POP Lab Manuel

Principles of programming using C (Visvesvaraya Technological University)

Mission and Vision

Mission

To achieve academic excellence in science, engineering and technology through dedication to duty, innovation in teaching and faith in human values.

To enable our students to develop into outstanding professional with high ethical standards to face the challenges of 21st century.

To provide educational opportunities to the deprived and weaker section of society to uplift their socio-economic status.

Vision

To empower students through wholesome education and enable the students to develop into highly qualified and trained professionals with ethics and emerge as responsible citizen with broad outlook to build a vibrant nation.

To advance the intellectual capacity of the nation and the international community by imparting knowledge to graduates who are globally recognized as innovators, entrepreneurs and competent professionals.

To involve in research activities and be committed to lifelong learning to make positive contributions to society

Program Outcomes	
PO1	**Engineering Knowledge:** Apply knowledge of mathematics, science, engineering fundamentals and an engineering specialization to the solution of complex engineering problems.
PO2	**Problem Analysis: Identify,** formulate, research literature and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences and engineering sciences
PO3	**Design/ Development of Solutions**: Design solutions for complex engineering problems and design system components or processes that meet specified needs with appropriate consideration for public health and safety, cultural, societal and environmental considerations.
PO4	**Conduct investigations of complex problems** using research-based knowledge and research methods including design of experiments, analysis and interpretation of data and synthesis of information to provide valid conclusions.
PO5	**Modern Tool Usage**: Create, select and apply appropriate techniques, resources and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
PO6	**The Engineer and Society**: Apply reasoning informed by contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to professional engineering practice.
PO7	**Environment and Sustainability**: Understand the impact of professional engineering solutions in societal and environmental contexts and demonstrate knowledge of and need for sustainable development.
PO8	**Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of engineering practice.
PO9	**Individual and Team Work**: Function effectively as an individual, and as a member or leader in diverse teams and in multi-disciplinary settings.
PO10	**Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as being able to comprehend and write effective reports and design documentation, make effective presentations and give and receive clear instructions.
PO11	**Life-long Learning**: Recognize the need for and have the preparation and ability to engage in independent and life- long learning in the broadest context of technological change.
PO12	**Project Management and Finance**: Demonstrate knowledge and understanding of engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

Program Specific Outcomes	
PSO1	**Problem-Solving Skills:** An ability to investigate and solve a problem by analysis, interpretation of data, design and implementation through appropriate techniques, tools and skills.
PSO2	**Professional Skills**: An ability to apply algorithmic principles, computing skills and computer science theory in the modelling and design of computer-based systems.
PSO3	**Entrepreneurial Ability:** An ability to apply design, development principles and management skills in the construction of software products of varying complexity to become an entrepreneur

Principles of Programming in C Laboratory (BPOPS103P)

Descriptions:

- The laboratory should be preceded or followed by a tutorial to explain the approach or algorithm being implemented for the problems given.
- Every experiment should have algorithm and flowchart be written before writing the program.
- Code should be traced using minimum two test cases which should be recorded.
- It is preferred to implement using Linux and GCC.

Course Objectives:

1. Elucidate the basic architecture and functionalities of a computer and recognize the hardware parts.
2. Apply programming constructs of C language to solve the real-world problem.
3. Explore user-defined data structures like arrays in implementing solutions to problems like searching and sorting
4. Explore user-defined data structures like structures, unions and pointers in implementing solutions
5. Design and Develop Solutions to problems using modular programming constructs

Laboratory Experiments

1. Simulation of a Simple Calculator.
2. Compute the roots of a quadratic equation by accepting the coefficients. Print appropriate messages.
3. An electricity board charges the following rates for the use of electricity: for the first 200 units 80 paise per unit: for the next 100 units 90 paise per unit: beyond 300 units Rs 1 per unit. All users are charged a minimum of Rs. 100 as meter charge. If the total amount is more than Rs 400, then an additional surcharge of 15% of total amount is charged. Write a program to read the name of the user, number of units consumed and print out the charges.
4. Write a C Program to display the following by reading the number of rows as input

```
      1
    1 2 1
  1 2 3 2 1
1 2 3 4 3 2 1
```

5. Write functions to implement string operations such as compare, concatenate, string length. Convince the parameter passing techniques.
6. Implement structures to read, write and compute average marks and students scoring above and below the average marks for a class of N students.
7. Develop a program using pointers to compute the sum, mean and standard deviation of all elements stored in an array of N real numbers.
8. Write a C program to copy a text file to another, read both input file name and target file name
9. Write functions to implement string operations such as compare, concatenate, string length. Convince the parameter passing techniques.
10. Implement structures to read, write and compute average marks and students scoring above and below the average marks for a class of N students.
11. Develop a program using pointers to compute the sum, mean and standard deviation of all elements stored in an array of N real numbers.
12. Write a C program to copy a text file to another, read both input file name and target file name

Assessment Details (both CIE and SEE)

The weightage of Continuous Internal Evaluation (CIE) is 50% and for Semester End Exam (SEE) is 50%. The minimum passing mark for the CIE is 40% of the maximum marks (20 marks out of 50). The minimum passing mark for the SEE is 35% of the maximum marks (18 marks out of 50). A student shall be deemed to have satisfied the academic requirements and earned the credits allotted to each subject/ course if the student secures not less than 35% (18 Marks out of 50) in the semester-end examination(SEE), and a minimum of 40% (40 marks out of 100) in the sum total of the CIE (Continuous Internal Evaluation) and SEE (Semester End Examination) taken together.

Continuous Internal Evaluation (CIE):

Two Unit Tests each of20 Marks (duration 01 hour)

First test after the completion of 30-40 % of the syllabus

Second test after completion of 80-90% of the syllabus

One Improvement test before the closing of the academic term may be conducted if necessary. However best two tests out of three shall be taken into consideration.

Two assignments each of 10 Marks

The teacher has to plan the assignments and get them completed by the students well before the closing of the term so that marks entry in the examination portal shall be done on time. Formative (Successive) Assessments include Assignments/Quizzes/Seminars/ Course projects/Field surveys/ Case studies/ Hands-on practice (experiments)/Group Discussions/ others. . The Teachers shall choose the types of assignments depending on the requirement of the course and plan to attain the Cos and Poss. (to have a less stressed CIE, the portion of the syllabus should not be common /repeated for any of the methods of the CIE. Each method of CIE should have a different syllabus portion of the course). CIE methods /test question paper is designed to attain the different levels of Bloom's taxonomy as per the outcome defined for the course.

The sum of two tests, two assignments, will be out of 60 marks and will be scaled down to 30 marks CIE for the practical component of the Integrated Course

On completion of every experiment/program in the laboratory, the students shall be evaluated, and marks shall be awarded on the same day. The 15 marks are for conducting the experiment and preparation of the laboratory record, the other 05 marks shall be for the test conducted at the end of the semester.

The CIE marks awarded in the case of the Practical component shall be based on the continuous evaluation of the laboratory report. Each experiment report can be evaluated for 10 marks. Marks of all experiments' write-ups are added and scaled down to 15 marks.

The laboratory test (duration 02/03 hours) at the end of the 14th /15th week of the semester /after completion of all the experiments (whichever is early) shall be conducted for 50 marks and scaled down to 05 marks.

Scaled-down marks of write-up evaluations and tests added will be CIE marks for the laboratory component of IPCC for 20 marks.

Semester End Examination (SEE):

SEE for IC

Theory SEE will be conducted by University as per the scheduled timetable, with common question papers for the course (duration 03 hours)

1. The question paper will have ten questions. Each question is set for 20 marks.
2. There will be 2 questions from each module. Each of the two questions under a module (with a maximum of 3 sub-questions), should have a mix of topics under that module.
3. The students have to answer 5 full questions, selecting one full question from each module.

The theory portion of the Integrated Course shall be for both CIE and SEE, whereas the practical portion will have a CIE component only. Questions mentioned in the SEE paper shall include questions from the practical component).

Passing standard:

- The minimum marks to be secured in CIE to appear for SEE shall be 12 (40% of maximum marks- 30) in the theory component and 08 (40% of maximum marks -20) in the practical component. The laboratory component of the IPCC shall be for CIE only. However, in SEE, the questions from the laboratory component shall be included. The maximum of 04/05 questions to be set from the practical component of IPCC, the total marks of all questions should not be more than 30 marks.

SEE will be conducted for 100 marks and students shall secure 35% of the maximum marks to qualify for the SEE. Marks secured will be scaled down to 50.

