APPENDIX

Solutions to the Exercises

NOTE

The solution programs given here may not be the unique solutions to the exercise question. Readers can try to write the solution programs using their own logic too.

Chapter 2

Fundamentals of C

Q.2.1

```c
#include<stdio.h>
void main()
{
    printf("I asked,\"How are you?\"");
    printf("\nShe replied,\"I am fine\"");
}
```

Q.2.2

```c
#include<stdio.h>
void main()
{
    int a,b,temp;
    printf("Enter a: ");
    scanf("%d",&a);          /*Read a*/
    printf("Enter b: ");
    scanf("%d",&b);          /*Read b*/
    temp=a;                  /*assign a and temp*/
    a=b;                     /*assign b to a*/
    b=temp;                  /*assign temp to b*/
    printf("After swapping, a=%d, b=%d",a,b);
}
```

Q.2.3

```c
#include<stdio.h>
void main()
{
    int a,b;
    printf("Enter a: ");
    scanf("%d",&a);          /*Read a*/
    printf("Enter b: ");
    scanf("%d",&b);          /*Read b*/
    a=a+b;
    b=a-b;
    a=a-b;
    printf("After swapping, a=%d, b=%d",a,b);
}
```

Q.2.5

```c
#include<stdio.h>
void main()
{
    unsigned p,n;
    float r,si;
    printf("Enter p :");
    scanf("%u",&p);         /*Read principal amount*/
    printf("Enter n :");
    scanf("%u",&n);         /*Read no. of years*/
    printf("Enter r :");
    scanf("%f",&r);         /*Read rate of interest*/
    si=p*n*r/100;           /*Calculate simple interest*/
    printf("Simple interest=%f",si);
}
```

Q.2.6

```c
#include<stdio.h>
#include<math.h>
void main()
{
    float x1,y1,x2,y2,dist;
    printf("Enter coordinates of first point: ");
    scanf("%f%f",&x1,&y1);
    printf("Enter coordinates of second point: ");
    scanf("%f%f",&x2,&y2);
    dist=sqrt((x2-x1)*(x2-x1)+(y2-y1)*(y2-y1));
    printf("Euclidean distance=%f",dist);
}
```

Q.2.7

```c
#include<stdio.h>
void main()
{
    float altitude, bp;
    printf("Enter altitude: ");
    scanf("%f",&altitude);     /*Read altitude*/
    bp=100-(altitude/1100);    /*Calculate boiling point*/
    printf("Boiling point=%f",bp);
}
```

Chapter 3

Decisions

Q.3.1

```c
#include<stdio.h>
void main()
{
    int n;
    printf("Enter a number: ");
    scanf("%d",&n);
    if(n%2==0)      /*If divisible by 2*/
        printf("Number is even");
    else
        printf("Number is odd");
}
```

Q.3.2.i

```c
#include<stdio.h>
void main()
{
    unsigned m,n;
    printf("Enter two numbers, m and n: ");
    scanf("%d%d",&m,&n);
    if(m%n==0)    /*Check if m is multiple of n*/
        printf("m is multiple of n");
    else
    {
        if(n%m==0)    /*Check if n is multiple of m*/
            printf("n is multiple of m");
        else
            printf("No multiples");
    }
}
```

Q.3.2.ii

A.6 Appendix

```c
#include<stdio.h>
void main()
{
    unsigned m,n;
    printf("Enter two numbers, m and n: ");
    scanf("%d%d",&m,&n);
    if(m%n==0 || n%m==0)
        printf("A number is multiple of other");
    else
        printf("No multiples");
}
```

Q.3.2.iii

```c
#include<stdio.h>
void main()
{
    unsigned m,n;
    printf("Enter two numbers, m and n: ");
    scanf("%d%d",&m,&n);
    printf(m%n==0||n%m==0?"A number is multiple of other":"No
        multiples");
}
```

Q.3.3

```c
#include<stdio.h>
void main()
{
    float a,b,c;
    printf("Enter three sides of triangle: ");
    scanf("%f%f%f",&a,&b,&c);
    if(a+b>c && a+c>b && b+c>a)    /*Check validity- whether
        addition of any two sides is greater than third side
        */
    {
        printf("It is valid ");
        if(a==b && b==c && a==c)    /*Check whether all sides
            are same*/
            printf("equilateral triangle");
        else
        {
            if(a==b || b==c || a==c)   /*Check whether any two
                sides are same*/
                printf("isosceles triangle");
            else
                printf("scalene triangle");
        }
    }
    else
```

```
        printf("It is not a valid triangle");
}
```

Q.3.4

```
#include<stdio.h>
void main()
{
    char sex;
    int age;
    printf("Enter sex (m/f): ");
    scanf("%c",&sex);
    printf("Enter age: ");
    scanf("%d",&age);
    if((sex=='m' && age>=21) || (sex=='f' && age>=18))
        printf("Eligible to get married");
    else
        printf("Not eligible to get married");
}

/*The same program can be written without using logical
    operators as follows*/
/*
#include<stdio.h>
void main()
{
    char sex;
    int age;
    printf("Enter sex (m/f): ");
    scanf("%c",&sex);
    printf("Enter age: ");
    scanf("%d",&age);
    if((sex=='m')
    {
        if(age>=21)
            printf("Eligible to get married");
        else
            printf("Not eligible to get married");
    }
    else
    {
        if(age>=18)
            printf("Eligible to get married");
        else
            printf("Not eligible to get married");
    }
}
*/
```

Chapter 4

Loops

Q.4.1

```c
#include<stdio.h>
void main()
{
    int i,sum;
    sum=0;                  /*Initialize sum to zero*/
    for(i=2;i<=200;i++)     /*Iterate i from 2 to 200*/
    {
        if(i%7==0)          /*Check if i is divisible by 7*/
            sum=sum+i;      /*then add i to sum*/
    }
    printf("Addition=%d",sum);  /*Display cumulated sum*/
}
```

Q.4.2

```c
#include<stdio.h>
void main()
{
   unsigned n, rev;
   printf("Enter a positive number: ");
   scanf("%u",&n);
   rev=0;      /*initialize rev to zero*/
   while(n!=0)
   {
      rev = rev*10 + n%10; /*add last digit to rev*10 and
          assign to rev*/
      n = n/10;   /*exclude last digit from n*/
   }
   printf("Reverse of a number = %u",rev);
}
```

