Principles Of Programming Language

**Laboratory file**

**Submitted By:** SAIRANJAN SUBUDHI

SAP ID: 500101861

BATCH: 24

B.Tech(Hons)Computer Science & Engineering

**Submitted To:**

Kalpana Rangra

Experiment No. 1

**Objective:** To familiarize Linux Commands and demonstrate a clear understanding of the C-programming environment.

**List of Lab Activities:**

1. Study of Linux Commands

a. Working with Directories: mkdir, rmdir, dir, pwd, cd, ls

b. Handling Files: vi, gedit, more, cp, mv, rm

2. Familiarization of the following

a. Structure of C program [Example: C program to print “My name is ...”]

b. Execution Environment & Stages of Compilation. Use Linux vi editor on the terminal window.

**“Mkdir”:** It is a command used in Linux to create new directories.

“**Rmdir**”: It is a command used to remove any empty directiories but if a file is present in it, first you have to remove the file then you can remove the directory.

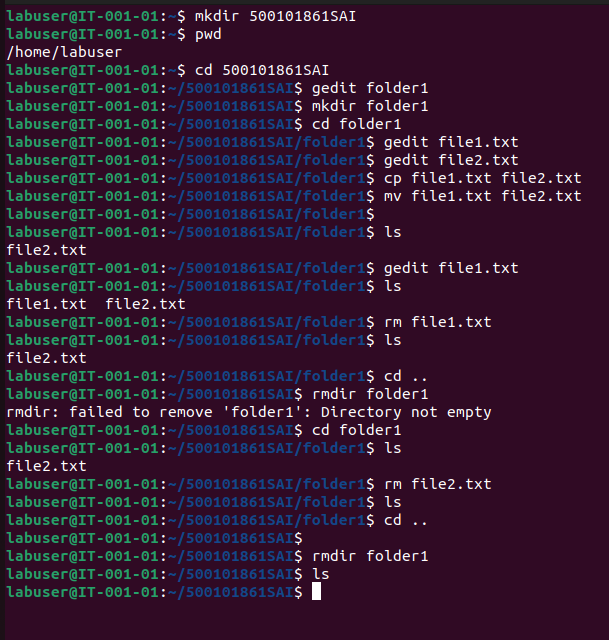
**“ls”:** It lists the content of a directory or file information.

**“cd”:** It changes the current directory to the previous directory.

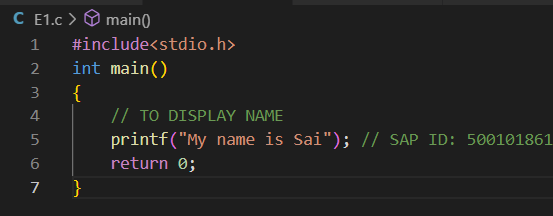
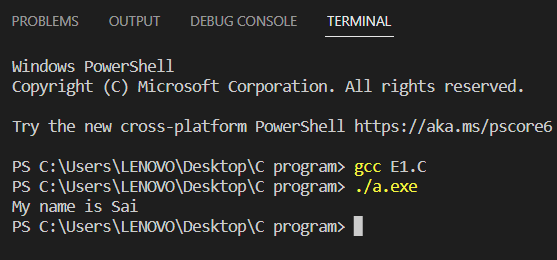
**“pwd”:** It commands to print the absolute path to current working directory.

**“rm”:** It is used to remove files, directories etc. from the file system.

**“cp”:** It copies file from one directory to the designated directory.



**WRITE A CODE TO PRINT YOUR NAME**

**EXPERIMENT-2**: Basics of Problem Solving and Program Control Flows

**Objective:** To demonstrate use of algorithm and flowchart to design solutions for problems with appropriate selection of program control flow.

**List of Lab Activities:** Identify the suitable program control flow to solve the given problem. Write algorithm, draw flow chart, prepare test cases and test the design for completeness.

1. Given 2 numbers. Calculate sum, difference, multiplication and division.

2. Find if the given number is even or not.

3. Find the biggest of three numbers.

4. Multiply two numbers without using arithmetic multiplication operator (\*).

1. **Given 2 numbers. Calculate sum, difference, multiplication and division.**

**ALGORITHM:**

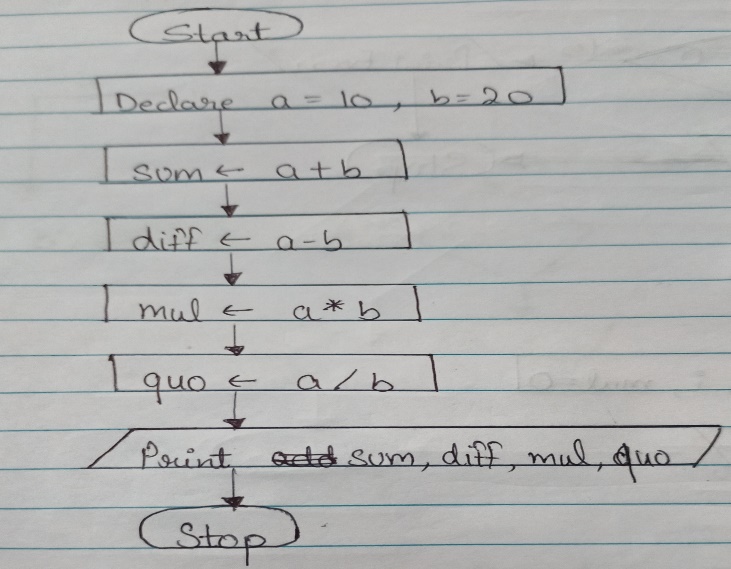
STEP-1: Input two numbers a and b.

STEP-2: sum=a+b, diff=a-b, mul=a\*b, quo= a/b.

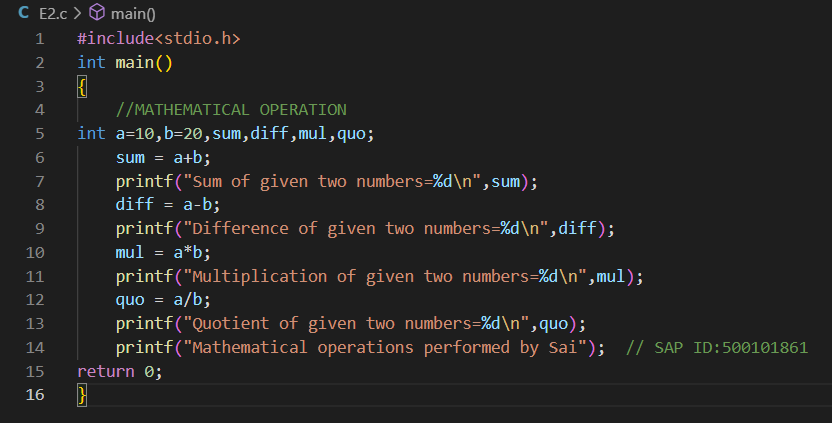
STEP-3: Print sum, diff, mul, quo .

STEP-4: STOP TEST CASES (a=10&b=20).

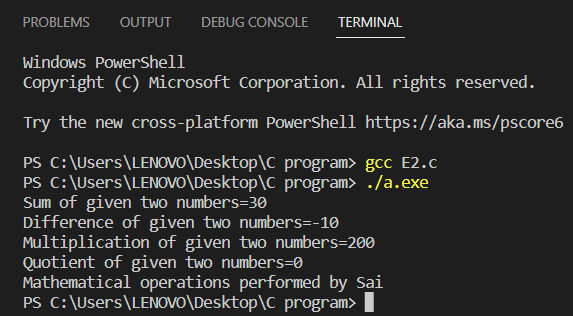
**FLOWCHART:**



**CODE SNIPPET:**



**OUTPUT SNIPPET:**



1. **Find if the given number is even or not**

**ALGORITHM:**

STEP1: Start

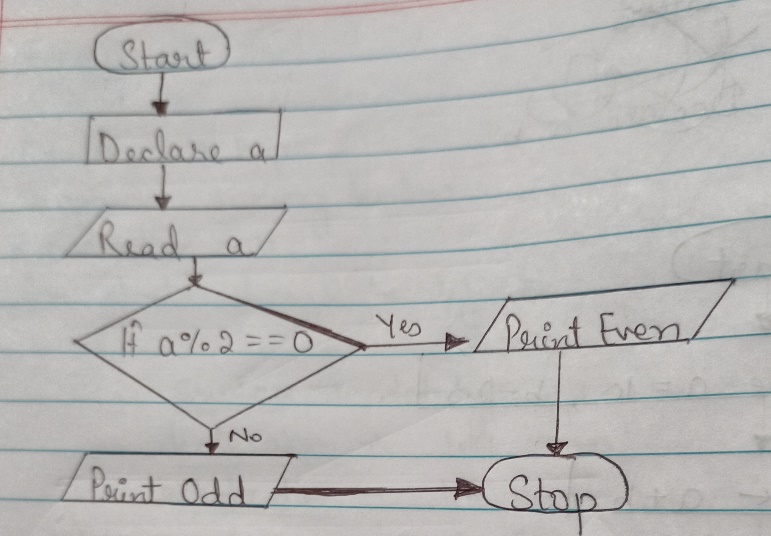
STEP2: Input the number say;a

STEP3: Divide the number with 2

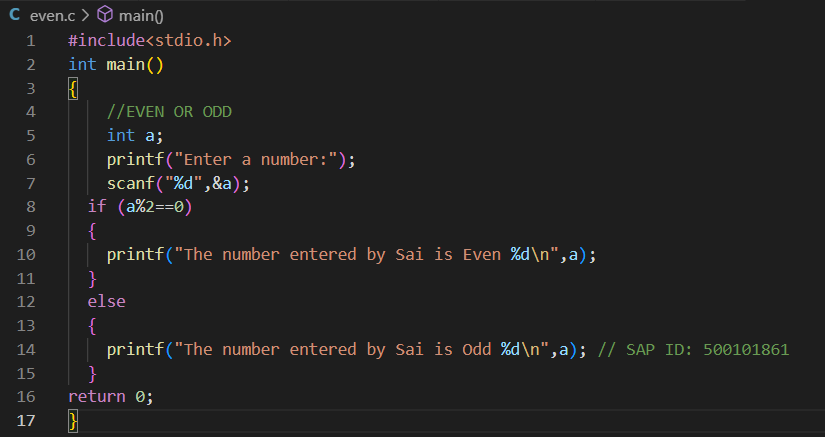
STEP4: If the remainder is 0 then the number is even.if the remainder is not zero the number is odd.

STEP5: Stop TEST CASE: IF a=20 divide by 2, remainder is 0 ,therefore a is even. If a=25 divide by 2 remainder is 1. Therefore a is odd.

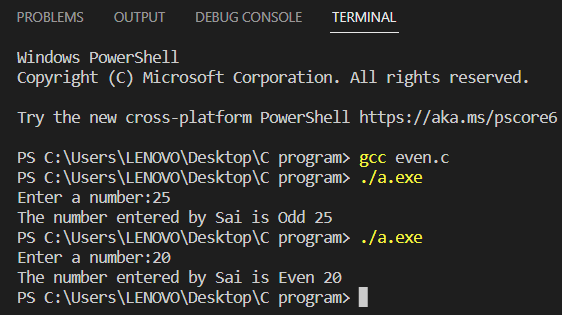
**FLOWCHART:**



**CODE SNIPPET:**



**OUTPUT SNIPPET:**



1. **Find the biggest of three numbers**.

**ALGORITHM:**

Step1: Input the values of the three numbers saying a,b,c.

Step2: If a>b.

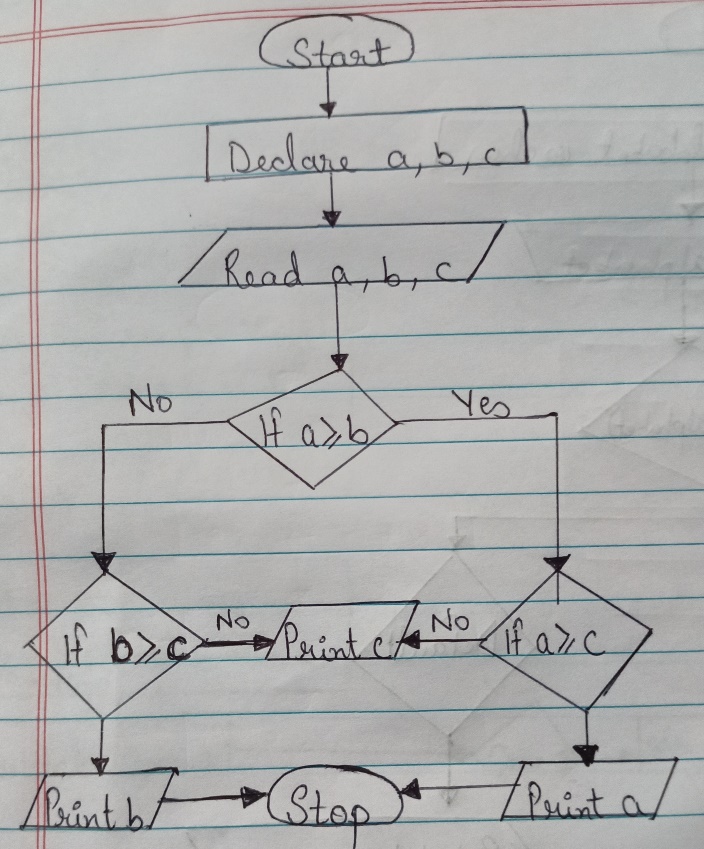
Step3: If yes then check if a>c.

