using user defined functions

- i *fnSetBit* - sets a specified bit to 1
- ii *fnMaskBit* - masks a specified bit to 0

C Code

```
/******  
*File           : B11BitMask.c  
*Descriptio     : program to illustrate bitwise operations  
*Author         : Prabodh C P  
*Compiler       : gcc compiler, Ubuntu 14.04  
*Date          : 18 Novemeber 2014  
*****/  
#include <stdio.h>  
#include <stdlib.h>  
  
unsigned short fnSetBit(unsigned short, unsigned short);  
unsigned short fnMaskBit(unsigned short, unsigned short);  
  
int main(void)  
{  
    unsigned short uNum,uPos;  
  
    printf("\nEnter a value\n");  
    scanf("%hu",&uNum);  
  
    printf("\nEnter bit uPosition (1-16) to set to one\n");  
    scanf("%hu",&uPos);  
  
    uNum = fnSetBit(uNum, uPos);  
  
    printf("\nValue after setting the specified bit = %hu\n",uNum);  
  
    printf("\nEnter a value\n");  
    scanf("%hu",&uNum);  
  
    printf("\nEnter bit uPosition (1-16) to mask to zero\n");  
    scanf("%hu",&uPos);  
  
    uNum = fnMaskBit(uNum, uPos);  
  
    printf("\nValue after masking the specified bit = %hu\n",uNum);  
  
    return 0;  
}
```

```
}  
  
unsigned short fnSetBit(unsigned short uVal, unsigned short uPos)  
{  
    unsigned short uMask=1;  
    uMask = uMask << (uPos-1);  
    uVal = uVal | uMask;  
    return uVal;  
}  
unsigned short fnMaskBit(unsigned short uVal, unsigned short uPos)  
{  
    unsigned short uMask=1;  
    uMask = uMask << (uPos-1);  
    uMask = ~uMask;  
    uVal = uVal & uMask;  
    return uVal;  
}
```

Output

Enter a value

56

Enter bit uPosition (1-16) to set to one

3

Value after setting the specified bit = 60

Enter a value

63

Enter bit uPosition (1-16) to mask to zero

3

Value after masking the specified bit = 59