Q.4.3

```c
#include<stdio.h>
void main()
{
    unsigned n, rev, temp;
    printf("Enter a positive number: ");
    scanf("%u",&n);
    temp=n;  /*Assign n to temp for future use, as n is going
        to change*/
    rev=0;
    while(n!=0)
    {
        rev = rev*10 + n%10;
        n = n/10;
    }
    if(rev==temp)    /*Check whether given number & its reverse
         are same*/
        printf("Number is Palindrome");
    else
        printf("Number is not Palindrome");
}
```

Q.4.4

```c
#include<stdio.h>
void main()
{
    int i,j;
    for(i=2;i<=20;i++)     /*For i from 2 to 20*/
    {
        for(j=1;j<=10;j++)     /*For j from 1 to 10*/
            printf("%4d",i*j);  /*For each i display all i*j in
                one line*/
        printf("\n");            /*Take cursor to new line*/
    }
}
```

Q.4.5

```c
#include<stdio.h>
void main()
{
    unsigned m,n,r;
    printf("Enter two posotive numbers: ");
    scanf("%u%u",&m,&n);
    while(m%n != 0)
    {
        r=m%n;
        m=n;
        n=r;
    }
    printf("GCD=%u",n);
}
```

Q.4.6

```c
#include<stdio.h>
void main()
{
    int n, guess, tries;
    printf("Think of a number between 1 to 100 and enter: ");
    scanf("%d",&n);
    printf("\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n
        \n\n\n");
    printf("Now ask your friend to guess the number");
    tries=0;
    do
    {
        tries++;
        printf("\nEnter guess: ");
        scanf("%d",&guess);
        if(guess>n)
            printf("Your guess is larger than the number");
        if(guess<n)
            printf("Your guess is smaller than the number");
    }while(guess!=n);
    printf("Correct!!! You took %d tries", tries);
}
```

Q.4.7

```c
#include<stdio.h>
void main()
{
    int rows,columns,r,c;
    printf("Enter no. of rows and columns: ");
    scanf("%d%d",&rows,&columns); /*Read no. of rows and
        columns*/
    for(r=1;r<=rows;r++)    /*Iterate through all rows*/
    {
        for(c=1;c<=columns;c++)  /*For each row, iterate
            through all columns*/
        {
            if((r+c)%2==0)    /*if r+c is even, display 1*/
                printf("1");
            else
                printf("0"); /*else display 0*/
        }
        printf("\n"); /*Go to new line for next row*/
    }
}
```

Q.4.8.i

```
#include<stdio.h>
void main()
{
    int n,line,i,space;
    printf("Enter no. of lines: ");
    scanf("%d",&n); /*Read no. of lines*/
    for(line=1;line<=n;line++)    /*Iterate through all lines
        */
    {
        for(space=1;space<=n-line;space++)   /*display n-line
            spaces*/
            printf(" ");
        for(i=line;i>=1;i--)    /*display line downto 1*/
            printf("%d",i);
        for(i=1;i<=line;i++)    /*display characters with
            ASCII value 64+i*/
            printf("%c",64+i);
        printf("\n"); /*Go to next line*/
    }
}
```

Q.4.8.ii

```
#include<stdio.h>
void main()
{
    int n,line,i,x;
    printf("Enter no. of lines: ");
    scanf("%d",&n); /*Read no. of lines*/
    x=1;      /*Initialize to first number to be displayedx*/
    for(line=1;line<=n;line++)    /*Iterate through all lines
        */
    {
        for(i=1;i<=line;i++)      /*display line no. of
            numbers(incremented x) in each line*/
            printf("%3d",x++);
        printf("\n"); /*Go to next line*/
    }
}
```

Q.4.9

```
#include<stdio.h>
void main()
{
    unsigned n,i,f;
    printf("Enter n: ");
    scanf("%u",&n); /*Read n*/
    f=1;      /*Initialize f=1 for further iterative
        multiplication*/
    for(i=1;i<=n;i++)     /*Iterate i from 1 to n*/
        f=f*i;            /*Calculate iterative multiplication
            1*2*3...n*/
    printf("Factorial=%u",f);     /*Display factorial*/
}
```

Q.4.10

```
#include<stdio.h>
void main()
{
    int i;
    float x,term,sum;
    printf("Enter x: ");
    scanf("%f", &x);
    term=x; /*initialize term=first term*/
    sum=0;  /*initialize sum to zero*/
    for(i=1; (term>=0?term:-term)>1e-3; i++)   /*iterate till |
        term|>1e-4 */
    {
        sum = sum + term;   /*update sum by adding term*/
        term = -term*x*x/(2*i*(2*i+1));     /*update term using
            previous term*/
    }
    printf("sin(x) = %f", sum);   /*display final sum*/
}
```

Chapter 5

The switch control instruction

Q.5.1

```c
#include<stdio.h>
void main()
{
    int n;
    printf("Enter n: ");
    scanf("%d",&n);
    switch(n%2)  /* Switch on remainder n%2 */
    {
        case 0: printf("It is even");
                break;
        case 1:
        case -1:printf("It is odd");
    }
}
```

Q.5.2

```c
#include<stdio.h>
void main()
{
    float a,b;
    int choice;
    printf("Enter two numbers : ");
    scanf("%f%f",&a,&b);
    printf("\n1: Addition\n2:Subtraction\n3:Multiplication\n4
        :Division");
    printf("\nEnter your choice: ");
    scanf("%d",&choice);
    switch(choice)
    {
          case 1:  printf("a+b=%f",a+b);
                   break;
          case 2:  printf("a-b=%f",a-b);
                   break;
          case 3:  printf("a*b=%f",a*b);
                   break;
          case 4:  printf("a/b=%f",a/b);
                   break;
          default: printf("Invalid choice");
     }
}
```