Suggested Learning Resources:

Textbooks

1. Computer fundamentals and programming in c, “Reema Thareja”, Oxford University, Second edition, 2017.

Reference Books:

1. E. Balaguruswamy, Programming in ANSI C, 7th Edition, Tata McGraw-Hill. Brian W. Kernighan and Dennis M. Ritchie, The ‘C’ Programming Language, Prentice Hall of India

Weblinks and Video Lectures (e-Resources):

1. elearning.vtu.ac.in/contents/courses/video/BS/15PCD23.html
2. https://nptel.ac.in/courses/106/105/106105171/ MOOC courses can be adopted for more clarity in understanding the topics and verities of problem-solving methods.

https://tinyurl.com/4xmrexre

Laboratory Outcomes:

CO1. Elucidate the basic architecture and functionalities of a computer and recognize the hardware parts.

CO 2. Apply programming constructs of C language to solve the real-world problem

CO 3. Explore user-defined data structures like arrays in implementing solutions to problems like searching and sorting

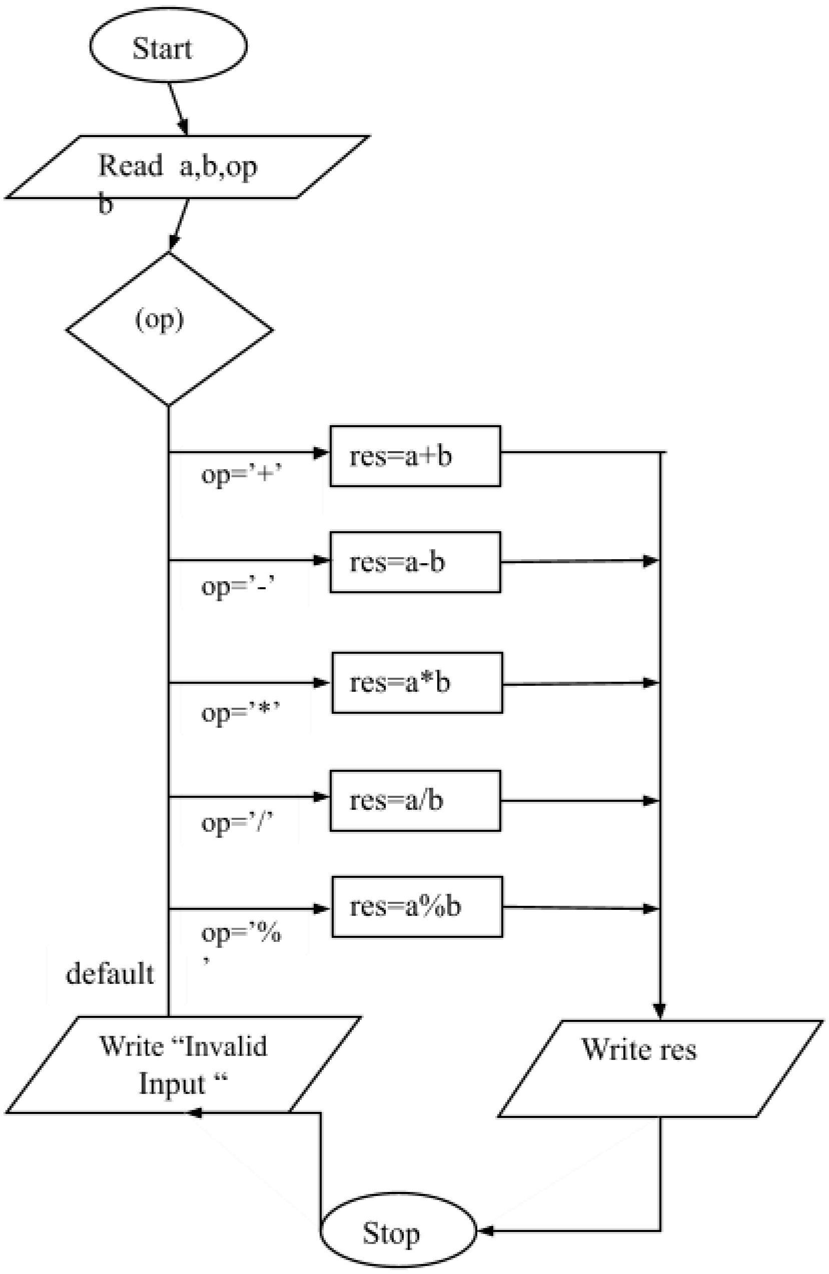
CO 4. Explore user-defined data structures like structures, unions and pointers in implementing solutions

CO5.Design and Develop Solutions to problems using modular programming constructs

Laboratory Programs:

1. Simulation of Simple Calculator

Flowchart



Algorithm:

```
Step 1: Start
Step 2: Read operand1 operator operand
Step 3: Switch for the value of operator

        case '+':  res=a+b;
                   Display the result
                   goto step 5
        case'-':  res=a-b;
                   Display the result
                   goto step 5
        case'*': res=a*b;
                   Display the result
                   goto step 5
        case'/': res=a/b;
                   Display the result
                   goto step 5
Step 4: Display "Invalid operator!"
Step 5: Stop
```

Program:

```c
#include <stdio.h>
void main()
{
  float a, b, res;
  char op;
  printf("Enter arithmetic expression[operand1 operator operand2]:");
  scanf("%f%c%f", &a, &op, &b);
  switch (op)
  {
  case '+':
    res = a + b;
    printf("\nsum=%f", res);
    break;
  case '-':
    res = a - b;
    printf("\nDifference=%f", res);
    break;
  case '*':
    res = a * b;
    printf("\nProduct=%f", res);
    break;
  case '/':
    res = a / b;
    printf("\nQuotient=%f", res);
    break;
  default:
    printf("Invalid operator!");
    break;
  }
}
```

Test Cases:

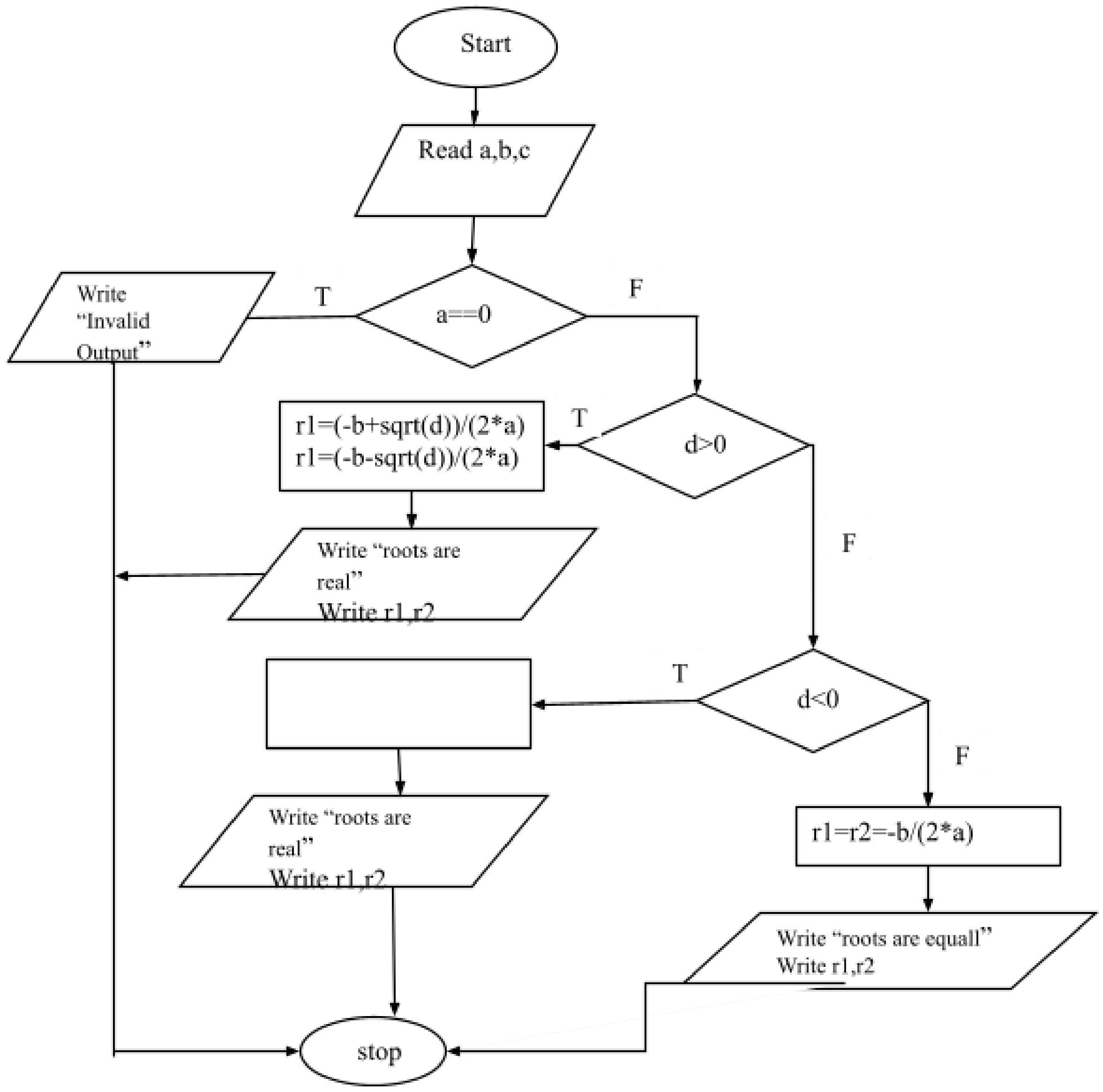
Sl no.	Test Data	Expected Output
1	10+20	Res=30
2	20-15	Res=5
3	20*30	Res=600
4	20/2	Res=10
5	20%6	Res=2
6	20@10	Invalid Operator