Step4: If yes then a is the largest else c is the largest.

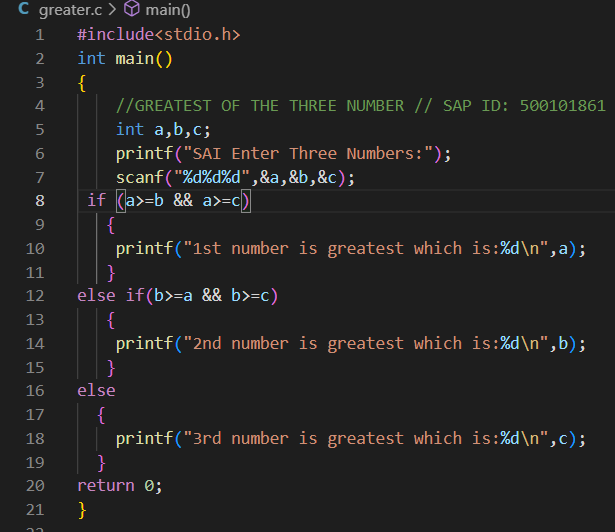
Step5: If a is not greater than b then check b>c or not. Step6: If yes then b is the largest.

Step7: Else c is the largest.

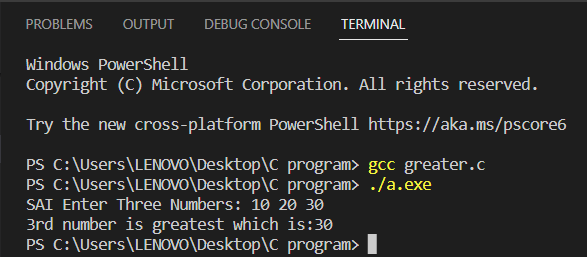
**FLOWCHART:**



**CODE SNIPPET:**



**OUTPUT SNIPPET:**



1. **Multiply two numbers without using arithmetic multiplication operator (\*).**

**ALGORITHM:**

Step1: Start

Step2: Take input from user in a,b and also declare mul and I as int data type.

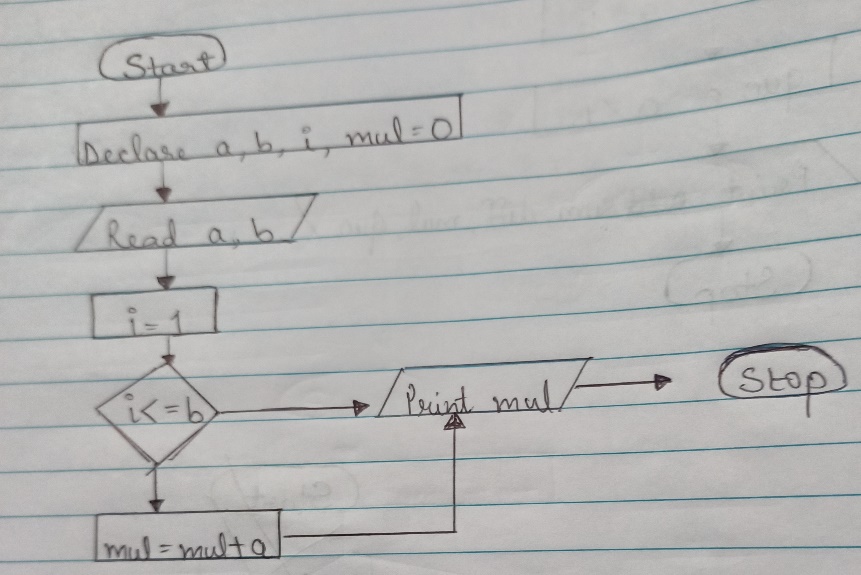
Step3: Take i=1

Step4: If i<=b

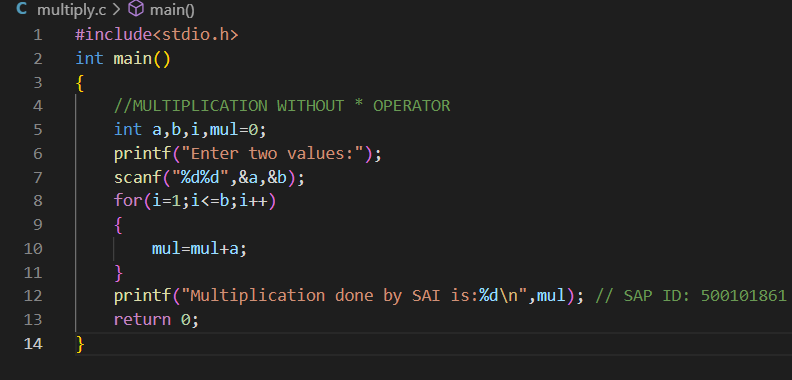
Step5: If the condition is true, then apply mul=mul+a and then print the value of mul else directly print the value of mul.

Step6: Stop

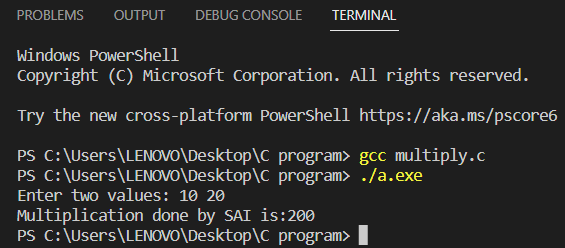
**FLOWCHART:**



**CODE SNIPPET:**



**OUTPUT SNIPPET:**



**Experiment no. 3:** Programming Sequential Logic

**Objective:** To code sequential logic in C language

**List of Lab Activities:**

Write algorithm and C program, compile, execute and test the code with Linux C compiler with suitable test cases.

1. Obtain the required inputs and compute the areas of the following shapes: (i) Parallelogram (with base and height), (ii) Trapezoid (with height, long base, short base), (iii) Rhombus (with height and side), (iv) Sphere(with radius), (v) Ellipse (with major and minor radius).

2. Given two numbers. Demonstrate the swapping of the values (i) using a third variable (ii) without using a third variable.

3. Convert temperature from Celsius to Fahrenheit and Kelvin.

4. Print the given days in years-month-days format. E.g. 396 days = 1 year, 1 month, 1 day

1. **Obtain the required inputs and compute the areas of the following shapes: (i) Parallelogram (with base and height), (ii) Trapezoid (with height, long base, short base), (iii) Rhombus (with height and side), (iv) Sphere(with radius), (v) Ellipse (with major and minor radius)**

**ALGORITHM:**

1. START
2. Declare shape,pbase,pheight,tlb,tsb,theight, rside,rheight,area\_p,area\_t,area\_r as int data type.
3. Declare sradius,maj\_radius,min\_radius,area\_s, area\_r as float data type.
4. Use switch statement.
5. Case1, we find out area of parallelogram.

Case2, we find out area of trapezoid.

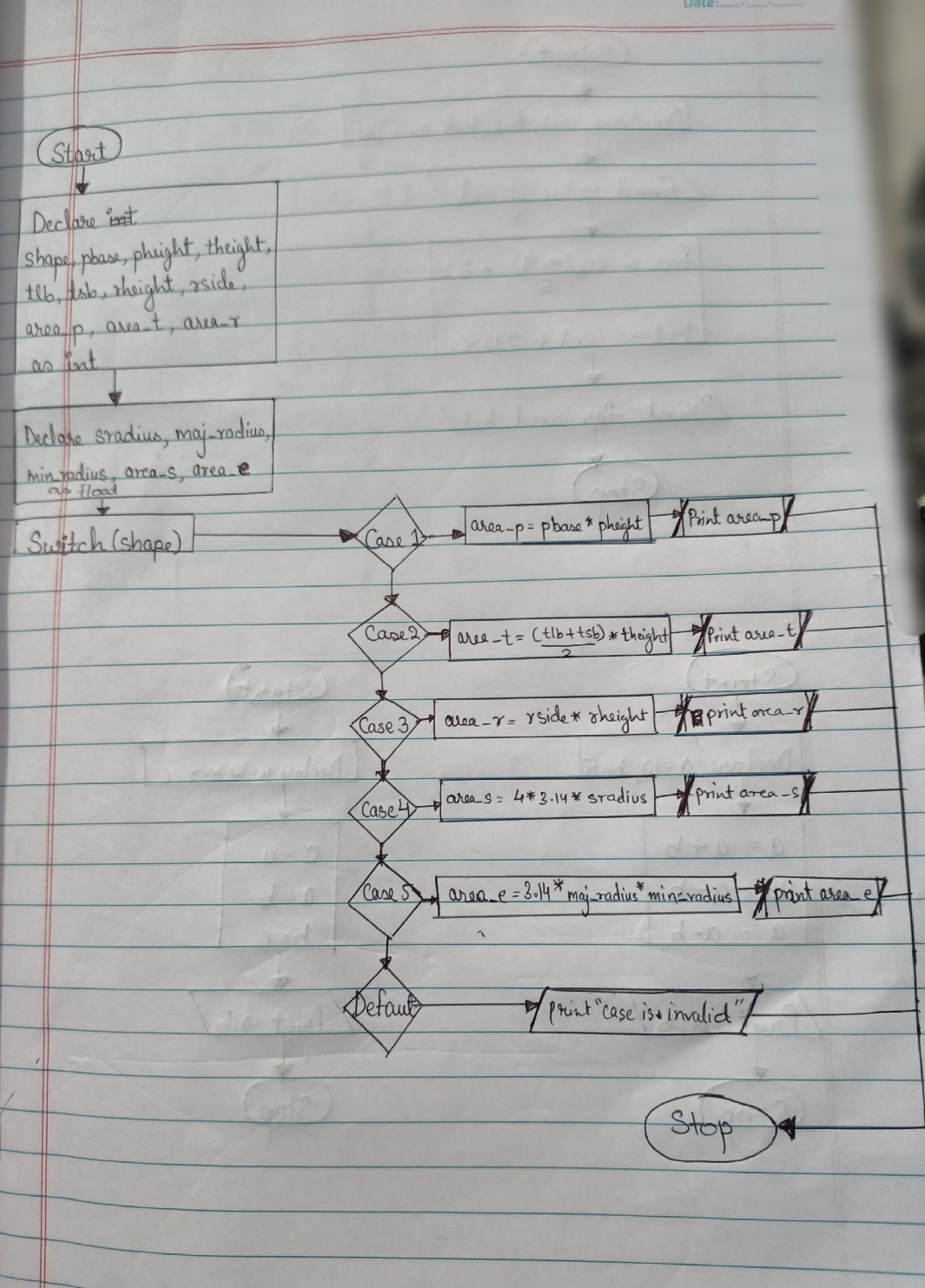
Case3, we find out area of rhombus.

Case4, we find out area of sphere.

Case5, we find out area of ellipse.