Chapter 6

Functions

Q.6.1

```c
#include<stdio.h>

void rangeDisplay(int,int);  /*Function prototype declaration
    */

void main()
{
    rangeDisplay(2,9);         /*Function calls*/
    rangeDisplay(10,25);
    rangeDisplay(-5,5);
}

void rangeDisplay(int a, int b)    /*Function definition*/
{
    int i;
    for(i=a; i<=b; i++)  /*For i= a to b*/
        printf("%d ",i); /*display i*/
    printf("\n");
}
```

Q.6.2

```c
#include<stdio.h>

float calculateCurrent(float,float);   /*Function prototype
    declaration*/

void main()
{
    float R,V;
    printf("Enter value of resistance in Ohms: ");
    scanf("%f",&R);
    printf("Enter voltage applied in Volts: ");
    scanf("%f",&V);

    printf("Current = %f Amperes",calculateCurrent(V,R)); /*
        Function call*/
}

float calculateCurrent(float V,float R)    /*Function
    definition*/
{
    return V/R;
}
```

Q.6.3

```
#include<stdio.h>

char changeCase(char);  /*Function prototype declaration*/

void main()
{
    char ch;
    printf("Enter a character: ");
    scanf("%c",&ch);
    printf("Case changed character=%c",changeCase(ch)); /*
        Function call*/
}

char changeCase(char ch)    /*Function definition*/
{
    if(ch>=65&&ch<=90)        /*If uppercase, return
        corresponding lowercase*/
        return ch+32;
    else
    {
        if(ch>=97&&ch<=122)  /*If lowercase, return
            corresponding uppercase*/
            return ch-32;
        else                  /*If non-alphabet, returm the
            same character*/
            return ch;
    }
}
```

Q.6.4

```
#include<stdio.h>

unsigned fact(unsigned);  /*Function prototype declaration*/

void main()
{
    unsigned n,r,nCr;
    printf("Enter n: ");
    scanf("%u",&n);
    printf("Enter r: ");
    scanf("%u",&r);
    nCr = fact(n)/(fact(n-r)*fact(r));   /*Calculate Bianomial
          coefficient, nCr=n!/((n-r)!.r!)*/
    printf("Bianomial coefficient, nCr=%u",nCr); /*Display
        nCr*/
}

unsigned fact(unsigned n)    /*Function definition*/
{
    if(n==0)
        return 1;
    else
        return n*fact(n-1);
}
```

Q.6.5

```c
#include<stdio.h>

unsigned product(unsigned,unsigned);   /*Function prototype declaration*/

void main()
{
    unsigned m,n;
    printf("Enter m: ");
    scanf("%u",&m);
    printf("Enter n: ");
    scanf("%u",&n);
    printf("m*n=%u",product(m,n));
}

unsigned product(unsigned m,unsigned n)    /*Function definition*/
{
    if(n==0)       /*If n==0, return 0*/
        return 0;
    else           /*otherwise return recursive definition*/
        return m+product(m,n-1);
}
```

Q.6.6

```c
#include<stdio.h>

int isEligible(char,int);   /*Function prototype declaration*/

void main()
{
    char sex;
    int age;
    printf("Enter sex (m/f): ");
    scanf("%c",&sex);
    printf("Enter age: ");
    scanf("%d",&age);
    if(isEligible(sex,age))
        printf("Eligible to get married");
    else
        printf("Not eligible to get married");
}

int isEligible(char sex,int age)    /*Function definition*/
{
    if((sex=='m' && age>=21) || (sex=='f' && age>=18))
        return 1;
    else
        return 0;
}
```

Q.6.7

```
#include<stdio.h>
#include<math.h>
float euclidean(float,float,float,float);   /*Function
    prototype declaration*/

void main()
{
    float x1,y1,x2,y2;
    printf("Enter coordinates of first point: ");
    scanf("%f%f",&x1,&y1);
    printf("Enter coordinates of second point: ");
    scanf("%f%f",&x2,&y2);
    printf("Euclidean distance=%f",euclidean(x1,y1,x2,y2));
          /*Function call and display*/
}

float euclidean(float x1, float y1, float x2, float y2)   /*
    Function definition*/
{
    return sqrt((x2-x1)*(x2-x1)+(y2-y1)*(y2-y1));
}
```

Chapter 7

Preprocessor directives

Q.7.1

```c
#include<stdio.h>
#define PI 3.14
#define circleArea(r) (PI*r*r)
#define surfaceArea(r,h) (2*PI*r*h+2*circleArea(r))
#define volume(r,h) (circleArea(r)*h)

void main()
{
    float r,h;
    printf("Enter radius & height of a cylinder: ");
    scanf("%f%f",&r,&h);
    printf("Base area=%f",circleArea(r));
    printf("\nSurface area=%f",surfaceArea(r,h));
    printf("\nVolume=%f",volume(r,h));
}
```

Q.7.2

```c
#include<stdio.h>
#include "areas.h"   /*Include header file containing the
    function definitions*/

void main()
{
    float l=2.5,w=3.2,r=5.5,a=3.2,b=4.2,c=5.3;
    printf("Area of rectangle=%f",rectArea(l,w));
    printf("\nArea of circle=%f",circleArea(r));
    printf("\nArea of triangle=%f",triangleArea(a,b,c));
}
```

Chapter 8

Pointers

Q.8.1

```
#include<stdio.h>

void swap(int*,int*);    /*Function prototype declaration*/

void main()
{
    int a=55,b=77;
    printf("Before swapping, a=%d, b=%d",a,b);
    swap(&a,&b);         /*Pass addresses of a and b as
        parameters*/
    printf("\nAfter swapping, a=%d, b=%d",a,b);
}

void swap(int *p, int *q)     /*Function definition*/
{
    int temp;
    temp = *p;      /*Assign value at address p to temp*/
    *p=*q;          /*Assign value at address q to value at
        address p*/
    *q=temp;        /*Assign temp to value at address q*/
}
```

Chapter 9

Arrays

Q.9.1

```c
#include<stdio.h>
#define N 15
void main()
{
    float x[N],sum,mean,variance;
    int i;
    printf("Enter %d values of array: ",N);
    for(i=0;i<=N-1;i++)              /*Read entire array*/
        scanf("%f",&x[i]);

    sum=0;
    for(i=0;i<=N-1;i++)          /*Iterative evaluation of sum
        of all elements*/
        sum=sum+x[i];            /*Evaluate mean*/
    mean=sum/N;

    sum=0;
    for(i=0;i<=N-1;i++)         /*Iterative evaluation of sum
        of (x[i]-mean)^2 for all elements*/
        sum=sum+(x[i]-mean)*(x[i]-mean);
    variance=sum/N;             /*Evaluate variance*/

    printf("Mean=%f, and Variance=%f",mean,variance);
}
```