Viva Questions:

Sl.NO	Questions
1.	What is algorithm?
2.	What is flowchart?
3.	What is an operator?
4.	What is an operand?
5.	Name arithmetic operators?
6.	What is the syntax of switch statement?

2. **Develop a program to compute the roots of a quadratic equation by accepting the coefficients. Print appropriate messages.**

Flowchart:



Algorithm:

1. Start.
2. Read the coefficients a, b and c.
3. If any of the coefficients is zero display "Invalid Input"
4. Otherwise,
    1. calculate d□b*b-4*a*c
    2. Find r□sqrt(d)
    3. If d is greater than zero
        i. r1□(-b+r)/(2*a)
        ii. r2□(-b-r)/(2*a)
        iii. Display "Roots are real and distinct"
        iv. Display the roots r1 and r2

```
        4.  If d is equal to zero
                i.  r1=r2=-b/(2*a)
               ii.  Display “Roots are real and equal”
              iii.  Display r1 and r2.
        5.  Otherwise
                i.  rp□-b/(2*a)
               ii.  ip□r/(2*a)
              iii.  Display “Roots are imaginary”
               iv.  Display the roots
5.  End
```

Program:

```
#include <stdio.h>
#include <math.h>
void main()
{
        int a,b,c;
        float r1,r2,d,rp,ip,r;
        printf("Enter three coefficients\n");
        scanf("%d%d%d",&a,&b,&c);
        if (a==0)
                printf("Invalid  Input\n");
        else
        {
                d=b*b-4*a*c;
                r=sqrt(fabs(d));
                if (d>0)
                {
                r1=(-b+r)/(2.0*a);
                r2=(-b-r)/(2.0*a);
                printf("Roots are Real and Distinct\n");
                printf("Root1=%f\nRoot2=%f\n",r1,r2);
                }
                else if (d==0)
                {
                r1=r2=-b/(2.0*a);
                printf("Roots are Real and Equal\n");
                printf("Root1=Root2=%f\n",r1);
                }
                else
                {
                rp=-b/(2.0*a);
                ip=r/(2.0*a);
                printf("Roots are Real and Imaginary\n");
                printf("Root1=%f+i%f\nRoot2=%f-i%f\n",rp,ip,rp,ip);
                }
        }
}
```

Test Cases:

Sl no.	Test Data			Expected Output
	A	B	c	
1	1	4	4	Roots are Real and Equal Root1=Root2=-2.000000

2	1	-5	6	Roots are Real and Distinct
				Root1=3.000000
				Root2=2
				.000000

3	2	3	4	Roots are Real and Imaginary
				Root1=-0.750000+i1.198958
				Root2=-0.750000-i1.198958

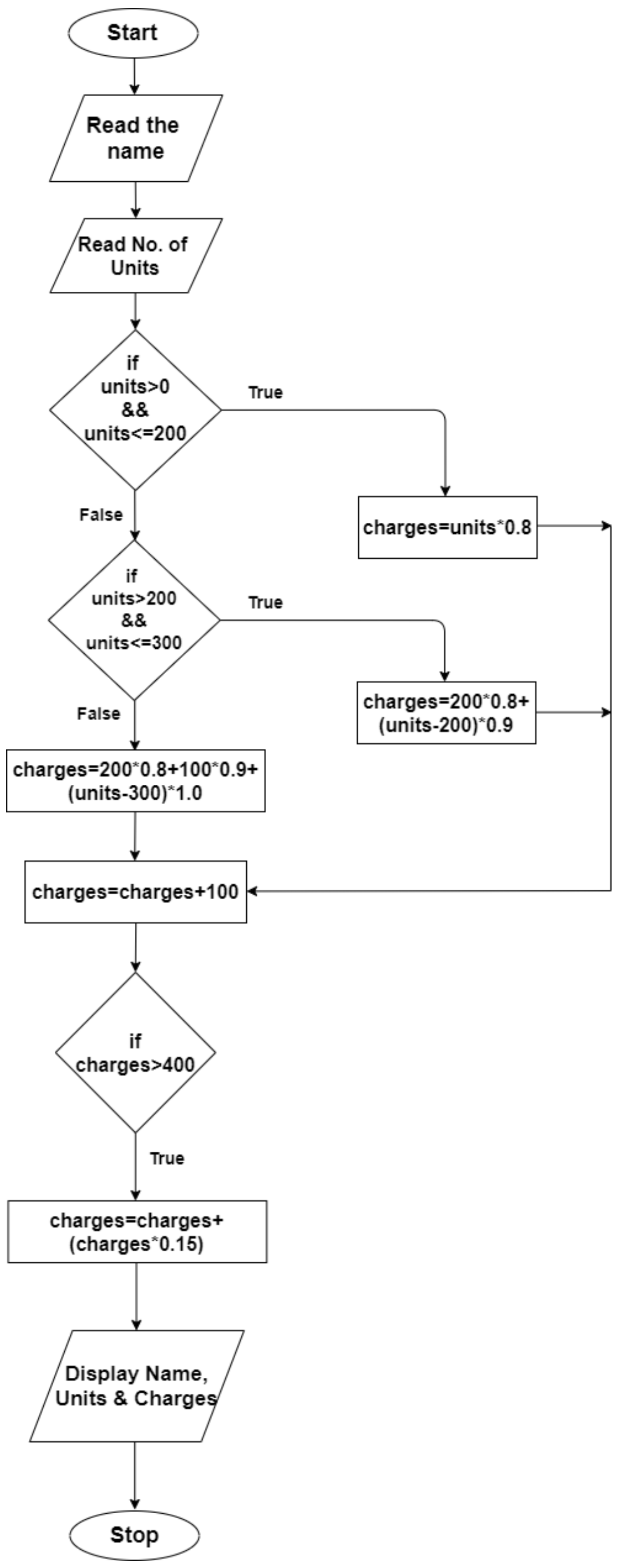
4	1	0	8	
5	0	3	6	

Viva Questions:

Sl.NO	Questions
1.	What is a Quadratic equation?
2.	When are the roots equal distinct and imaginary?
3.	What are input and output functions?
4.	What is the syntax of if and if else condition?
5.	What is sqrt function and fab's function?
6.	What is math.h header?
7.	What is the size of int,float,double,char and void?

3. An electricity board charges the following rates for the use of electricity: for the first 200 units 80 paise per unit: for the next 100 units 90 paise per unit: beyond 300 units Rs 1 per unit. All users are charged a minimum of Rs. 100 as meter charge. If the total amount is more than Rs 400, then an additional surcharge of 15% of total amount is charged. Write a program to read the name of the user, number of units consumed and print out the charges.