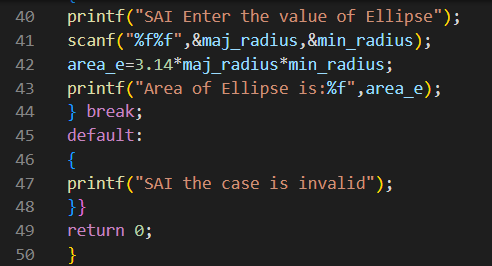
1. It will check which is the case or else it will print default case which would print that case is invalid.
2. Stop

**FLOWCHART:**

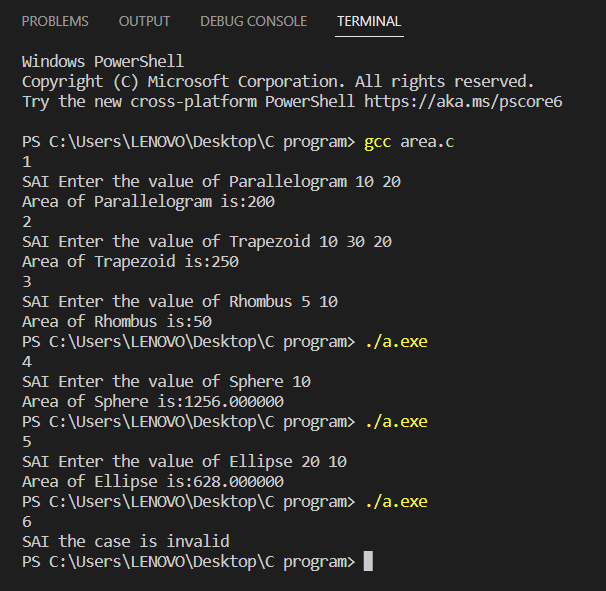


**CODE SNIPPET:**





**OUTPUT SNIPPET:**

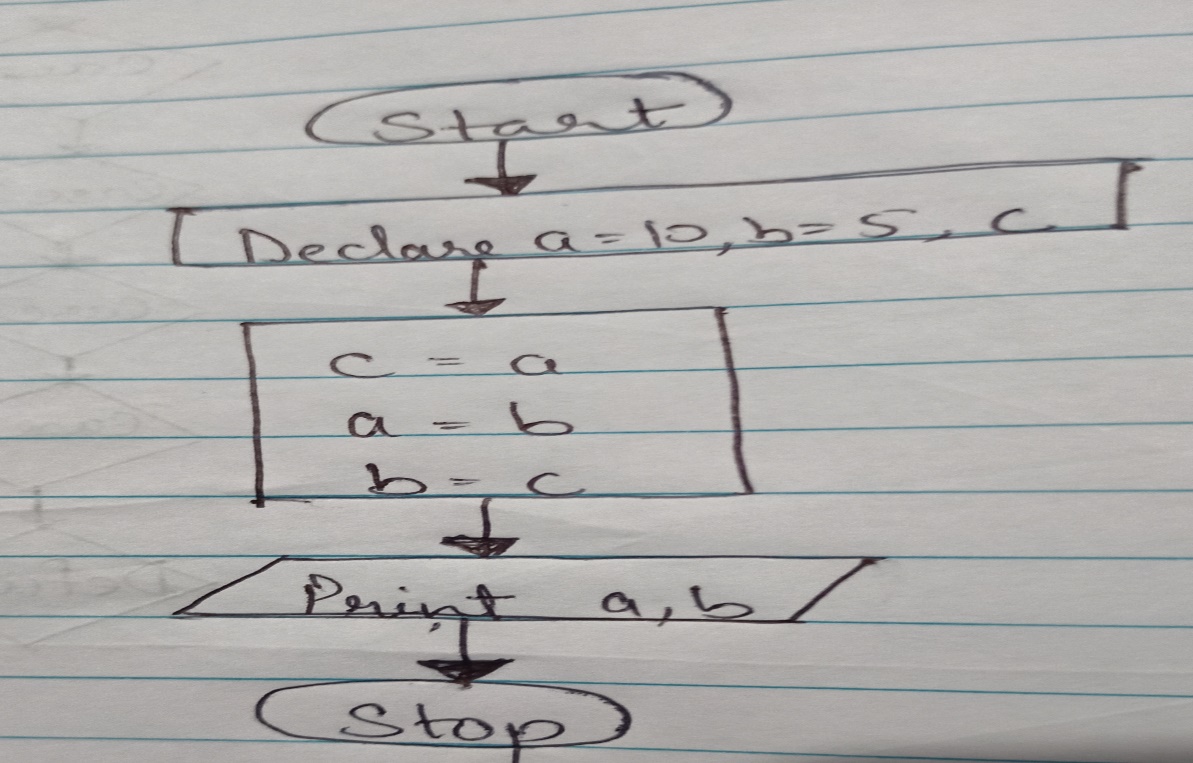


1. **Given two numbers. Demonstrate the swapping of the values (i) using a third variable (ii) without using a third variable.**

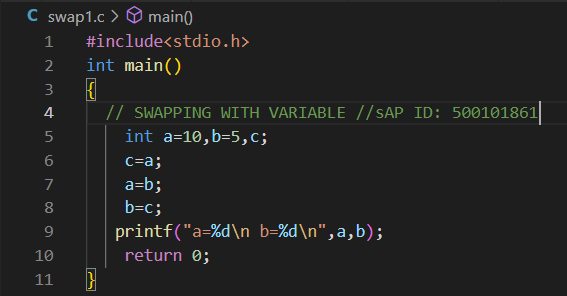
**ALGORITHM:**

Using a third variable  
STEP1: Start  
STEP2:Read a,b  
STEP3: c=a  
STEP4: a= b  
STEP5: b=c  
STEP6: Print a,b  
STEP7: Stop TEST CASE:  
Input values  
a=10 b=5  
Output values  
a=5 b=10

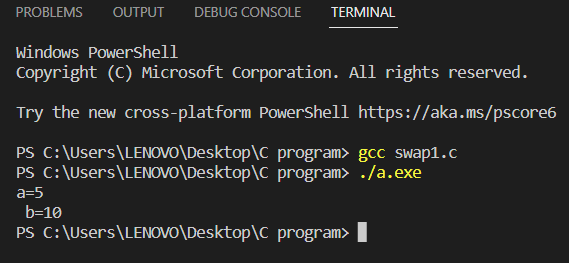
**FLOWCHART:**



**CODE SNIPPET:**



**OUTPUT SNIPPET:**

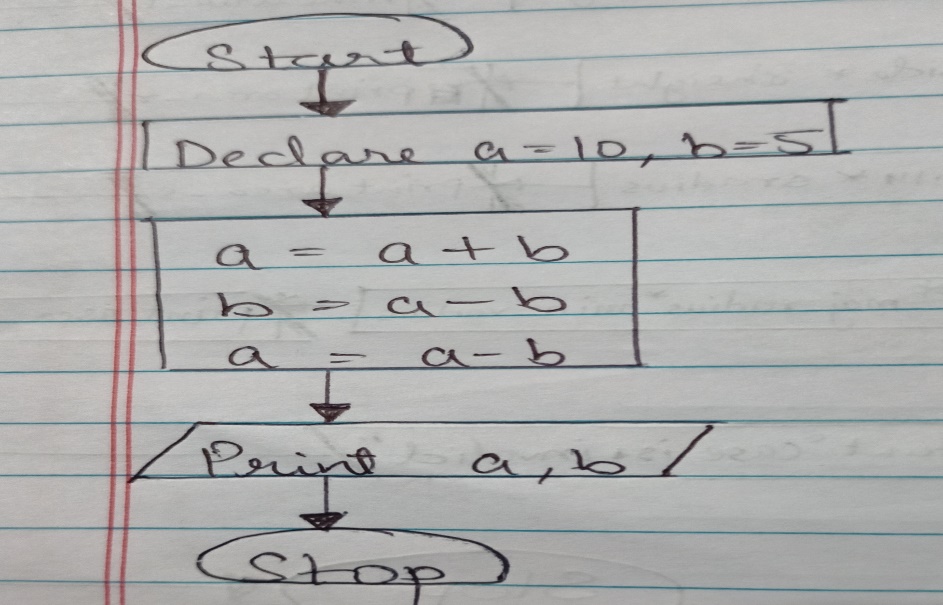


**(ii)**

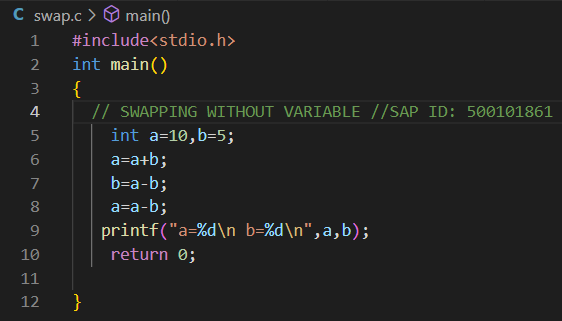
**ALGORITHM:**

Without using a third variable  
 STEP1: Start  
 STEP2: Read a,b  
 STEP3: a=a+ b  
 STEP4: b=a- b  
 STEP5: a=a-b  
 STEP6:Print a,b  
 STEP7: Stop

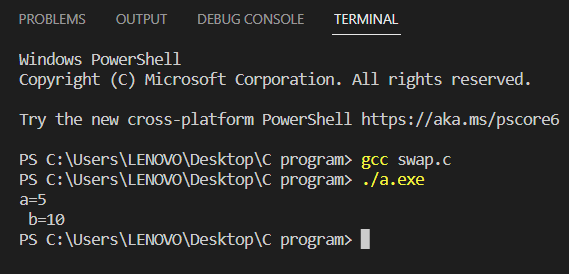
**FLOWCHART:**



**CODE SNIPPET :**



**OUTPUT SNIPPET:**



1. **Convert temperature from Celsius to Fahrenheit and Kelvin.**

**ALGORITHM:**

## Step1: Start

## Step2: Take a value x in Celsius

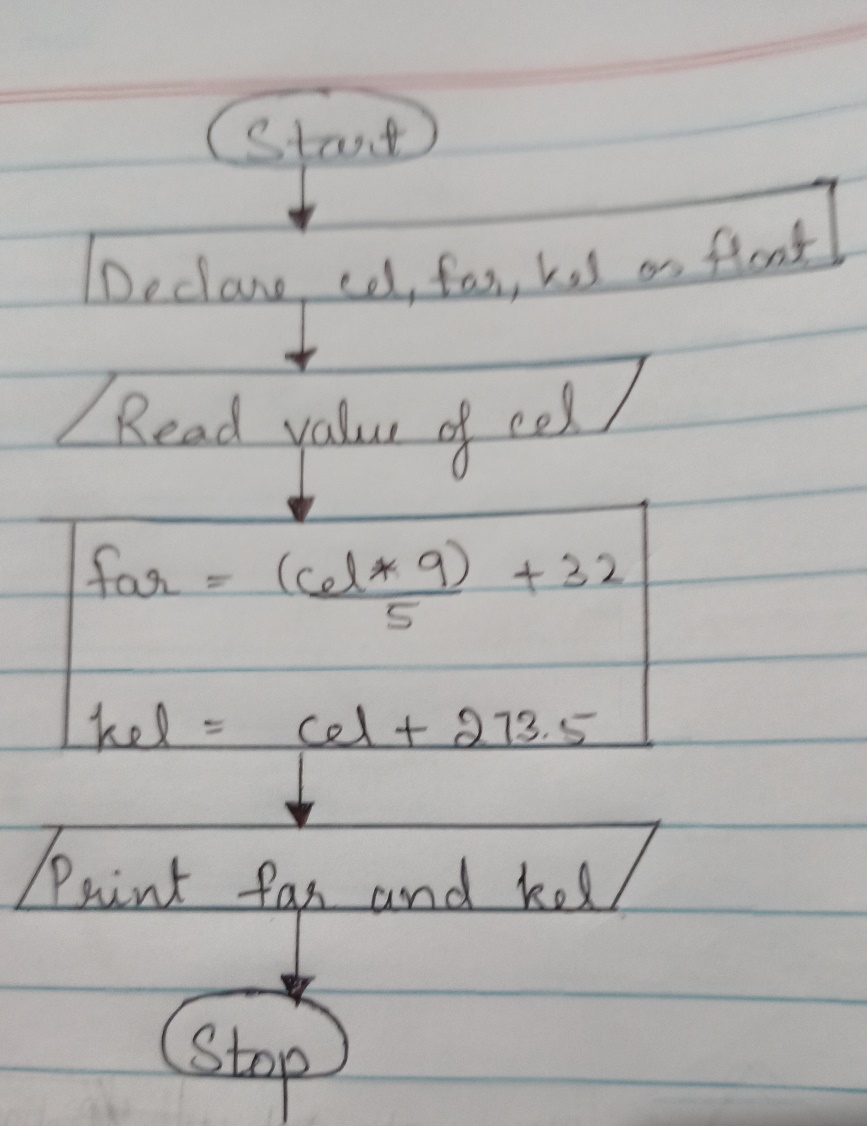
## Step3: Now, (i) (x \* (9/5)) + 32 = F (ii) x + 273 = K