Q.9.2

```c
#include<stdio.h>
#define M 4
#define N 5
void main()
{
    int A[M][N],rowSum[M],columnSum[N],i,j;
    printf("Enter %dx%d array: \n",M,N);
    for(i=0;i<=M-1;i++)              /*Read the matrix*/
        for(j=0;j<=N-1;j++)
            scanf("%d",&A[i][j]);

    for(i=0;i<=M-1;i++)              /*Iterate through all rows
        */
    {
        rowSum[i]=0;
        for(j=0;j<=N-1;j++)          /*Calculate sum of i^th row
            iteratively*/
            rowSum[i]=rowSum[i]+A[i][j];
    }

    for(j=0;j<=N-1;j++)              /*Iterate through all
        columns*/
    {
        columnSum[j]=0;
        for(i=0;i<=M-1;i++)          /*Calculate sum of j^th
            column iteratively*/
            columnSum[j]=columnSum[j]+A[i][j];
    }

    printf("Row wise sums: ");       /*Display 1-D array
        containing row-wise sums*/
    for(i=0;i<=M-1;i++)
        printf("%d ",rowSum[i]);

    printf("\nColumn wise sums: ");  /*Display 1-D array
        containing row-wise sums*/
    for(j=0;j<=N-1;j++)
        printf("%d ",columnSum[j]);
}
```

Q.9.3

```
#include<stdio.h>
#define N 5
void main()
{
    int a[N],i,j,temp;
    printf("Enter array elements: ");
    for(i=0; i<=N-1; i++)
        scanf("%d",&a[i]);
    for(i=0; i<=N-2; i++)    /*selection sort algorithm*/
    {
       for(j=i+1; j<=N-1; j++)
       {
           if(a[i]<a[j])     /*if a[i]<a[j], then- */
           {
               temp=a[i];    /*swap a[i] and a[j]*/
```

```
                    a[i]=a[j];
                    a[j]=temp;
                }
            }
    }
    printf("Sorted array is: ");
    for(i=0; i<=N-1; i++)
        printf("%d  ",a[i]);
}
```

Q.9.4

```
#include<stdio.h>
#define N 4
int secondaryTrace(int [][N]);    /*Function declaration*/
void main()
{
    int A[N][N], i, j;
    printf("Enter a 4X4 matrix:\n");
    for(i=0; i<=N-1; i++)
        for(j=0; j<=N-1; j++)
            scanf("%d",&A[i][j]);
    printf("Trace of matrix = %d",secondaryTrace(A));    /*
        Function call*/
}

int secondaryTrace(int A[][N])     /*Function definition*/
{
    int i, j, sum=0;
    for(i=0; i<=N-1; i++)
       sum = sum + A[i][N-1-i]; /*Add all secondary diagonal
            elements*/
    return sum;
}
```

Q.9.5

```
#include<stdio.h>
#define N 4
int isSymmetric(int [][N]);    /*Function declaration*/
void main()
{
    int A[N][N], i, j;
    printf("Enter a 4X4 matrix:\n");
    for(i=0; i<=N-1; i++)
        for(j=0; j<=N-1; j++)
            scanf("%d",&A[i][j]);
    if(isSymmetric(A))           /*Function call*/
        printf("The matrix is symmetric");
    else
        printf("The matrix is not symmetric");
}

int isSymmetric(int A[][N])     /*Function definition*/
{
    int i,j;
    for(i=1;i<=N-1;i++)  /*Iterate through all elements below
          the main diagonal*/
        for(j=0;j<i;j++)
            if(A[i][j]!=A[j][i])    /*If any element below the
                  diagonal is not matching with -*/
                return 0;           /*the corresponding
                      element above the diagonal, then return 0
                      */
    return 1;       /*If no mismatching found, return 1*/
}
```

Q.9.6

```
#include<stdio.h>
#define M 3
#define N 4
void main()
{
    int A[M][N],B[N][M],i,j;
    printf("Enter a %dX%d matrix:\n",M,N);
    for(i=0; i<=M-1; i++)       /*Read matrix*/
        for(j=0; j<=N-1; j++)
            scanf("%d",&A[i][j]);

    for(j=0;j<=N-1;j++)
        for(i=0;i<=M-1;i++)
            B[j][i]=A[i][j];    /*Assign each element Aij to
                 Bji*/

    printf("Transposed matrix is: \n");
    for(j=0;j<=N-1;j++)          /*Display transposed matrix B
         */
    {
        for(i=0;i<=M-1;i++)
            printf("%d ",B[j][i]);
        printf("\n");;
    }
}
```

Q.9.7

```
#include<stdio.h>
#define M 3
#define W 2
#define N 4

```

```
void main()
{
    int A[M][W],B[W][N],C[M][N],i,j,k;   /*Declare matrices of
        matching dimensions for C=A*B, W is common dimension
        */
    printf("Enter matrix A of dimensions %dX%d matrix:\n",M,W
        );
    for(i=0; i<=M-1; i++)        /*Read matrix A*/
        for(j=0; j<=W-1; j++)
           scanf("%d",&A[i][j]);

    printf("Enter matrix B of dimensions %dX%d matrix:\n",W,N
        );
    for(i=0; i<=W-1; i++)        /*Read matrix B*/
        for(j=0; j<=N-1; j++)
           scanf("%d",&B[i][j]);

     for(i=0; i<=M-1; i++)        /*Perform multiplication A*B
        & assign to C*/
    {
        for(j=0; j<=N-1; j++)
        {
            C[i][j]=0;
            for(k=0;k<=W-1;k++)
                C[i][j]=C[i][j]+A[i][k]*B[k][j];
        }
    }

    printf("Matrix C=A*B is: \n");
    for(i=0;i<=M-1;i++)           /*Display matrix C*/
    {
        for(j=0;j<=N-1;j++)
             printf("%4d",C[i][j]);
        printf("\n");;
    }
}
```