Flowchart:


Start
Read the name
Read No. of Units
if units>0 && units<=200
True
charges=units*0.8
False
if units>200 && units<=300
True
charges=200*0.8+ (units-200)*0.9
False
charges=200*0.8+100*0.9+ (units-300)*1.0
charges=charges+100
if charges>400
True
charges=charges+ (charges*0.15)
Display Name, Units & Charges
Stop

Algorithm:

```
Step 1: Start
Step 2: Read Name, Units
Step 3: if (units<=200)
charges=units *0.8;
Step 4: else if (units<=300)
charges=200*0.8+(units-200)*0.9;
else
Step  5: charges=200*0.8+100*0.9+(units-300)*1.0;
Step 6:  charges=charges+100;
Step 7:  if (charges> 400)
charges =charges +(charges*0.15);
Step 8: output name, units, charges
```

Program:

```
include <stdio.h>
void main()
{
        int units;
        float charges;
        char name[20];

        printf("Enter the name \n");
        scanf("%s",name);
        printf("Enter the units consumed\n");
        scanf("%d",&units);

        if (units<=200)
             charges=units  *0.8;

        else if (units<=300)
             charges=200*0.8+(units-200)*0.9;
        else
             charges=200*0.8+100*0.9+(units-300)*1.0;

        charges=charges+100;
        if (charges> 400)
            charges =charges +(charges*0.15);
        printf("name \t units\t charges\t\n");
        printf("%s \t %d\t %f\t\n",name,units,charges);
        return 0;
}
```

Test Case

Sl no.	Test Data	Expected Output
1	Name: Sachin Units: 670	Charges: 828.00
2	Name: Rahul Units: 380	Charges: 494.50
3	Name: Sunil Units: 250	Charges: 305.00
4	Name: Ravi Units: 180	Charges: 244.00

Viva Questions:

Sl.NO	Questions
1.	What is the data type for string?
2.	Write the equation for 200 units consumed.
3.	Write the equation for 300 units consumed.
4.	What is the syntax of if and if else condition?
5.	Write the equation if the amount exceeds Rs. 400.

4) Write a C Program to display the following by reading the number of rows as input

```
      1
    1 2 1
  1 2 3 2 1
1 2 3 4 3 2 1
```

Algorithm

```
1)  Start
2)  Let i be an integer number.
3)  Let j be an integer number.
4)  Let row  be a integer number.
5)  Repeat step 6 to 20 until all value parsed.
6)  Set i = 1 and check i<=row;
7)  Repeat step 8 to 10 until all value parsed
8)  Set j = 1 and check j <= row-i
9)  Print space
10) Then i++
11) Repeat step 12 to 14 until all value parsed
12) Set j = 1 and check j <i
13) Print j.
14) Then j++.
15) Repeat step 16 to 18until all value parsed
16) Set j =i-1 and check j>1
17) Print j
18) Then j--;
19) Print new line
20) Then row++;
21) End
```

Program

```
#include<stdio.h>
int main()
{
    int i,j,row;
    printf("Enter number of rows: ");
    scanf ("%d", &row);
    for(i=1;i<=row;i++)
    {
    for(j=1;j<=row-i;j++)
              {
                printf(" ");
               }
            for(j=1; j<=i; j++)
               {
                printf("%d",  j);
                }
            for(j=i-1;j>=1;j--)
              {
                printf("%d",  j);
              }
     printf("\n");
    }
    return  0;
}
```

Test Case

Sl no.	Test Data	Expected Output
1	2	1 1 2 1

2	4	1
		1 2 1
		1 2 3 2 1
		1 2 3 4 3 2 1

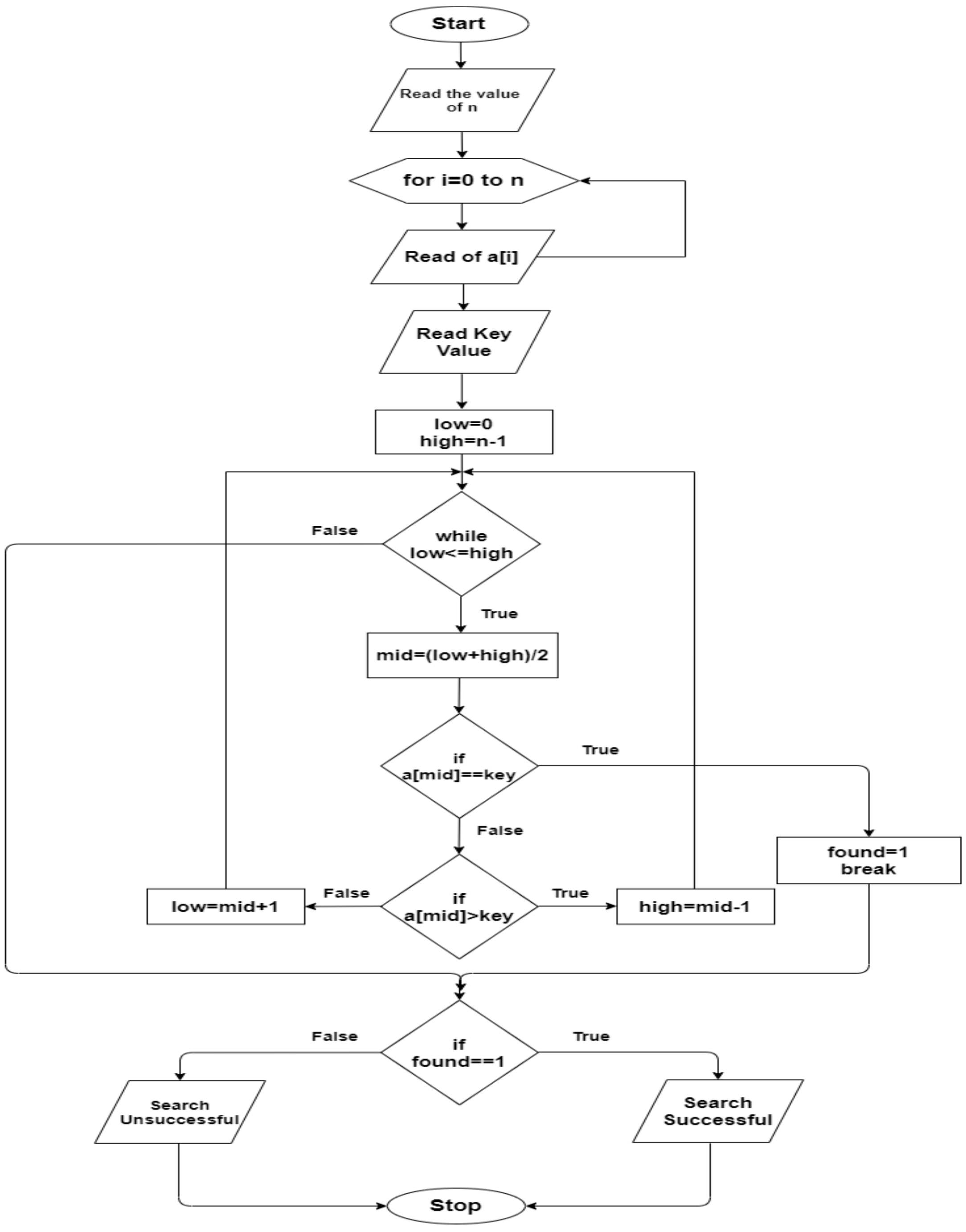
3	6	1
		1 2 1
		1 2 3 2 1
		1 23 4 3 2 1
		1 2 3 4 5 4 3 2 1
		1 2 3 4 5 6 5 4 3 2 1

Viva Questions

Sl. No	Questions
1.	Syntax for for loop
2.	What is nested for loop
3.	Syntax for nested for loop

5 Implement Binary Search on Integers or Name

Flow chart:



Algorithm:

```
Step 1: Start
Step 2: Read the value of n
Step 3: Enter the elements in ascending order one by one using for loop
Step 4: Read the element to be searched
Step 5: Assign low=0 and high=n-1
Step 6: while(low<=high) do the following steps
                mid = (low + high) / 2;
            if (key == a[mid])
                found =1;
    break;
otherwise
    if (key < a[mid])
    high = mid - 1;
    otherwise if (key > a[mid])

    low = mid + 1;
Step 7: if found=1
                Display "Search is successful"
        Otherwise
                Display "Search is not successful"
    Step 8:Stop
```

Program:

```
#include <stdio.h>
void main()
{
    int a[10];
    int i, j, n, key;
    int low, mid, high,found=0;

    printf("Enter the value of num \n");
    scanf("%d", &n);
    printf("Enter the elements one by one \n");
    for (i=0;i<n;i++)
    {
        scanf("%d",&a[i]);
     }
    printf("Enter the element to be searched \n");
    scanf("%d", &key );

   /* Binary searching begins */
    low = 0;
    high = n-1;
    while(low<=high)
    {
            mid = (low + high) / 2;
            if (key == a[mid])
            {
                found =1;
                break;
            }
            if (key < a[mid])
            high = mid - 1;
            else if (key > a[mid])
            low = mid + 1;
    }
    if (found==1)
    {
            printf("SEARCH  SUCCESSFUL  \n");
    }
    else
    {
            printf("SEARCH  UNSUCCESSFUL  \n");
    }
}
```