## Step4: Display value of F and K

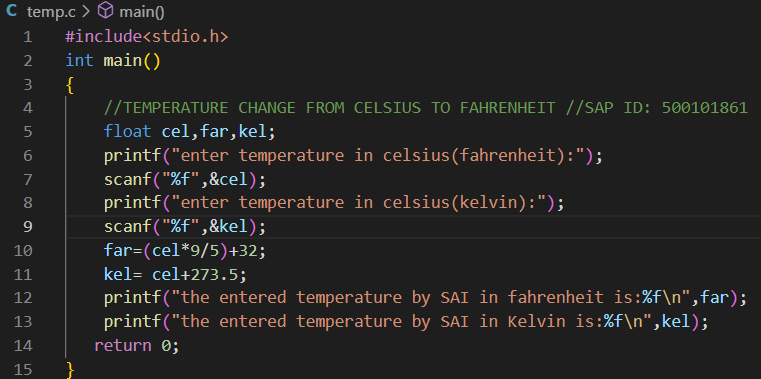
## Step5: Stop

## 

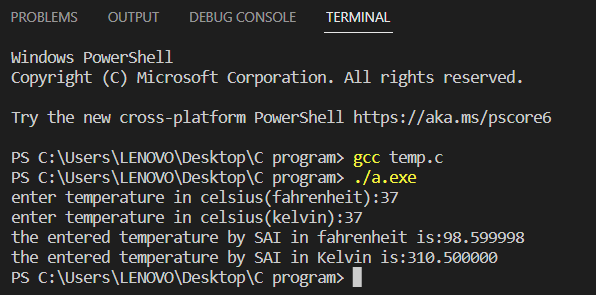
**FLOWCHART:**



**CODE SNIPPET:**



**OUTPUT SNIPPET:**

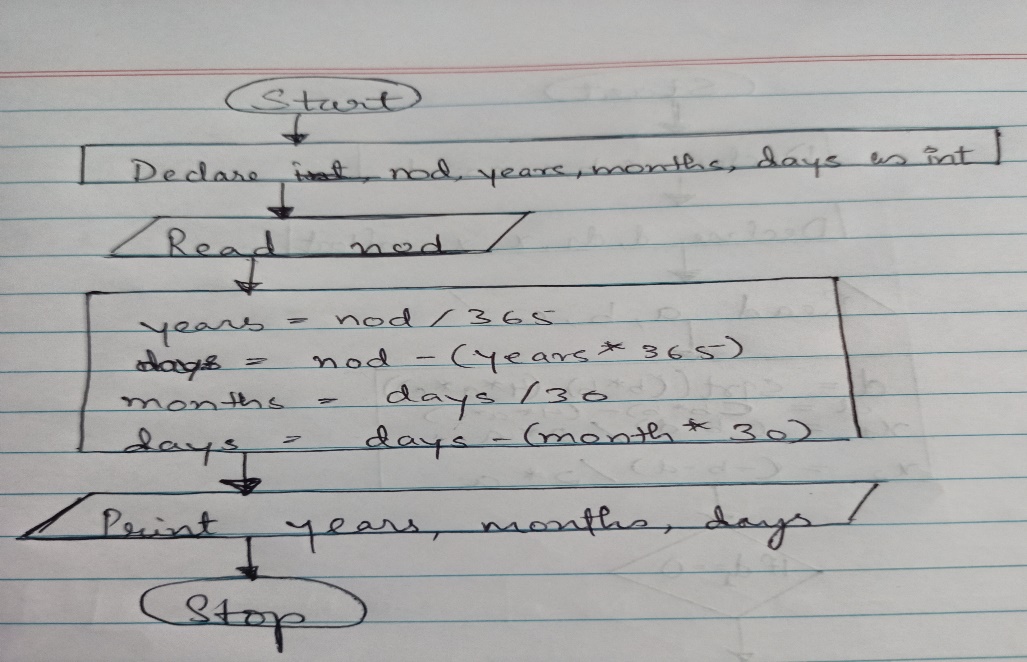


**4. Print the given days in years-month-days format. E.g. 396 days = 1 year, 1 month, 1 day**

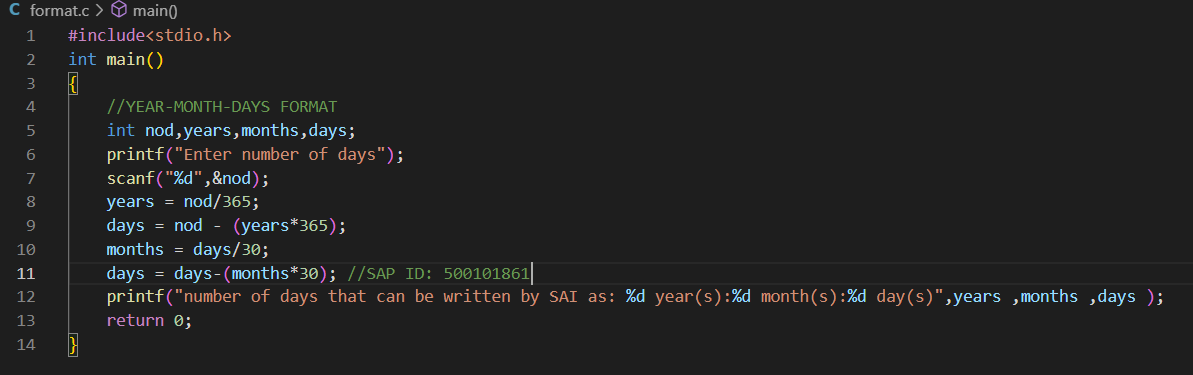
**ALGORITHM:**

Step1: Start  
Step2: Take a value from the user.  
Step3: Now,  
(a) year =+ user\_input/365  
(b) month =+ (user\_input –  
(year\*365))/30  
(c) day =+ (user\_input – (year\*365))%30  
Step4: Display year , month , days  
Step5: Stop

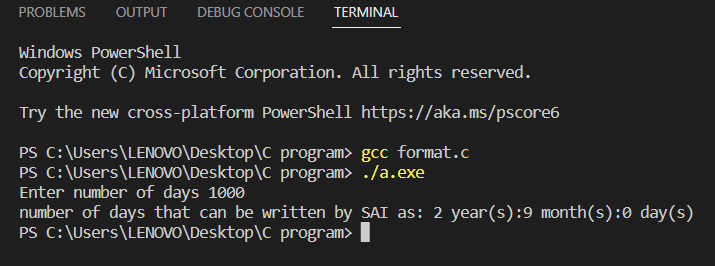
**FLOWCHART:**



**CODE SNIPPET:**



**OUTPUT SNIPPET:**



**EXPERIMENT-4: Conditional Branching**

**Objective:** To understand conditional logic of execution; applying conditional branching structures in C (if, if-else, if-else-if ladder, nested-if, switch-case).

**List of Lab Activities:** Write algorithm and C program, compile, execute and test the code with Linux C compiler with suitable test cases.

1. Find the biggest of 3 numbers.

2. Check whether a given year is leap year or not.

3. Find the roots of a quadratic equation.

4. Check if a given character is a vowel or consonant using Switch-Case statement.

**1. Find the biggest of 3 numbers.**

**ALGORITHM:**

Step1: Input the values of the three numbers saying a,b,c.

Step2: If a>b.

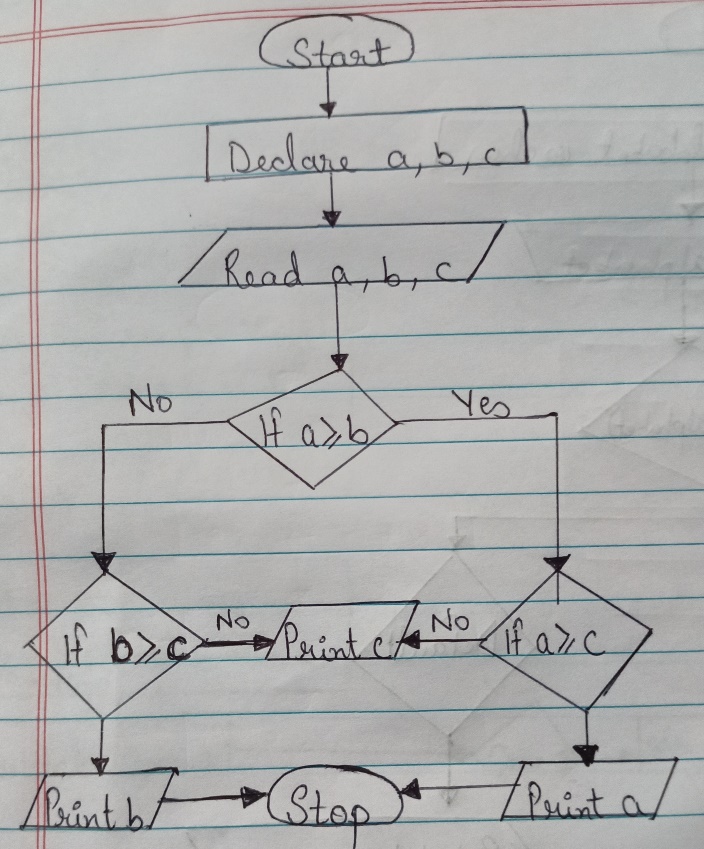
Step3: If yes then check if a>c.

Step4: If yes then a is the largest else c is the largest.

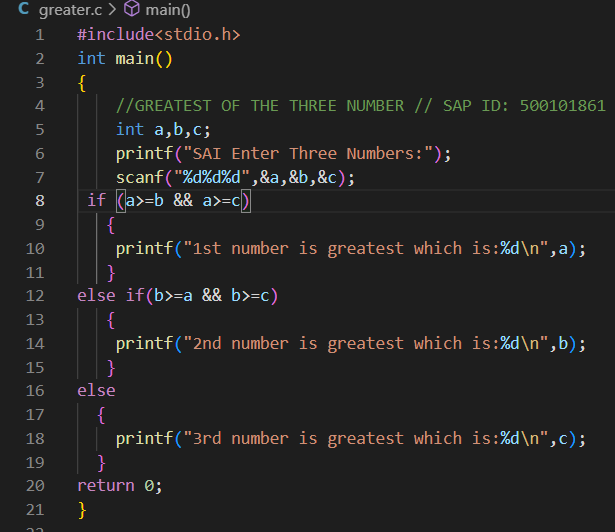
Step5: If a is not greater than b then check b>c or not. Step6: If yes then b is the largest.

Step7: Else c is the largest.