Q.9.8

```
#include<stdio.h>
#define M 50        /*Define maximum allowed dimension*/
int det(int [][M],int);     /*Function declaration*/
void main()
{
    int A[M][M],i,j,k,m;      /*Declare matrix of maximum
        allowed dimensions MxM*/
    printf("Enter length of square matrix: ");   /*Read actual
        dimension of matrix mxm*/
    scanf("%d",&m);
    printf("Enter matrix A of mensions %dX%d matrix:\n",m,m);
         /*Read the natrix*/
    for(i=0; i<=m-1; i++)          /*Read matrix A*/
        for(j=0; j<=m-1; j++)
            scanf("%d",&A[i][j]);

    printf("Determinant, |A|=%d", det(A,m));     /*Function
        call to calculate and display determinant*/
}

int det(int A[][M],int m)
{
    int i,j,k,sign,sum,B[M][M];
    if(m==2)                  /*For 2x2 matrix, calculate
        determinant*/
        return A[0][0]*A[1][1]-A[0][1]*A[1][0];
    else                      /*else calculate determinant using
        recursive definition*/
    {
        sum=0;
        sign=1;
        for(k=0;k<=m-1;k++)
        {
            for(i=1;i<=m-1;i++)      /*calculate minor submatrix
                B of dimension m-1 x m-1 */
            {
                for(j=0;j<=m-1;j++)
                {
                    if(j<k)
                        B[i-1][j]=A[i][j];
                    if(j>k)
                        B[i-1][j-1]=A[i][j];
                }
            }
            sum=sum+sign*A[0][k]*det(B,m-1);
            sign=-sign;
        }
        return sum;
    }
}
```

Chapter 10

String: an array of characters

Q.10.1

```
#include<stdio.h>
void main()
{
    char s[30];
    int i;
    printf("Enter a string: ");
    gets(s);
    printf("String characters and its ASCII values:");
    for(i=0;s[i]!=NULL;i++)    /*For all characters*/
        printf("\n%c   %d",s[i],s[i]);   /*Display character &
            its ASCII value*/
}
```

Q.10.2

```
#include<stdio.h>
void main()
{
    char s[30];
    int i, count;
    printf("Enter a string: ");
    gets(s);
    count=0;        /*Initialize the count to zero*/
    for(i=0;s[i]!=NULL;i++)    /*For all characters, check
        whether it is vowel*/
    {
        if(s[i]=='a'||s[i]=='A'||s[i]=='e'||s[i]=='E'||s[i]==
            'i'||s[i]=='I'||s[i]=='o'||s[i]=='O'||s[i]=='u'||
            s[i]=='U')
            count++;            /*If s[i] is vowel, increment
                the count*/
    }
    printf("No. of vowels = %d",count);   /*Display no. of
        vowels*/
}
```

Q.10.3

```
#include<stdio.h>
void main()
{
    char s[30],t;
    int i,length;
    printf("Enter a string: ");
    gets(s);
    length=0;        /*Initialize length to zero*/
    for(i=0;s[i]!=NULL;i++)    /*Find length of string
        iteratively*/
        length++;

    for(i=0;i<length/2;i++) /*Swap first half characters with
         its corresponding mirrored characters*/
    {
        t=s[i];
        s[i]=s[length-1-i];
        s[length-1-i]=t;
    }

    printf("Reversed string = %s",s);   /*Display reversed
        string*/
}
```

Q.10.4

```
#include<stdio.h>
void main()
{
    char s[50];
    int i,no_of_spaces;
    printf("Enter a string: ");
    gets(s);
    no_of_spaces=0;
    for(i=0;s[i]!=NULL;i++)   /*Count no. of spaces*/
    {
        if(s[i]==' ')
            no_of_spaces++;
    }
    printf("No. of words = %d",no_of_spaces+1); /*No. of
        words = no. of spaces+1*/
}
```

Q.10.5

```
#include<stdio.h>
void main()
{
    char s[50];
    int i;
    printf("Enter a string: ");
    gets(s);
    for(i=0;s[i]!=NULL;i++)     /*Iterate through characters
        of first string*/
    {
        if(s[i]==' ')               /*At each space, go to new
            line*/
            printf("\n");
        else
            printf("%c",s[i]); /*else display the character*/
    }
}
```

Q.10.6

```
#include<stdio.h>
void main()
{
    char s[25],t[25],ans[50];
    int i,k;
    printf("Enter first string: ");
    scanf("%s",s);
    printf("Enter second string: ");
    scanf("%s",t);
    k=0;            /*Initialize index of string ans to zero*/
    ans[0]=NULL;    /*Initialize string ans to NULL*/
    for(i=0;s[i]!=NULL;i++) /*Iterate through characters of
        first string*/
    {
        ans[k]=s[i];    /*Go on appending characters of string
            s to string ans*/
        k++;            /*Increment index of string ans*/
    }
    for(i=0;t[i]!=NULL;i++) /*Iterate through characters of
        second string*/
    {
        ans[k]=t[i];     /*Perform similar operations*/
        k++;
    }
    ans[k]=NULL;
    printf("Concatenated string = %s",ans);
}
```

Q.10.7

```
#include<stdio.h>
#include<string.h>
#define N 7       /*Let there are 7 strings in array*/
void main()
{
    char s[N][30];    /*declare array of strings*/
    int i,index;
    printf("Enter %d strings:\n",N);
    for(i=0;i<=N-1;i++)  /*Read all strings*/
        gets(s[i]);
    index=0;             /*Let first string is longest string
        */
    for(i=1;i<=N-1;i++)  /*Iterate from second string onward
        */
    {
        if(strlen(s[i])>strlen(s[index]))    /*If s[i] is
            longer tha s[index], then index=i*/
            index=i;
    }
    printf("Index of longest string=%d",index);
    printf("\nand the string = %s",s[index]);
}
```