Test Cases:

Sl no.	Test Data	Expected Output

1	n=5
	Elements: 1 2 3 4 5
	Key= 4
		SEARCH SUCCESSFUL

2	n=6
	Elements: 11 12 13 14 15 16
	Key= 11
		SEARCH SUCCESSFUL

3	n=5
	Elements: 3 4 5 6 7
	Key= 2
		SEARCH UNSUCCESSFUL

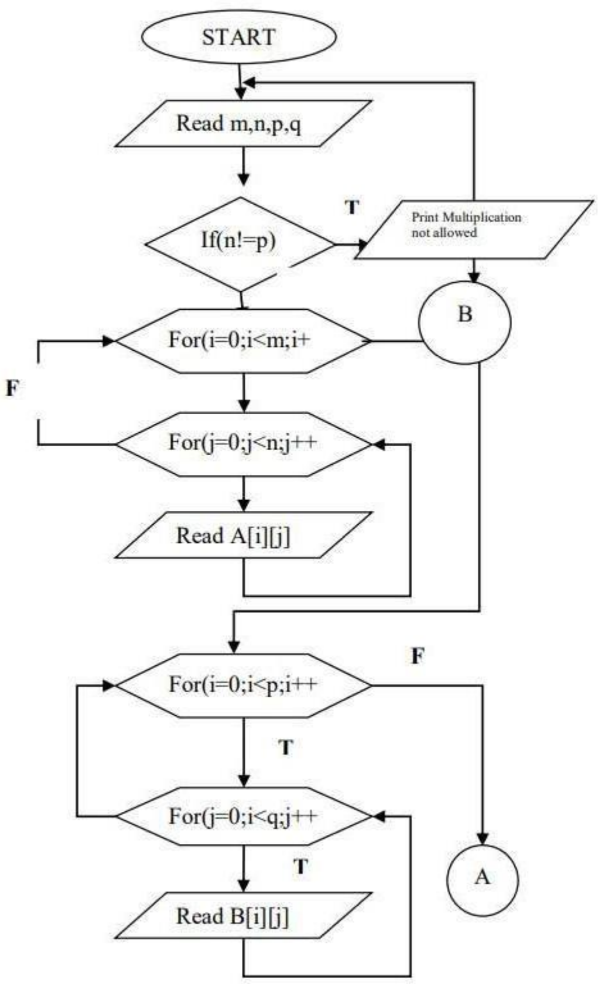
4	n=6
	Elements: 21 22 23 24 25 26
	Key= 11
		SEARCH UNSUCCESSFUL

Viva Questions:

Sl.No	Questions
1.	What are the two types of searches
2.	What is linear search
3.	What is binary search
4.	Syntax for while loop

6 Implement Matrix Multiplication and Validate the rules of Multiplication

Flowchart:



Algorithm:

```
Step 1: Start
Step 2: Read the order of two matrices A and B
Step 3: Check the number of rows and column of A and B matrices for compatibility
Step 4:  Read two matrices A and B
 Step 5: For i from 0 to m:
            For j from 0 to q:
                a.  Let c[i][j] = 0
                b.  For k from 1 to n:
                        Set c[i][j] ← c[i][j]+a[i][k]*b[k][j];
Step 6: Display the Matrix A and B. Resultant Matrix C
Step 7: Stop
```

Program:
```
#include<stdio.h>
void main()
{
        int i,j,k,a[10][10],b[10][10],c[10][10],m,n,p,q;

        printf("Enter the order of matrix A:\n");
        scanf("%d%d",&m,&n);
        printf("Enter the order of matrix B:\n");
        scanf("%d%d",&p,&q);
        if (n!=p)
        printf("Matrices are Incompatible\n");
        else
        {
                printf("Enter Matrix A\n");
                for (i=0;i<m;i++)
                        for (j=0;j<n;j++)
                            scanf("%d",&a[i][j]);
                printf("Enter Matrix B\n");
                for (i=0;i<p;i++)
                        for (j=0;j<q;j++)
                             scanf("%d",&b[i][j]);
                //matrix multiplication
                for (i=0;i<m;i++)
                {
                   for (j=0;j<q;j++)
                   {
                        c[i][j]=0;
                        for(k=0;k<n;k++)
                                c[i][j]=c[i][j]+a[i][k]*b[k][j];
                   }
                }
                printf("Matrix A:\n");
                for (i=0;i<m;i++)
                {
                for(j=0;j<n;j++)
                        printf("%d\t",a[i][j]);
                printf("\n");
                }
                printf("Matrix B:\n");
                for (i=0;i<p;i++)
                {
                for(j=0;j<q;j++)
                        printf("%d\t",b[i][j]);
                printf("\n");
                }
                printf("Matrix C:\n");
                for (i=0;i<m;i++)
                {
                for(j=0;j<q;j++)
                        printf("%d\t",c[i][j]);
                printf("\n");
                }
        }
}
```

Test Cases:

Sl. No.	Test Data				Expected Output		
	Order of Matrix A	Order of Matrix B	Enter Matrix A	Enter Matrix B			
1	2 2	2 2	1 2 3 4	1 2 3 4	Matrix A: 1 2 3 4	Matrix B: 1 2 3 4	Matrix C: 7 10 15 22
2	3 2	2 2	1 2 3 3 2 1	1 2 3 1	Matrix A: 1 2 3 3 2 1	Matrix B: 1 2 3 1	Matrix C: 7 4 12 9 5 5
3	2 4	1 2			Matrices are Incompatible		

VIVA Questions

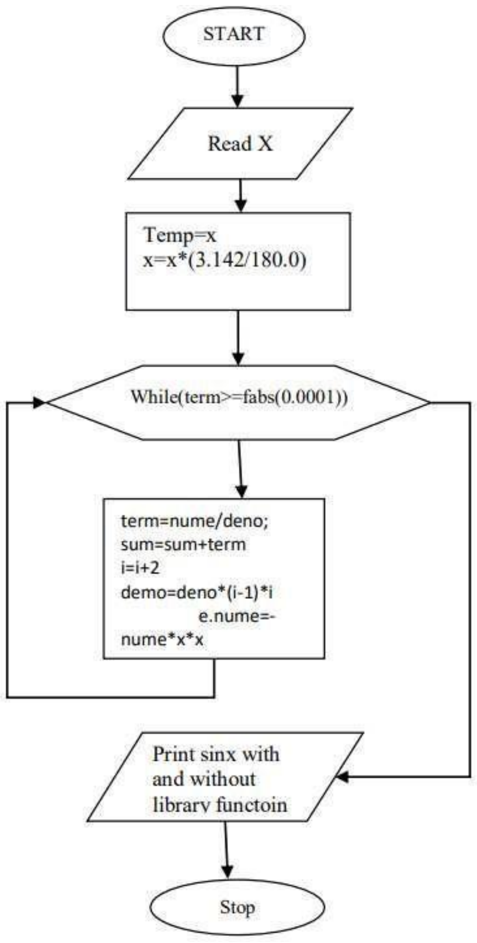
Sl. No	Questions
1.	What is matrix?
2.	What is one dimensional array?
3.	Write the Syntax for one dimensional array.
4.	What is two dimensional array?
5.	Write the Syntax for two dimensional array.

7 Develop a Program to compute Sin(x)/ Cos(x)using Taylor series approximation. Compare your result with the built- in Library function. Print both the results with appropriate messages.