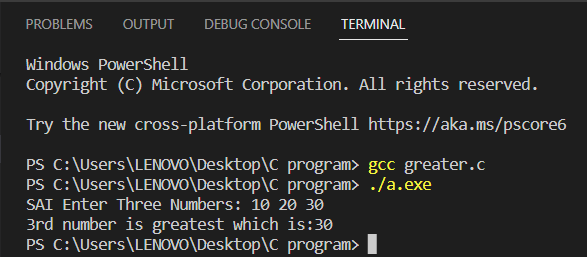
**FLOWCHART:**



**CODE SNIPPET:**



**OUTPUT SNIPPET:**

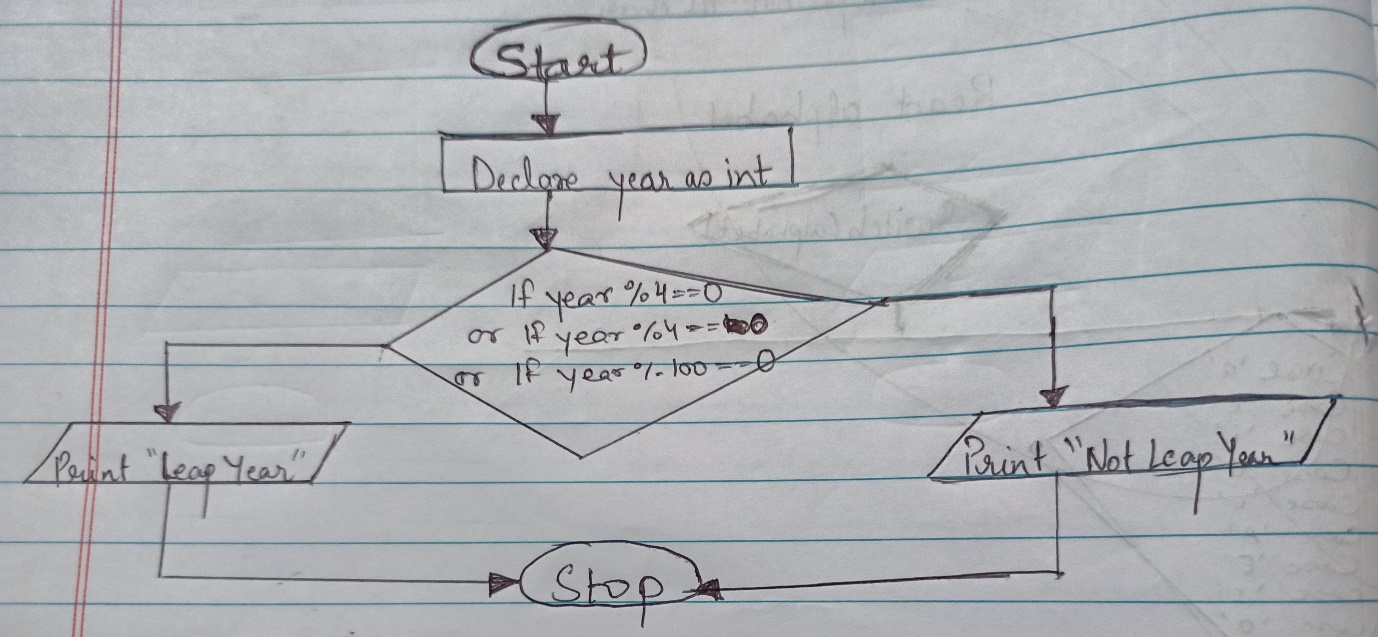


**2. Check whether a given year is leap year or not.**

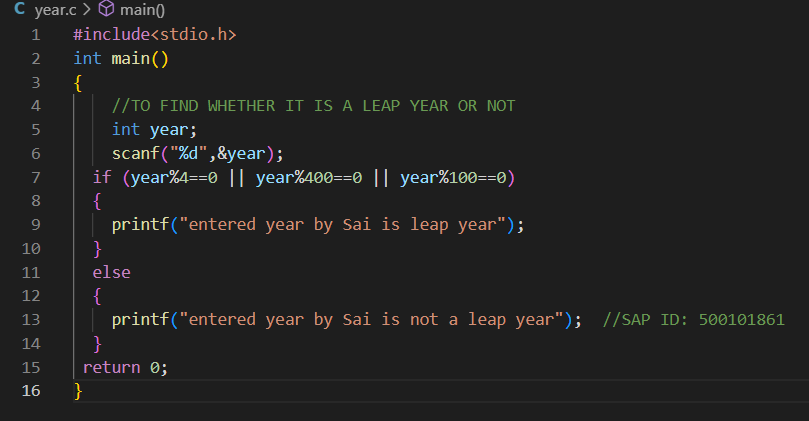
**ALGORITHM:**

Step-1: Start  
Step-2: Input the year  
Step-3: IF (y%100==0 OR y%4==0 OR y%400==0) print y is a leap year ELSE print y is not a leap year  
Step-4: Stop Where, y= year.

**FLOWCHART:**



**CODE SNIPPET:**



**OUTPUT SNIPPET:**

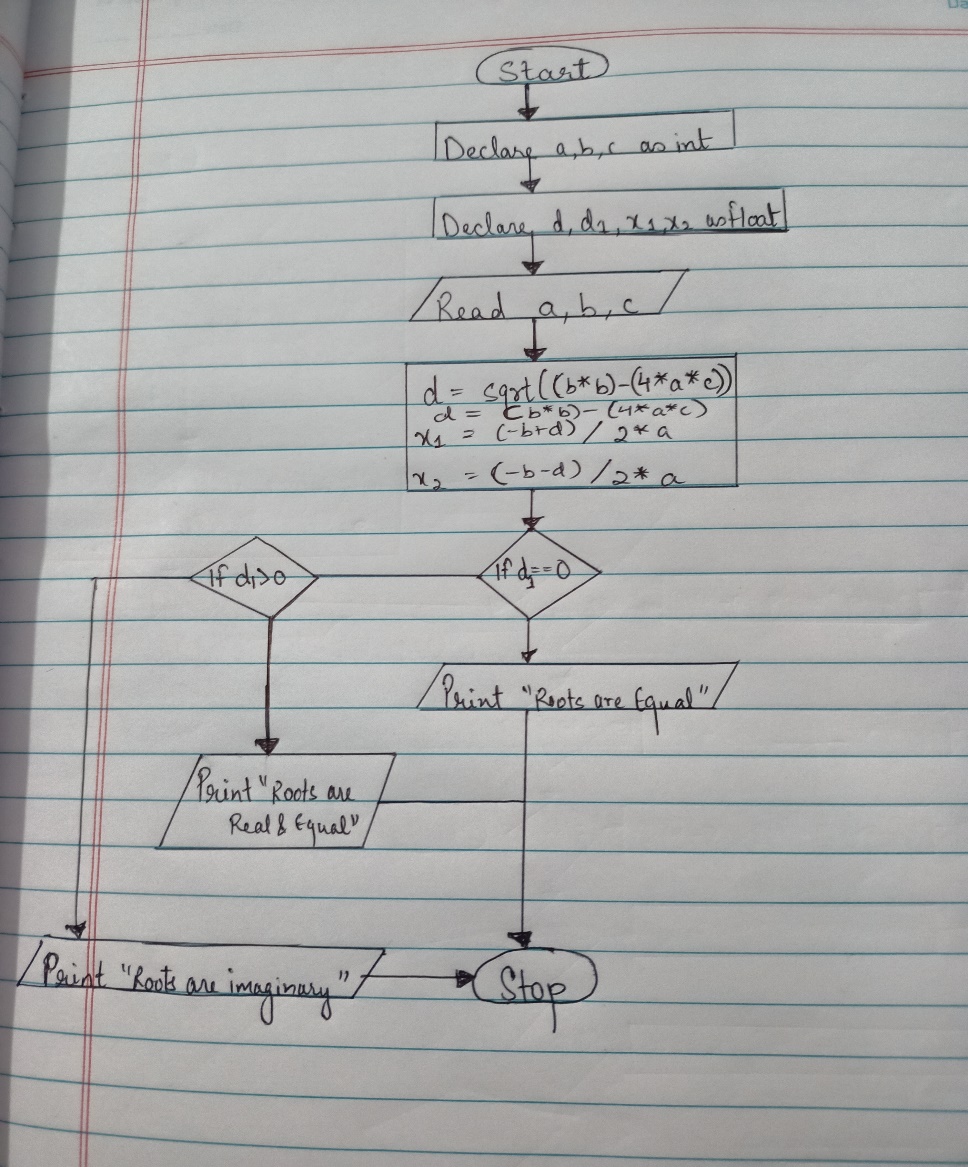


**3. Find the roots of a quadratic equation.**

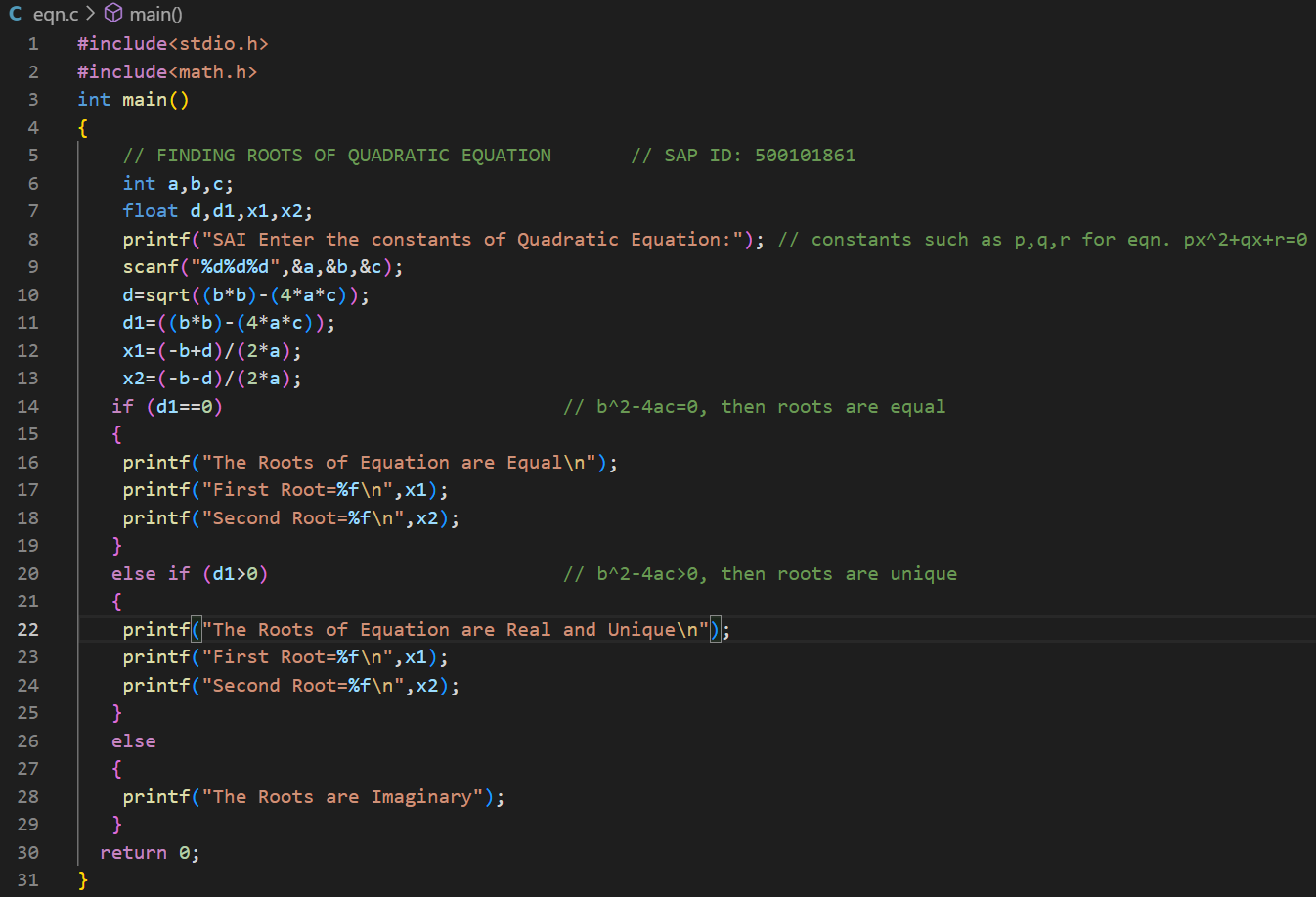
**ALGORITHM:**

Step-1 Start  
Step-2 Enter coefficient of x2 , x and the constant term  
Step-3 Root1= (-b + (b2 -4ac)1/2 )/2a Root2= (-b - (b2 -4ac)1/2 )/2a  
Step-4 Print the roots of the quadratic equation  
Step-5 Stop Where, a= coefficient of x^2 , b= coefficient of x , c= constant term.