Chapter 11

Structures and Unions

Q.11.1

```c
#include<stdio.h>

struct PERSON
{
    char name[30],sex;
    float height,weight;
};

void main()
{
    float bmi;
    struct PERSON p;
    printf("Enter name:");          /*Read details of a person
        */
    gets(p.name);
    printf("Enter sex(m/f): ");
    scanf("%c",&p.sex);
    printf("Enter height: ");
    scanf("%f",&p.height);
    printf("Enter weight: ");
    scanf("%f",&p.weight);
    bmi = p.weight/(p.height*p.height);    /*Calculate BMI*/
    printf("Body-mass-index = %f \n",bmi);
    if((p.sex=='m' && bmi>=18.5 && bmi<=24.9) || (p.sex=='f'
       && bmi>=16.5 && bmi<=22.9))
        printf("BMI is normal");
    else
        printf("BMI is abnormal");
}
```

Q.11.2

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

```

struct DATE
{
    int day,month,year;
};

struct PERSON
{
    char name[30];
    struct DATE dob;      /*date-of-birth is one of the fields
        of PERSON*/
};

void main()
{
    struct PERSON p;
    struct DATE today;
    printf("Your name: ");
    gets(p.name);
    printf("Enter your birth date(dd mm yyyy): ");      /*
        Read p's dob*/
    scanf("%d%d%d",&p.dob.day, &p.dob.month, &p.dob.year);
    printf("Enter today's date(dd mm yyyy): ");
    scanf("%d%d%d",&today.day, &today.month, &today.year);
        /*Read today's date*/
    printf("Your age on today is %d years", today.year-p.dob.
        year); /*Calculate & display the age*/
}
```

Q.11.3

```
#include<stdio.h>
#define N 5              /*For five cricketers*/
struct CRICKETER
{
    char name[30];
    float average;
    int age;
};

void sort(struct CRICKETER[]);    /*Function declaration*/

void main()
{
    struct CRICKETER c[N];     /*Cricketers' array declaration
        */
    int i;
    for(i=0;i<=N-1;i++)
    {
        printf("Enter data of cricketer c[%d]: \n",i);
        printf("Name: ");
        gets(c[i].name);
        printf("Batting average: ");
        scanf("%f",&c[i].average);
        printf("Age: ");
        scanf("%d",&c[i].age);
        fflush(stdin);          /*Clear keyboard buffer*/
    }
    sort(c);        /*Function call to sort the array*/
    printf("Sorted array as per the batting averages:\n");
    for(i=0;i<=N-1;i++)
    {
        printf("\n%s, battiang average=%f, age=%d",c[i].name,
            c[i].average,c[i].age);
    }
}

void sort(struct CRICKETER c[])    /*Function definition for
    selection sort*/
{
    struct CRICKETER temp;
    int i,j;
    for(i=0;i<=N-2;i++)
    {
        for(j=i+1;j<=N-1;j++)
        {
            if(c[i].average<c[j].average)    /*Sort in
                dscending order of batting average*/
            {
                temp=c[i];
                c[i]=c[j];
                c[j]=temp;
            }
        }
    }
}
```

Q.11.4

```
#include<stdio.h>

struct CRICKETER        /*Define the structure*/
{
    char name[30];
    float average;
    int age;
};

void main()
{
    struct CRICKETER *p;    /*declare CRICKETER type pointer p
        */
    p=(struct CRICKETER*)malloc(sizeof(struct CRICKETER)); /*
        Allocate memory for CRICKETER and assign address to p
        */
    printf("Enter cricketer's name: ");  /*Read details*/
    gets(p->name);
    printf("Enter cricketer's batting average: ");
    scanf("%f",&p->average);
    printf("Enter cricketer's age: ");
    scanf("%d",&p->age);
    printf("Cricketer's details: ");    /*Display details*/
    printf("\nName: %s",p->name);
    printf("\nBatting average: %f",p->average);
    printf("\nCricketer's age: %d",p->age);
    free(p);        /*Deallocate allocated memory for p*/
}
```

Chapter 12

File Handling in C

Q.12.1

```
#include<stdio.h>
void main()
{
    FILE *fp;              /*Declare FILE pointer*/
    char ch;
    int count;
    fp = fopen("test.txt","r");     /*Open the file*/
    if(fp==NULL)    /*Check whether problem in opening, exit
        if so*/
    {
        printf("Problem in opening the file");
        exit(0);
    }
    count=0;      /*Initialize count to zero*/
    while(1)
    {
        ch = fgetc(fp);     /*Get character ch from the file*/
        if(ch==EOF)       /*If ch is end-of-fiile, then break
            */
            break;
        if(ch=='A'||ch=='a'||ch=='E'||ch=='e'||ch=='I'||ch=='
            i'||ch=='O'||ch=='o'||ch=='U'||ch=='u')
            count++;     /*Increment count, if ch is a vowel*/
    }
    fclose(fp);
    printf("No of vowels=%d",count);      /*Display count*/
}
```

Q.12.2

```
#include<stdio.h>
void main()
{
    FILE *fin, *fout;
    char s[25];
    fin = fopen("test.txt","r");     /*Open the input file
        test.txt*/
    fout = fopen("outfile.txt","w");
    if(fin==NULL)     /*If problem in opening the input file,
        exit the program*/
    {
        printf("Problem in opening the input file");
        exit(0);
    }

    if(fout==NULL)     /*If problem in opening the output file
        , then close fin & exit the program*/
    {
        fclose(fin);
        printf("Problem in opening the output file");
        exit(0);
    }

    while(fscanf(fin,"%s",s)!=EOF)     /*Use formatted file
        input function to read a string*/
            fprintf(fout,"%s\n",s);          /*write the
                string and new line to the output file*/
    printf("Contents written to outfile.txt");
    fclose(fin);          /*Close the files*/
    fclose(fout);
}
```

Q.12.3

```
#include<stdio.h>
void main()
{
    FILE *fp;
    int i,n;
    fp = fopen("outfile.txt","w");    /*Open the file in write
        mode*/
    if(fp==NULL)     /*If problem in opening the file, exit
        the program*/
    {
        printf("Problem in opening the file");
        exit(0);
    }
    printf("Enter ten numbers to be written to the file: ");
    for(i=1;i<=10;i++)
    {
        scanf("%d",&n);          /*Read a number*/
        fprintf(fp,"%d ",n);  /*Write it to the file using
            formatted file output*/
    }
```

```c
    printf("Numbers written to outfile.txt");
    fclose(fp);             /*Close the file*/
}
```