Sin(x)= x- x^3/3! +x^5/5!+x^7/7!+……

Cos(x)=1- x^2/2! +x^4/4!+x^6/6!+……

Flowchart:



Algorithm:

```
Step 1: Start
Step 2: Read degree
Step 3: Convert degree into radians- x=degree*(3.142/180);
Step 4: Assign sum=0,deno=x,nume=1,i=1;
Step 5:do - while term >.0001 repeat
        a.term=nume/deno;
        b.sum=sum+term
        c.i=i+2
        d.deno=deno*(i-1)*i
        e.nume=-nume*x*x
Step 6: Display Sin(x) and Sum using Taylor Series and Built in library function
Step 7: Stop
```

Program:

```
#include<stdio.h>
#include<math.h>
void main()
{
        float x,sum,nume,deno,term;
        int degree, i;
        printf("Enter degree:\n");
        scanf("%d",&degree);
        x=degree*(3.142/180);
        sum=0;
        nume=x;
        deno=1;
        i=1;
        do
        {
                term=nume/deno;
                sum=sum+term;
                i=i+2;
                deno=deno*(i-1)*i;
                nume=-nume*x*x;
        }while(fabs(term)>0.00001);

        printf("Sin %d using Taylor's Series:%f\n",degree,sum);
        printf("Sin %d using built-in Library function:%f\n",degree,sin(x));
}
```

Test Cases:

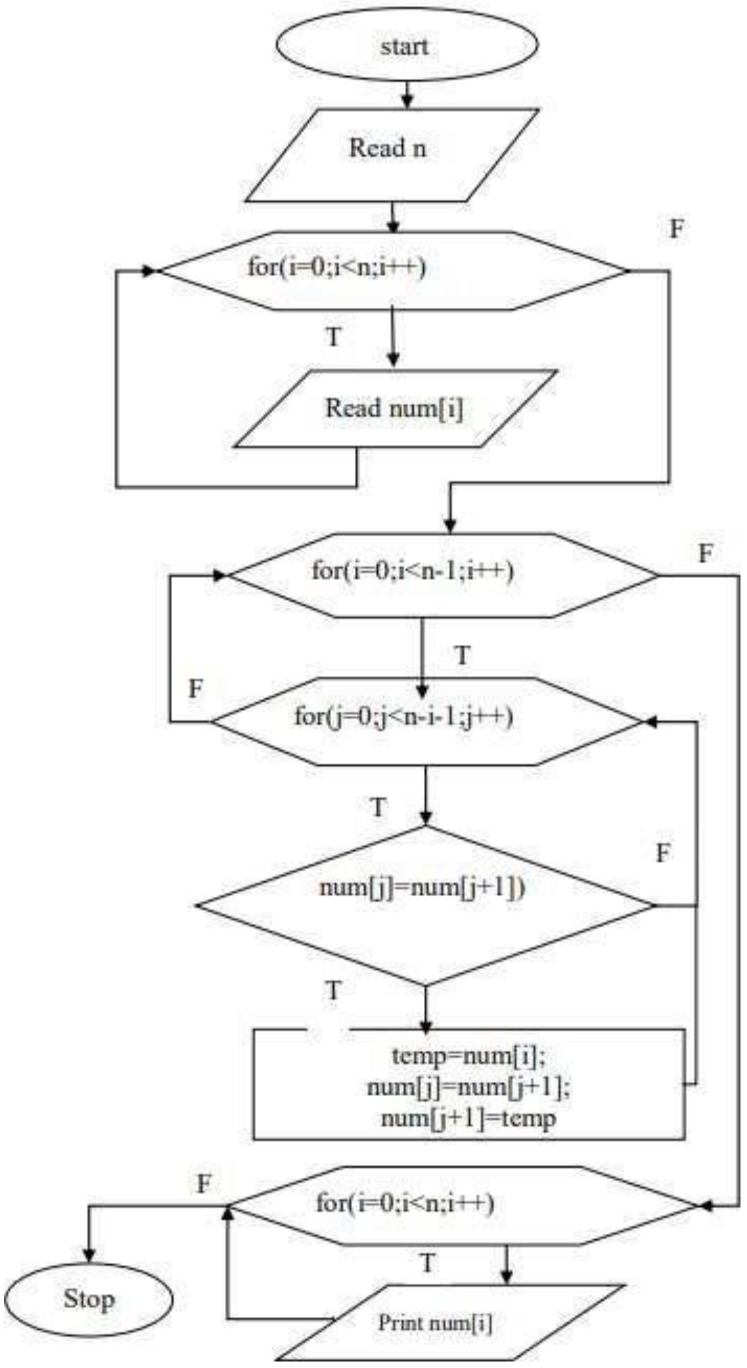
Sl no.	Test Data	Expected Output
1	90	Sin 90 using Taylor's Series:1.000000 Sin 90 using built-in Library function:1.000000
2	45	Sin 45 using Taylor's Series:0.707179 Sin 45 using built-in Library function:0.707179

Viva Questions

Sl. No	Questions
1.	What is Taylor series?
2	What is the built- in Library function
3	What is power functions
4	What is user defined functions?

8 Sort the given set of *N* numbers using Bubble Sort.

Flowchart:



Algorithm:

```
1.  Start
2.  Read n.
3.  Read the elements in an array using for loop.
4.  for i=0 to i<n-1
5.  for j=0 to  j<n-1
    if (a[j]>a[j+1])
    swap a[j],a[j+1]
6.  Display the sorted array
7.  End
```

Program:

```
#include<stdio.h>
void main()
{
        int n,i,j,a[10],t;
        printf("Enter n\n");
        scanf("%d",&n);
        printf("Enter elements\n");
        for (i=0;i<n;i++)
          {
            scanf("%d",&a[i]);
          }
        for  (i=0;i<n-1;i++)
        {
            for  (j=0;j<n-1;j++)
             {
                if (a[j]>a[j+1])
                {
                        t=a[j];
                        a[j]=a[j+1];
                        a[j+1]=t;
                }
             }
        }
        printf("Sorted Array\n");
        for (i=0;i<n;i++)
        {
              printf("%d\n",a[i]);
        }
}
```

Test Cases:

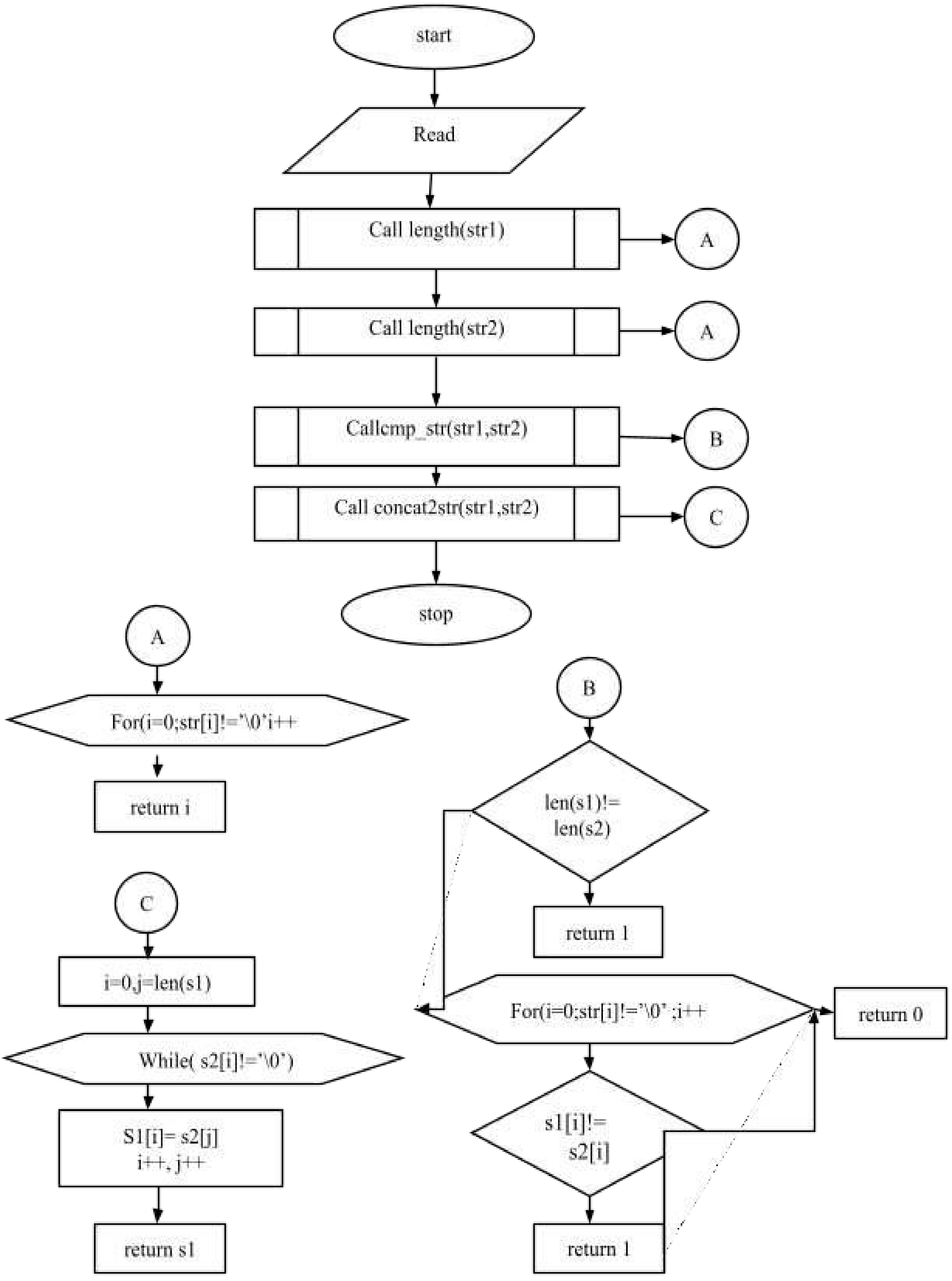
Sl.no.	Test Data		Expected Output
	N	Elements	
1	4	5 3 2 1	1 2 3 5
2	5	5 6 2 3 1	1 2 3 5 6

Viva Questions:

Sl.No	Questions
1	What is bubble sort algorithm?
2	For loop Syntax
3	If loop syntax
4.	What is an array?
5.	How to initialize and declare an array?