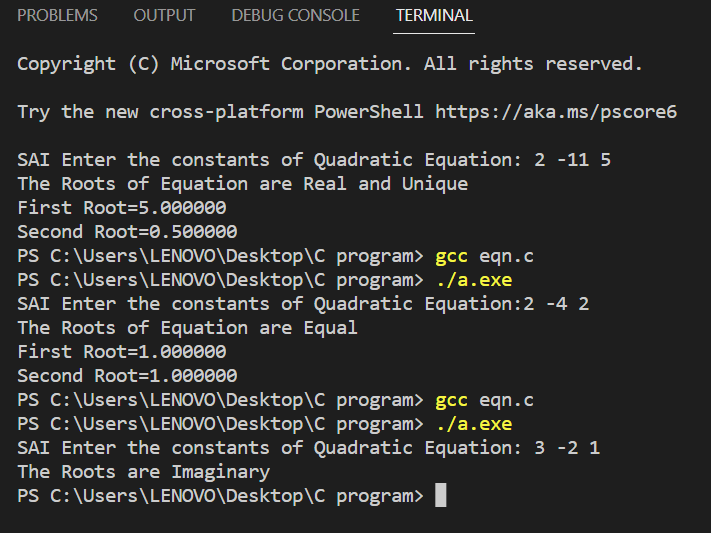
**FLOWCHART:**



**CODE SNIPPET:**



**OUTPUT SNIPPET:**



**4. Check if a given character is a vowel or consonant using Switch-Case statement.**

**ALGORITHM:**

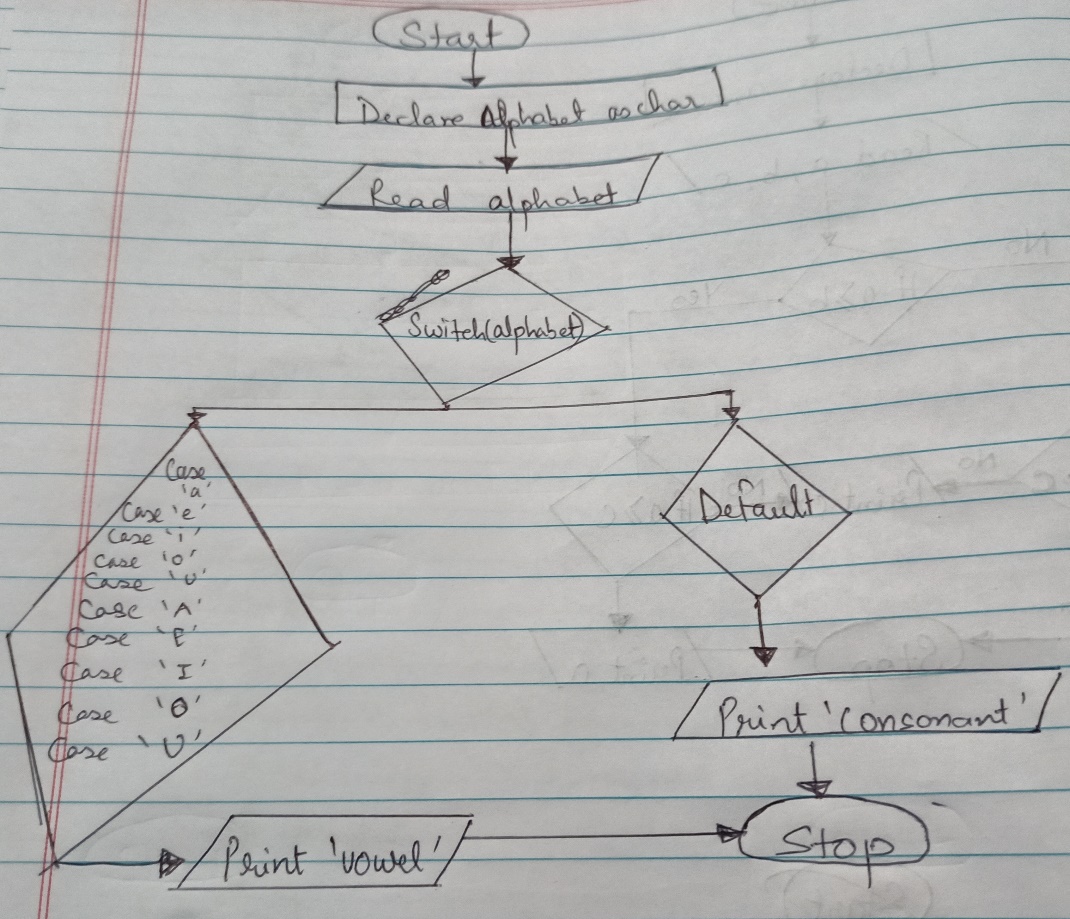
## Step-1: Start

## Step-2: Input a character alphabet.

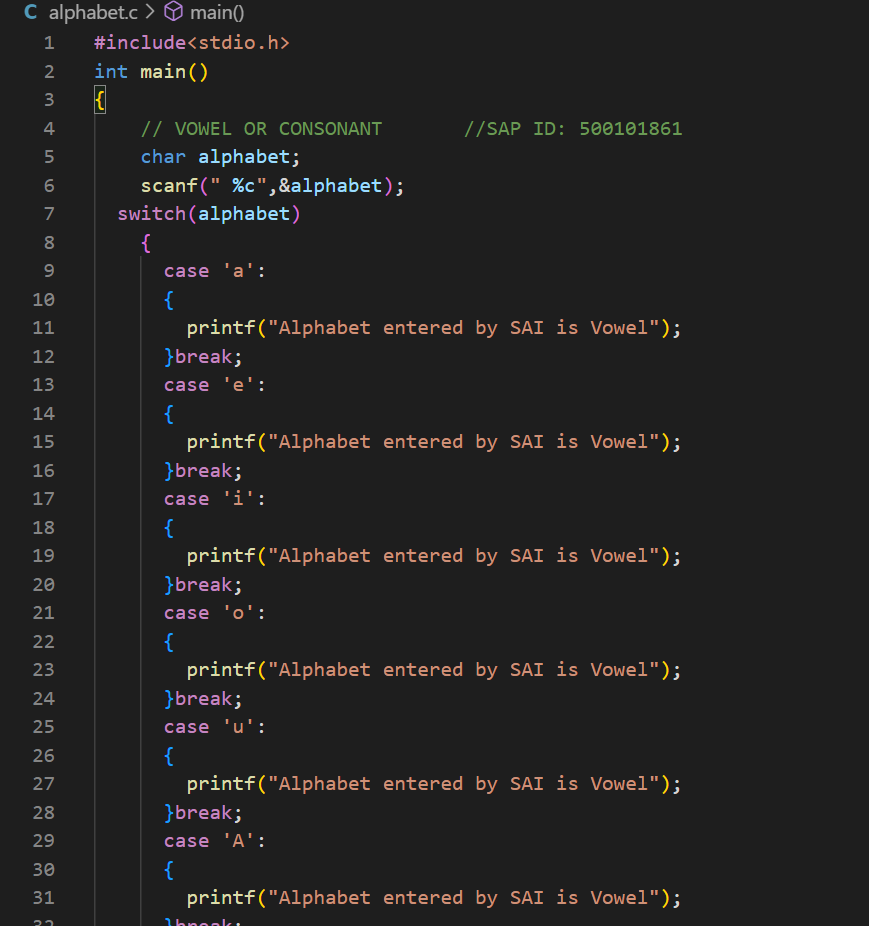
## Step-3: switch (alphabet) case ‘a’: (make similar cases for other vowels as well) print alphabet is a vowel break ; default: print alphabet is a consonant.

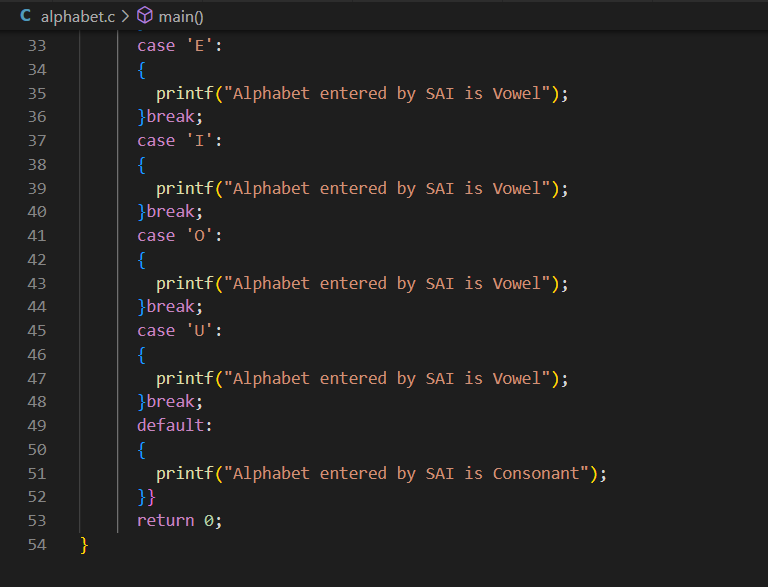
## Step-4: Stop

**FLOWCHART:**

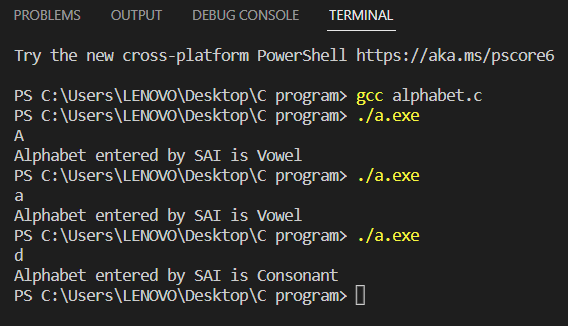


**CODE SNIPPET:**





**OUTPUT SNIPPET:**



**LAB MANUAL ( EXP 5-7)**

BY: SAIRANJAN SUBUDHI

SAP ID: 500101861

SUBMITTED TO: Mrs. KALPANA RANGRA

**EXPERIMENT NO 5**

1. Given positive number ‘n’, generate all the Armstrong numbers between 1 and n. [Hint: A 3-digit number (Ex. 153) is an Armstrong number if the sum of cube of each digit (13+53+33) is equal to 153]

**ALGORITHM:**

STEP1: Start

STEP2: Take input from user in end and also declare num, lastdigit, digits, sum, i, end as int data type.

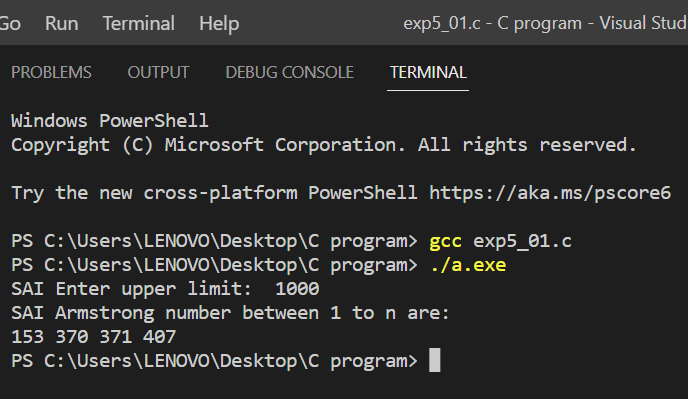
STEP3: Use for loop for evaluating series between 1 and n.

STEP4: The number which will match the condition will be printed.

**CODE SNIPPET:**



**OUTPUT SNIPPET:**



2. Multiple two given numbers without using the arithmetic binary multiplication operator usingfor loop

**ALGORITHM:**

Step1: Start

Step2: Take input from user in a,b and also declare mul and I as int data type.

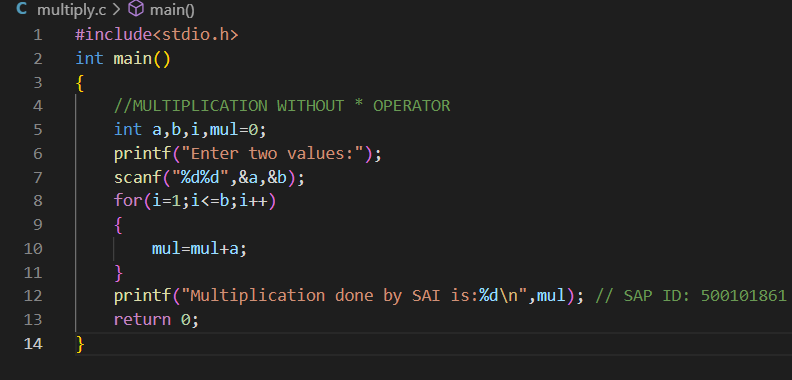
Step3: Take i=1

Step4: If i<=b

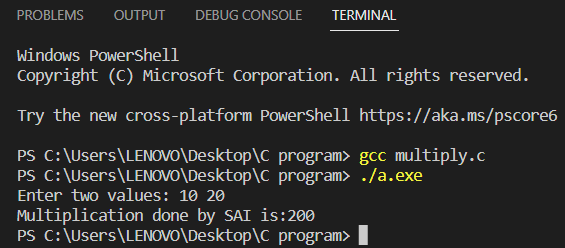
Step5: If the condition is true, then apply mul=mul+a and then print the value of mul else directly print the value of mul.