Q.12.4

```c
#include<stdio.h>
void main()
{
    FILE *fp;
    int n;
    fp = fopen("outfile.txt","r");    /*Open the file in read
        mode*/
    if(fp==NULL)     /*If problem in opening the file, exit
        the program*/
    {
        printf("Problem in opening the file");
        exit(0);
    }
    while(fscanf(fp,"%d",&n)!=EOF)   /*Read the numbers using
        formatted file input*/
        printf("%d ",n);   /*Display n on the screen*/
    fclose(fp);             /*Close the file*/
}
```

Q.12.5

```c
#include<stdio.h>
struct POINT
{
    float x,y;
};
void main()
{
    FILE *fp;
    int i;
    struct POINT p;
    fp = fopen("cluster.dat","wb");   /*Open the file in
        binary write mode*/
    if(fp==NULL)    /*If problem in opening the file, exit the
        program*/
    {
        printf("Problem in opening the file");
        exit(0);
    }
    printf("Enter coordinates of 7 points to be written to
        the file:\n");
    for(i=1;i<=7;i++)
    {
        printf("p%d: ",i);
        scanf("%f%f",&p.x,&p.y);      /*Read p.x and p.y*/
        fwrite(&p,sizeof(p),1,fp);    /*Write it to the file*/
    }
    printf("Seven points written to points.dat");
    fclose(fp);
}
```

Q.12.6

```
#include<stdio.h>
#include<math.h>
struct POINT
{
    float x,y;
};
void main()
{
    FILE *fp;
    int i;
    struct POINT p,pin,nearestPoint;
    float distance, distanceToNearest;
    fp = fopen("cluster.dat","rb");  /*Open the file in
        binary read mode*/
    if(fp==NULL)    /*If problem in opening the file, exit the
         program*/
    {
        printf("Problem in opening the file");
        exit(0);
    }
    printf("Enter coordinates of a point: ");
    scanf("%f%f",&p.x,&p.y);     /*Read coordinates of a point
         p from the keyboard*/
    fread(&nearestPoint,sizeof(nearestPoint),1,fp); /*Let
        first point in file is nearest*/
    distanceToNearest = sqrt((p.x-nearestPoint.x)*(p.x-
        nearestPoint.x)+(p.y-nearestPoint.y)*(p.y-
        nearestPoint.y));
    while(fread(&pin,sizeof(pin),1,fp)==1)     /*For all
        further points in the file*/
    {
        distance=sqrt((p.x-pin.x)*(p.x-pin.x)+(p.y-pin.y)*(p.
            y-pin.y));  /*Find distance from p*/
        if(distance<distanceToNearest)      /*See whether it is
            nearer than earlier point*/
        {
            nearestPoint=pin;              /*If so, the let it
                be the nearest*/
            distanceToNearest=distance;
        }
    }
    printf("Nearest point to (%f,%f) is (%f,%f)",p.x,p.y,
        nearestPoint.x,nearestPoint.y);
```

```c
    printf("\nand Distance=%f",distanceToNearest);
    fclose(fp);
}
```

Q.12.7

```c
#include<stdio.h>
struct POINT
{
    float x,y;
};
void main()
{
    FILE *fp;
    int i,no_of_points,n;
    struct POINT p[100],q;        /*Maximum no. of points to
         deal are 100*/
    fp = fopen("cluster.dat","rb");  /*Open the file in
         binary read mode*/
    if(fp==NULL)    /*If problem in opening the file, exit the
          program*/
    {
        printf("Problem in opening the file");
        exit(0);
    }

    printf("Currently points in the file are:");
    i=0;
    while(fread(&p[i],sizeof(p[i]),1,fp)==1)   /*Load all
         points stored in the file into an array of points p*/
    {
        printf("\n(%f,%f)",p[i].x,p[i].y);
        i++;                                      /*Also count no
             . of points*/
    }

    no_of_points=i;

    printf("Enter index of point to be deleted (begin with 0)
        , n: ");
    scanf("%d",&n);        /*Read index of point to be deleted
        */

    if(n<0 || n>=no_of_points)      /*If index is invalid,
         close fp and exit*/
    {
        printf("Invalid serial number, enter 0<=n<%d",
            no_of_points);
        fclose(fp);
        exit(0);
    }
    fclose(fp);

    fp = fopen("cluster.dat","wb");  /*Reopen the file in binary write mode*/
    for(i=0;i<no_of_points;i++)        /*Write all points from array p to the file*/
    {
        if(i!=n)                          /*except at index n*/
            fwrite(&p[i],sizeof(p[i]),1,fp);
    }
    fclose(fp);           /*Close the file*/

    printf("After deleting a point, the points in the file are:");
    fp = fopen("cluster.dat","rb");  /*Reopen the file again in binary read mode*/
    while(fread(&q,sizeof(q),1,fp)==1)    /*Read each points in point q, and display it*/
         printf("\n(%f,%f)",q.x,q.y);
    fclose(fp);
}
```

Q.12.8

```c
#include<stdio.h>
#include<fcntl.h>
#include<stdlib.h>
void main()
{
    char buffer[4],str[7],filename[20];
    long unsigned size,k;
    int n,i;
    int inhandle, outhandle;
    inhandle=open("rimzim.mp3",O_RDONLY|O_BINARY);
    if(inhandle==-1)
    {
       printf("Cannot open 'rimzim.mp3'");
       exit(0);
    }

    printf("Wait.. size of the file is being calculated");
    size=0;
    while(read(inhandle,buffer,1))  /*Find size of the file in bytes*/
         size++;
    printf("\nSize of the original file=%u bytes",size);
    close(inhandle);    /*Close the file*/

    printf("\nEnter number of files to which the audio file to be split: ");
    scanf("%d",&n);     /*Read no. of parts to which the file
```

```
        to be split*/

    printf("\nWait.. the file is being split..");
    inhandle=open("rimzim.mp3",O_RDONLY|O_BINARY);   /*Reopen
        the file*/
    for(i=1;i<=n;i++)      /*For i=1 to n*/
    {
        itoa(i,str,10);
        strcpy(filename,"part");
        strcat(strcat(filename,str),".mp3");    /*Let file
            name be part<i>.mp3*/
        outhandle=open(filename,O_CREAT|O_WRONLY|O_BINARY);
               /*Create new file with the name filename*/
        for(k=0;k<=size/n;k++)
        {
            if(read(inhandle,buffer,1)==0)    /*Read a byte
                from file and check whether no no more byte
                left for reading*/
            {
                close(inhandle);              /*If so, then
                    close both input and output files*/
                close(outhandle);
                printf("\n%s created",filename);
                exit(0);
            }
            write(outhandle,buffer,1);     /*Copy size/n bytes
                to the file named filename(bye-by-byte)*/
        }
        close(outhandle);   /*Close the file after copying
            size/n bytes to part<i>.dat*/
        printf("\n%s created",filename);
    }
}
```