9 Write functions to implement string operations such as compare, concatenate, string length. Convince the parameter passing techniques.

Flowchart:



Algorithm:

```
Step 1: Start
Step 2: Read two strings S1 and S2
Step 3: Compute len_str(s1)
Step 4: Display the length of string s1
Step 5: Compute len_str(s2)
Step 6: Display the length of string s2
Step 7: Compare two strings using comp_str(s1,s2)
        a. If the function returns 0 Display the strings are equal
        b. If the function returns 1 Display the strings are not equal
Step 8: Concatenate two strings using concat_str(s1,s2)
Step 9:Stop
```

Algorithm for String length:

```
Step 1: Initialize len=0
Step 2: for (i = 0; s[i] != '\0'; i++)
        Increment len
Step 3: Return len to the main program
```

Algorithm for String Comparision:

```
Step 1: Compute the length of string s1 and s2 using the function len_str(s1) and len_str(s2)
Step 2: if lengths are not equal
        Return 1
Step 3: for (i = 0; i< len1; i++)
        if (s1[i] !=s2[i])
         return 1
 Step 4: return 0
```

Algorithm for String Concatenation:

```
Step 1: Initialize i=0 and j=length of string1 using len_str(s1)
Step 2: while(s2[i]!='\0')
          Assign s1[j]=s2[i];
          Increment j and i
Step 3: Assign s1[j]=null
Step 4: Display Concatenated String S1
Step 5: Return to main Program
```

Program:

```
#include <stdio.h>

int len_str(char s[50]);
int comp_str(char s1[50], char s2[50]) ;
void concat_str(char s1[50],char s2[50]);
void  main()
{
        char s1[50],s2[50],len,c;
        printf("enter the 1st string\n");
        scanf("%s",s1);
        printf("enter the 2nd string\n");
        scanf("%s",s2);
        len=len_str(s1);
        printf("length of the string1 = %d\n",len);
```

```
            len=len_str(s2);
            printf("length of the string2 = %d\n",len);
            c=comp_str(s1,s2);
            if(c==0)
            {
                   printf("strings are equal\n");
            }
            else
            {
                  printf("strings are not equal\n");
            }
            concat_str(s1,s2);
}

int len_str(char s[50])
 {
            int len = 0, i;
            for (i = 0; s[i] != '\0'; i++)
           {
                  len++;
           }
            return  len;
  }

int comp_str(char s1[50], char s2[50])
{
            int len1, len2, i;
            len1 = len_str(s1);
            len2 = len_str(s2);
            if (len1 != len2)
            return 1;
            for (i = 0; i< len1; i++)
            {
                  if (s1[i] !=s2[i])
                  return 1;
            }
            return 0;
 }
void  concat_str(char  s1[50],char  s2[50])
{
            int i=0,j;
            j=len_str(s1);
            while(s2[i]!='\0')
          {
                   s1[j]=s2[i];
                   j++;
                   i++;
       }
       s1[j]='\0';
       printf("Concatenation of string1 and string2  %s \n",s1);
}
```

Test Cases:

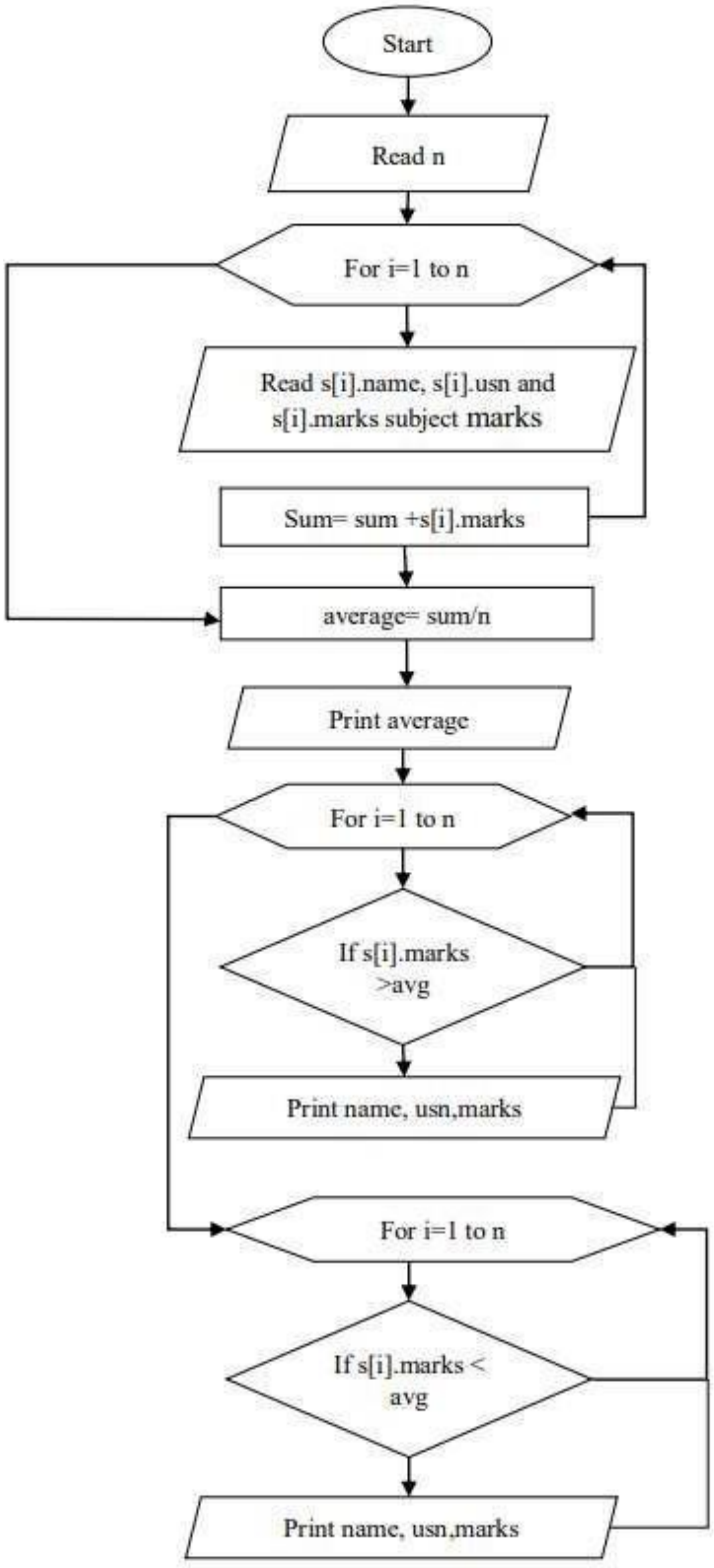
Sl No.	Test Data		Expected Output
	String1	**String2**	
1	Hello	World	Length of the string1=5 Length of the string2=5 Strings are not equal Concatenation of string1 and string2 HelloWorld
2	Computer	Science	Length of the string1=8 Length of the string2=7 Strings are not equal Concatenation of string1 and string2 Computer Science
3	Computer	Computer	Length of the string1=8 Length of the string2=8 Strings are not equal Concatenation of string1 and string2 Computer

Viva Questions

SL.NO	Questions
1.	What is the difference between string and character?
2.	What is string?
3.	What is the format specifier for the string?
4.	What is string copy function?
5.	Name the header file for string?

10 Implement structures to read, write and compute average marks and students scoring above and below the average marks for a class of N students.