Step6: Stop

CODE SNIPPET:



OUTPUT SNIPPET:

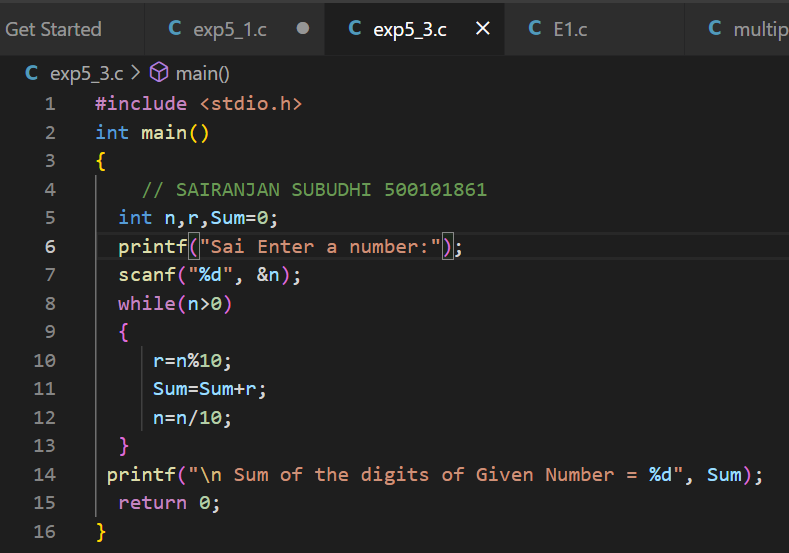


3. Find the sum of digits of a number using while loop

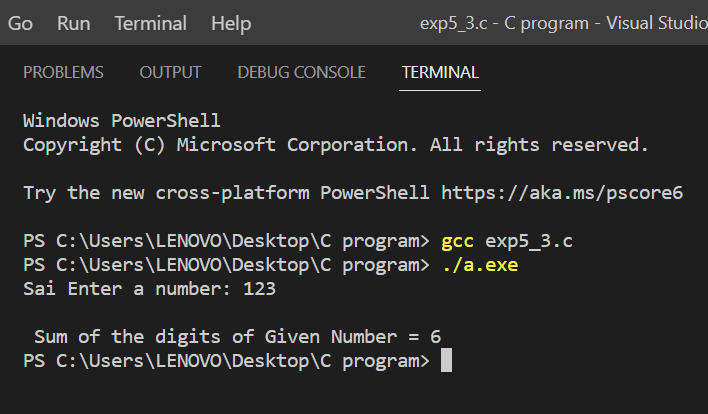
ALGORITHM:

1. Start
2. Take input n , r and sum=0.
3. Using while loop.
4. Apply modulus operator in r.
5. Add it to the sum.
6. Print sum

CODE SNIPPET:



OUTPUT SNIPPET:



4. Given value of ‘n’, find the sum of the series 1+ 1/2 + 1/3 + 1/4 + 1/5 + ...+ 1/n

**ALGORITHM:**

STEP1: Start

STEP2: Declare i, n, a as float data type.

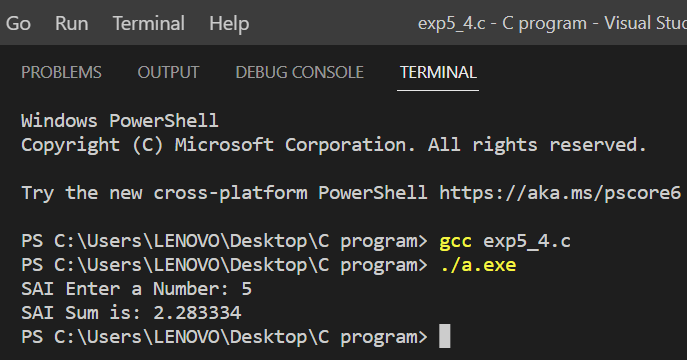
STEP3: Use for loop for executing the series.

STEP4: Print the series.

CODE SNIPPET:



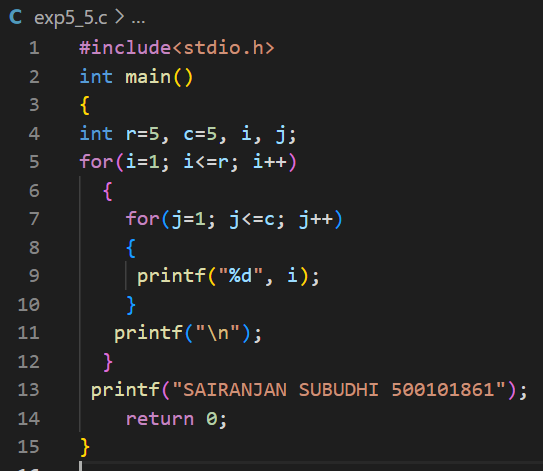
OUTPUT SNIPPET:



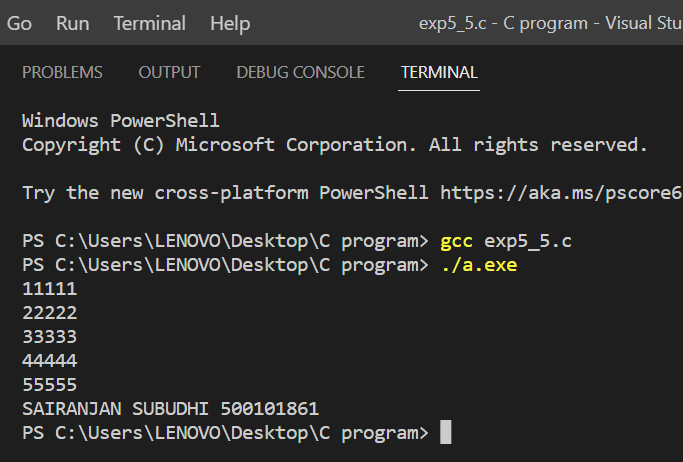
5. Print the given pattern using nested for loop.

ALGORITHM:

1. START
2. TAKE INPUT r, c, i, j.
3. USE FOR LOOP TO ENTER VALUES IN ROWS.
4. USE FOR LOOP TO ENTER VALUES IN COLUMNS.
5. PRINT THE VALUES.

CODE SNIPPET:

OUTPUT SNIPPET:



**EXPERIMENT NO 6**

1. Function main() gets a number and calls the following three functions
   1. “void armstrong(int)” checks if the given number is a Armstrong number or not.
   2. “void coprime(int) reverses the given number and checks if the given number and reversed number are coprime.
   3. “int factorial(int) computes the factorial of the given number using recursion and returns to main().

A)

**ALGORITHM:**

Step 1: START

Step 2: INPUT N

Step 3: SET SUM=0, TEMP=N

Step 4: REM= N%10

Step 5: SUM= SUM+(REM\*REM\*REM)

Step 6: N=N/10

Step 7: REPEAT STEPS 4 TO 6, UNTIL N>0

Step 8: IF SUM=D

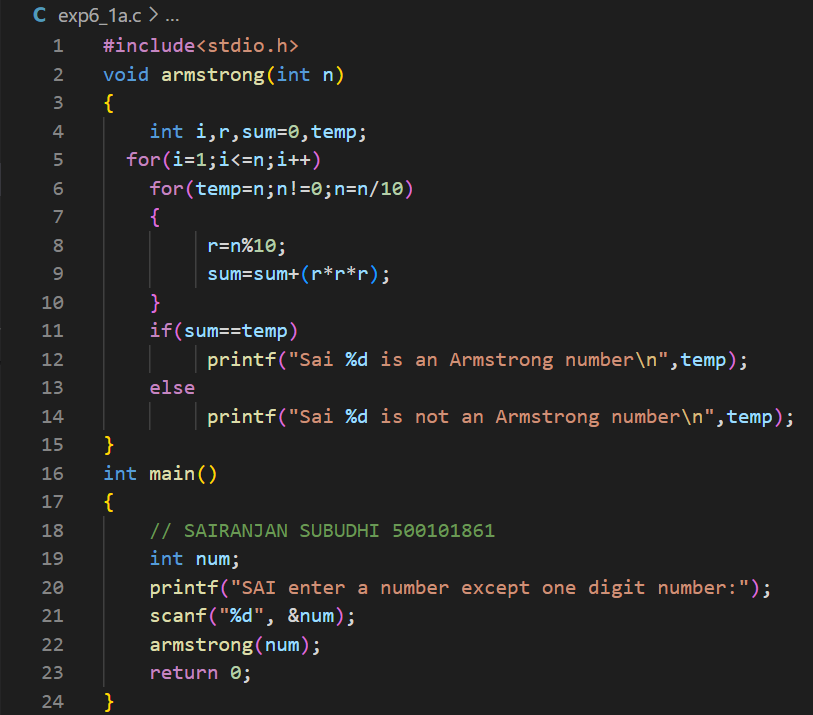
PRINT “NUMBER IS ARMSTRONG”

ELSE

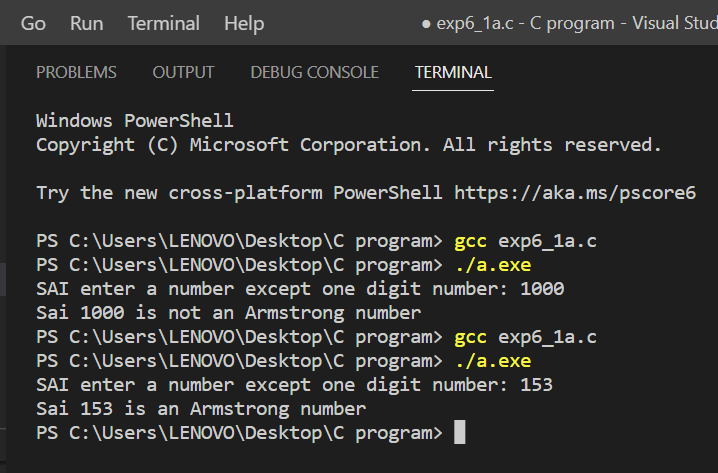
PRINT “NUMBER IS NOT ARMSTRONG”

Step 9: STOP

**CODE SNIPPET:**



**OUTPUT SNIPPET:**



B)

**ALGORITHM:**

STEP1: START

STEP2: Declare function revers(int) and coprime(int,int).

STEP3: In function revers(int), declare lastdigit,t,r=0.

STEP4: Use while loop for reversing the given number. Also declare function coprime(int,int) in it

STEP5: In function coprime(int,int), declare h1,h2 as int data type.

STEP6: Use for loop to test the condition of co prime numbers.

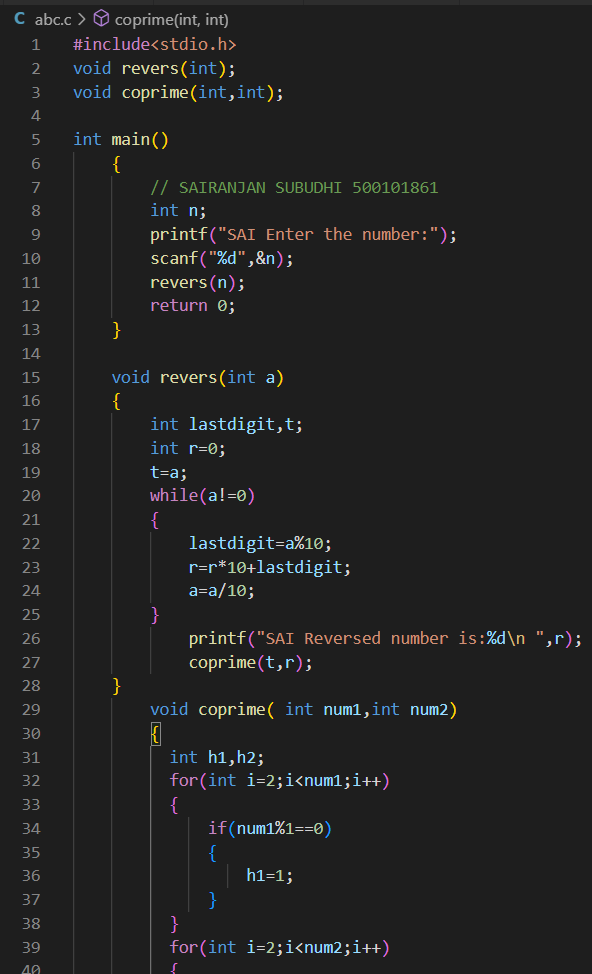
STEP7: In main(), declare n as int data type.

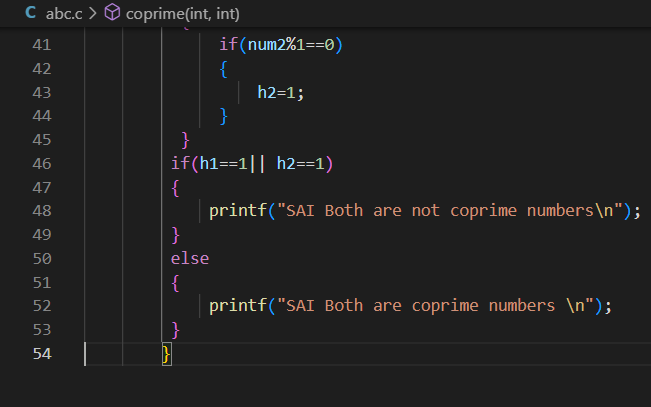
STEP8: Enter the number from the user.

STEP9: Use revers(int) in main().