Q.12.9

```
#include<stdio.h>
#include<fcntl.h>
#include<stdlib.h>
void main()
{
    char buffer[4],str[7],filename[20];
    long unsigned k;
    int n,i;
    int inhandle, outhandle;
    outhandle=open("output.mp3",O_CREAT|O_WRONLY|O_BINARY);
    printf("\nEnter number of files to be joined: ");
    scanf("%d",&n);      /*Read no. of parts to which the file
        to be split*/

    for(i=1;i<=n;i++)      /*For i=1 to n*/
    {
        itoa(i,str,10);
        strcpy(filename,"part");
        strcat(strcat(filename,str),".mp3");     /*i^th file
            name-> part<i>.mp3*/
        inhandle=open(filename,O_RDONLY|O_BINARY);    /*Open i^
            th file part<i>.mp3*/
        while(read(inhandle,buffer,1)>0)    /*Read byte-by-byte
              and write in output.mp3*/
            write(outhandle,buffer,1);
        close(inhandle);
        printf("\n%s joined",filename);
    }
    close(outhandle);
    printf("output.mp3 created");
}
```

Chapter 13

Miscellaneous topics

Q.13.1

```
#include<stdio.h>
void main()
{
    FILE *fp1,*fp2,*fp3;
    int a,b;
    fp1=fopen("first.txt","r");
    if(fp1==NULL)
    {
        printf("Problem in opening 'first.txt'");
        exit(0);
    }
    fp2=fopen("second.txt","r");
    if(fp2==NULL)
    {
        fclose(fp1);
        printf("Problem in opening 'second.txt'");
        exit(0);
    }
    fp3=fopen("result.txt","w");
    if(fp3==NULL)
    {
        fclose(fp1);
        fclose(fp2);
        printf("Problem in creating 'result.txt'");
        exit(0);
    }
    while(fscanf(fp1,"%d",&a)!=EOF && fscanf(fp2,"%d",&b)!=
        EOF)
        fprintf(fp3,"%d ",a+b);
    fclose(fp1);
    fclose(fp2);
    fclose(fp3);
```

```c
    printf("Added all numbers of 'first.txt' with 'second.txt
        ' and written in ''result.txt'");
}
```

Q.13.2

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

enum DAY        /*Enumerated type definition*/
{
    Monday,Tuesday,Wednesday,Thursday,Friday,Saturday,Sunday
};

enum DAY readDay(void);      /*Function declarations*/
void displayDay(enum DAY);

void main()
{
    enum DAY d;
    d = readDay();                /*Read day*/
    printf("Your day is: ");
    displayDay(d);                /*Display day*/
}

enum DAY readDay(void)     /*Function definition for readDay*/
{
    char dayName[20];
    enum DAY d;
    printf("\nEnter a day name: ");
    scanf("%s",dayName);              /*Read day da a string*/
    if(strcmpi(dayName,"Monday")==0)
        return Monday;                /*reurn enum DAY type
            value based on dayName*/
    if(strcmpi(dayName,"Tuesday")==0)
        return Tuesday;
    if(strcmpi(dayName,"Wednesday")==0)
        return Wednesday;
    if(strcmpi(dayName,"Thursday")==0)
        return Thursday;
    if(strcmpi(dayName,"Friday")==0)
        return Friday;
    if(strcmpi(dayName,"Saturday")==0)
        return Saturday;
    if(strcmpi(dayName,"Sunday")==0)
        return Sunday;
    return Sunday;        /*If dayName is not valid day, then
        return Sunday*/
}

void displayDay(enum DAY d)          /*Function definition for
```

```
    displayDay*/
{
    switch(d)        /*switch instruction to display day name
        string for enum DAY value d*/
    {
        case Monday:
                        printf("Monday");    break;
        case Tuesday:
                        printf("Tuesday");   break;
        case Wednesday:
                        printf("Wednesday");   break;
        case Thursday:
                        printf("Thursday");    break;
        case Friday:
                        printf("Friday");    break;
        case Saturday:
                        printf("Saturday");    break;
        case Sunday:
                        printf("Sunday");   break;
        default:
                        printf("Invalid day");
    }
}
```

Q.13.3

```
#include<stdio.h>

unsigned circularShift(unsigned,int);       /*Function
    declarations*/

void main()
{
    printf("circularShift(5,-1)=%u",circularShift(5,-1)); /*
        5=>0000 0000 0000 0101*/
    printf("\ncircularShift(32769,1)=%u",circularShift
        (32769,1)); /*32769=>1000 0000 0000 0001*/
}

unsigned circularShift(unsigned n, int k)      /*Function
    definition*/
{
    int i;
    if(k>=0)     /*For positive k*/
    {
        for(i=1;i<=k;i++) /*Repeat k times*/
        {
            if(n & 32768) /*If MSB=1*/
                n=(n<<1)|1; /*Left shift by one bit and
                    assign 1 to LSB*/
            else
```

```
                n=n<<1;   /*else Just left shift by one bit*/
            }
        }
        else        /*For negative k*/
        {
            for(i=1;i<=-k;i++)      /*Repeat k times*/
            {
                if(n & 1)           /*If MSB=1*/
                    n=(n>>1)|32768;   /*Right shift by one bit and
                        assign 1 to MSB*/
                else
                    n=n>>1;         /*else Just right shift by one
                        bit*/
            }
        }
        return n;   /*Return circularly shifted n*/
}
```