Flowchart:



Algorithm:
Step 1: Start
Step 2: Read the number of students
Step 3: Read all the members of the structure for n students
Step 4: Calculate the average marks of all the students
Step 5: Display all the members of the structure
Step 6: Display all the details of student whose average is greater than or equal to 35
Step 7: Display all the details of student whose average is less than 35
Step 8: Stop

Program:
```
include <stdio.h>
void main()
{
        struct stud
        {
            char name[25];
            char usn[25];
            int marks;
        };
        struct stud s[25];
        int n,i,sum;
        float average;
        sum=0;
        printf("\nenter number of students :");
        scanf("%d",&n);
        printf("enter the name usn and marks of %d students\n",n);
        for(i=0;i<n;i++)
        {
            scanf("%s%s%d",s[i].name,s[i].usn,&s[i].marks);
            sum=sum+s[i].marks;
        }
        average=sum/(float)n;
        printf("average marks=%f",average);
        printf(" \nDetails of students scoring above average marks\n");
        for(i=0;i<n;i++)
        {
             if (s[i].marks >= average)
                  printf("name=%s \t usn=%s \t  marks= %d\n",s[i].name,s[i].usn,s[i].marks);
        }
        printf(" \nDetails of students scoring below average marks\n");
        for(i=0;i<n;i++)
        {
            if (s[i].marks < average)
                  printf("name=%s \t usn=%s \t  marks= %d\n",s[i].name,s[i].usn,s[i].marks);
        }
}
```

Test Cases:

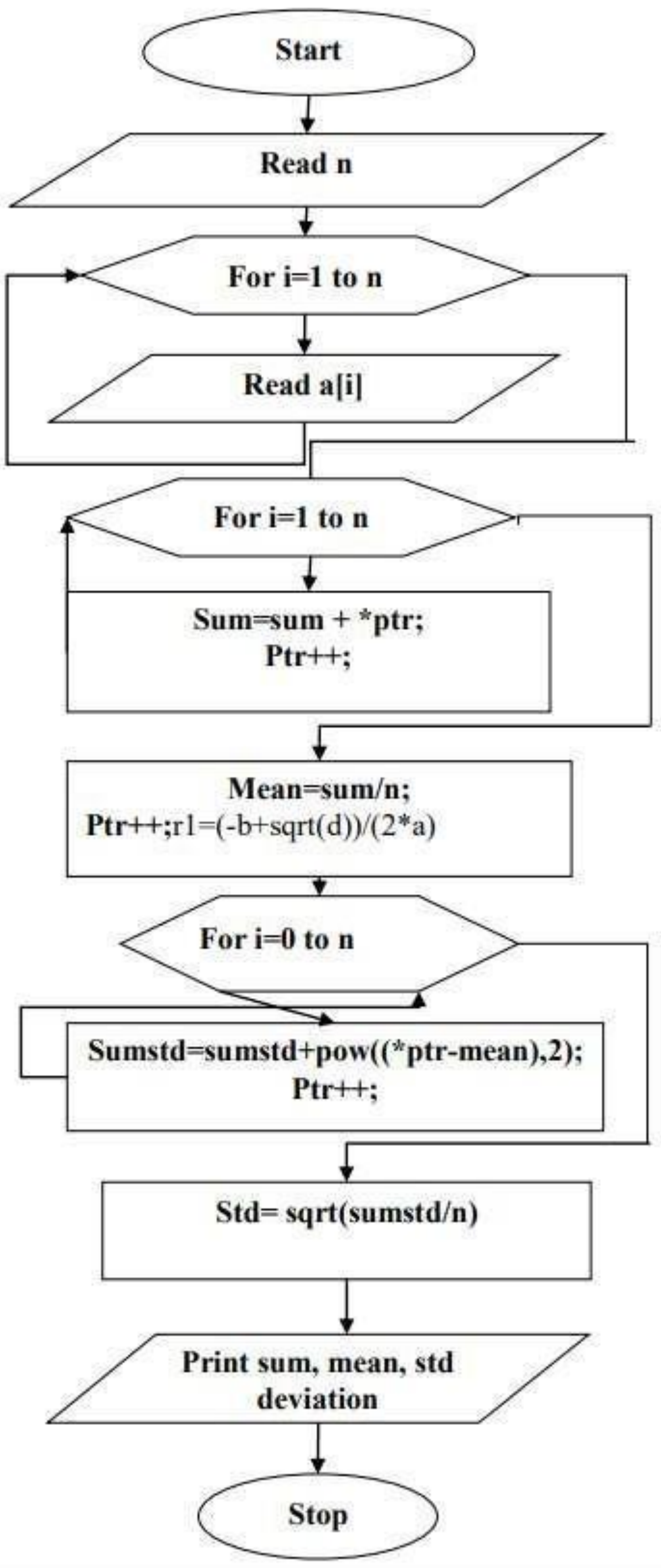
Sl no.	Test Data				Expected Output
	No. of students	Information of students			
		Roll no.	Name	Marks	
1	2	12	John	98	Average=97 Student above average 13 Edwin 99 Students below average 12 John 98
		13	Edwin	99	

Viva questions

SL NO	**Questions**
1	What is structure?
2	Explain the syntax of structure?
3	Distinguish between array and structure?
4	Syntax for accessing the member of structure?
5	Explain strcmp?

11 Develop a program using pointers to compute the sum, mean and standard deviation of all elements stored in an array of n real numbers.

<u>Flowchart:</u>



Algorithm:

```
Step 1: Start
Step 2: Read the number of elements
Step 3: Read the elements using for loop
Step 4: Initialize ptr=a
Step 5: Using for loop calculate sum
        sum=sum+*ptr
Step 6: Calculate Mean
        Mean=sum/n
Step 7: Initialize ptr=a
        Using for loop calculate sumstd
        sumstd=sumstd+pow((*ptr-mean),2)
Step 8: Calculate Standard Deviation
        std=sqrt(sumstd/n)
Step 9: Display Sum, Mean, Standard Deviation
Step 10: Stop
```

Program:

```
#include<stdio.h>
#include<math.h>
void main()
{
        float a[10],*ptr,mean,std,sum=0,sumstd=0;
        int n,i;
        printf("Enter the no of elements\n");
        scanf("%d",&n);
        printf("Enter the array elements\n");
        for(i=0;i<n;i++)
        {
                scanf("%f",&a[i]);
        }
        ptr=a;
        for(i=0;i<n;i++)
        {
                sum=sum+*ptr;
                ptr++;
        }
        mean=sum/n;
        ptr=a;
        for(i=0;i<n;i++)
        {
                sumstd=sumstd+pow((*ptr-mean),2);
                ptr++;
        }
        std=sqrt(sumstd/n);
        printf("sum=%f\n",sum);
        printf("Mean=%f\n",mean);
        printf("standard deviation=%f\n",std);
}
```

Test Cases:

Sl no.	Test Data		Expected Output		
	No. of Elements	Array Elements	Sum	Mean	Standard Deviation
1	5	1 2 3 4 5	15.000000	3.000000	1.414214
2	4	2 3 4 5	14.000000	3.500000	1.118034

Viva Questions:

SL NO	**Questions**
1	Define pointer?
2	What is formula for sum?
3	What is formula for standard deviation?
4	What is formula for mean?
5	How to declare and initialize pointer variables?

12 Write a C program to copy a text file to another, read both input file name and target file name

Algorithm

1) Start
2) Initialize the necessary variables
3) Create file pointers to respective files
4) Get the file name for reading the content
5) Open the file with read mode , fptr1
6) Get the filename for writing the content
7) Open the filename with write mode, fptr2
8) Read the content of fptr1 and write into fptr2 using fgetc and putfc function
9) Stop

Program

```
#include<stdio.h>
#include<stdlib.h>// For exit()
int main()
{
  FILE *fptr1,*fptr2;
  char filename[100], c;
  printf("Enter the filename to open for reading");
  scanf("%s",filename);
  // Open one file for reading
  fptr1 =fopen(filename,"r");
  if(fptr1 == NULL)
  {
    printf("Cannot open file %s ", filename);
    exit(0);
  }
  printf("Enter the filename to open for writing ");
  scanf("%s", filename);
  // Open another file for writing
  fptr2 =fopen(filename,"w");
  if(fptr2 == NULL)
  {
    printf("Cannot open file %s ", filename);
    exit(0);
  }
  // Read contents from file
  c =fgetc(fptr1);
  while(c != EOF)
 {
    fputc(c, fptr2);
    c =fgetc(fptr1);
  }
  printf("Contents copied to %s", filename);
  fclose(fptr1);
  fclose(fptr2);
  return0;
}
```

Output

When the above program is executed, it produces the following result −

```
Enter the filename to open for reading
file3.txt
Enter the filename to open for writing
file1.txt
Contents copied to file1.txt
```