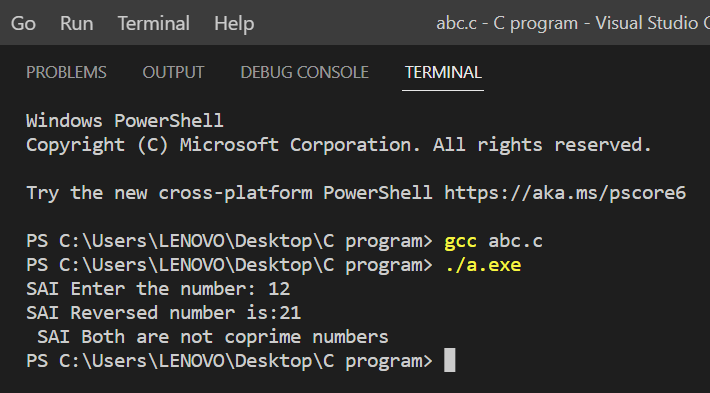
STEP10: END

**CODE SNIPPET:**





**OUTPUT SNIPPET:**



**C)**

**ALGORITHM:**

STEP1: START

STEP2: Declare a function factorial(int).

STEP3: In the function, declarei,f=1 as int data type.

STEP4: Use for loop to find the factorial of the given number by using the condition f=f\*1.

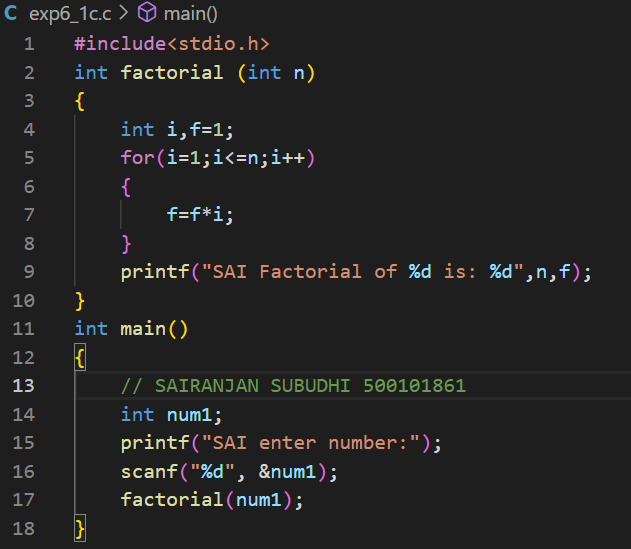
STEP5: In main(), declare num1 as int data type.

STEP6: Enter a number from the user.

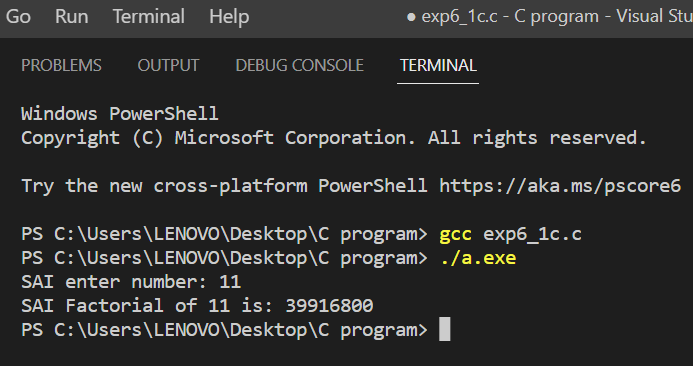
STEP7: Use function factorial(int) in main().

STEP8: END

**CODE SNIPPET:**



OUTPUT SNIPPET:



2. **Function main() gets two numbers from the user and calls three functions in the given order:**

* 1. **“int triangle\_area(int base, int height)” returns the area of the right-angled triangle to main().**
  2. **“void swap(int \*, int\*)” swaps the two numbers using bitwise operator and displays them.**
  3. **“float\* remainder (int a, int b)” returns the remainder of a/b to main().**

A)

**ALGORITHM:**

STEP1: START

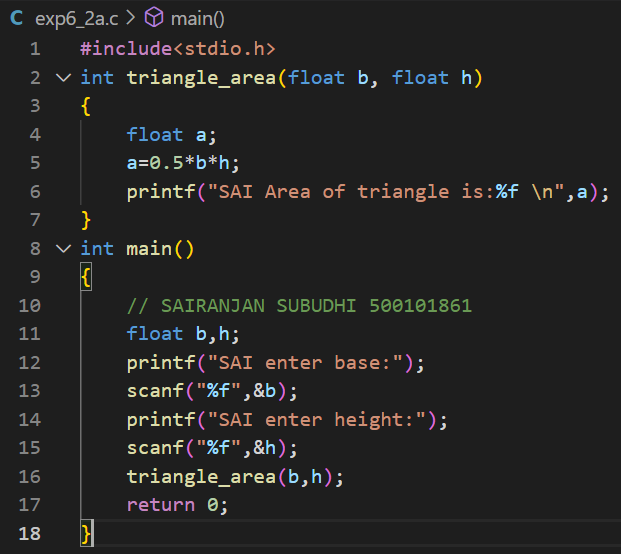
STEP2: Define base and height

STEP3: Apply in the formula.

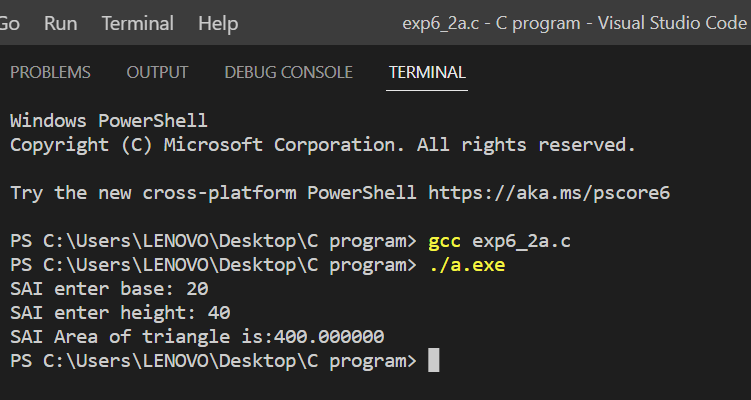
STEP4: Print the Area.

STEP5: END

**CODE SNIPPET:**



OUTPUT SNIPPET:



B)

ALGORITHM:

STEP1: START

STEP2: Declare a function swap(int,int).

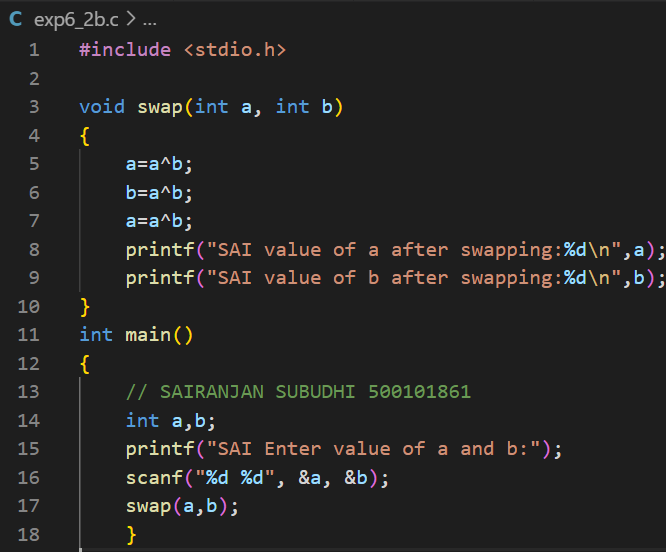
STEP3: Swap the value of and b using bitwise operator

STEP4: Declare a,b as int data type.

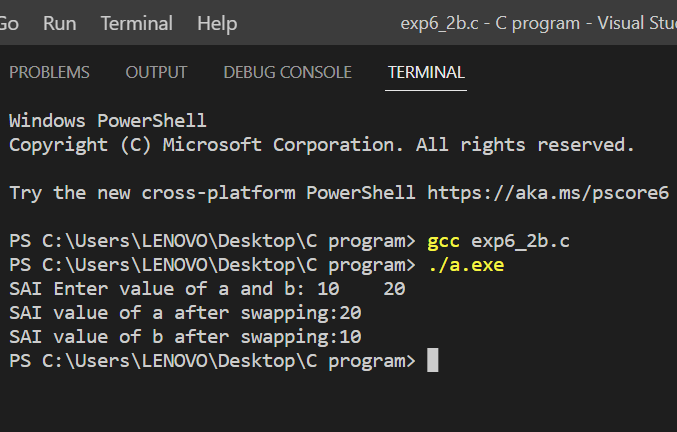
STEP5: Use function swap in main().

STEP6: END

CODE SNIPPET:



OUTPUT SNIPPET:



C)

ALGORITHM:

STEP1: START

STEP2: Declare a function float\_remainder.

STEP3: Declare a variable c as int data type.

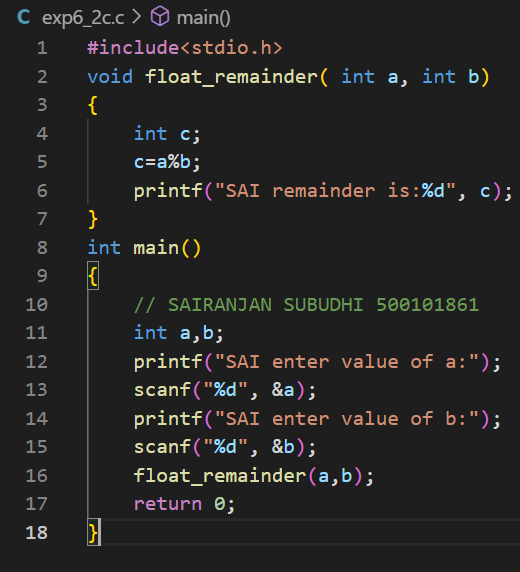
STEP4: c=a%b

STEP5: Enter the value of a and b from the user

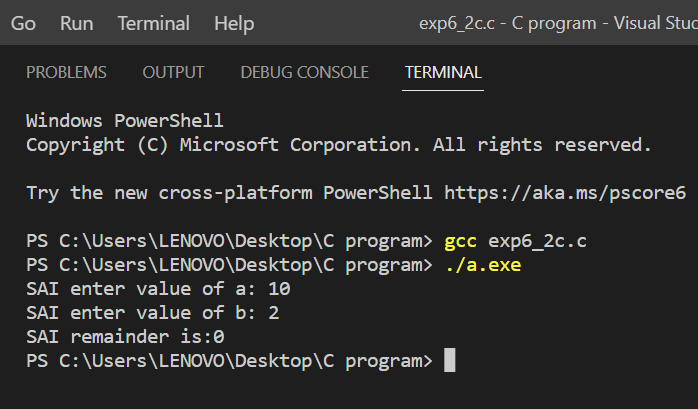
STEP6: Use the function in main().

STEP7: END

CODE SNIPPET:



OUTPUT SNIPPET:

**EXPERIMENT NO 7**

1. Find sum of all array elements using recursion.

ALGORITHM:

STEP1: START

STEP2: Let input a is an integer containing n elements from index 0 to n-1 and last index is an integer variable.

STEP3: Initialize last index with the index of last element in array( last index=n-1).

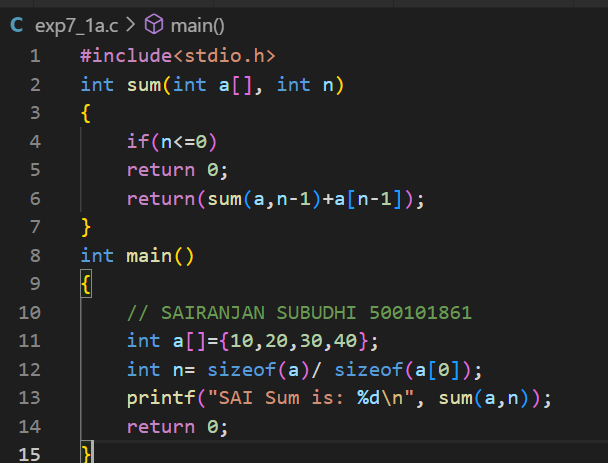
STEP4: We can calculate the sum of input array element from index 0 to n-1, by adding sum of element from 0 to n-2 and input array(n-1).

STEP5: Let getsum (input array, last index) function calculates sum of all elements of input array from index 0 to last index.

STEP6: Recursion will terminate when last index<0.

STEP7: END

CODE SNIPPET:



OUTPUT SNIPPET:



2. **Create an array ‘a1’ with ‘n’ elements. Insert an element in ith position of ‘a1’ and also delete an element from jth position of ‘a1’.**

**ALGORITHM**:

STEP1: START

STEP2: Declare a1[100]={0} as int data type, along with i, pos, x, n=10.

STEP3: for loop is first used for formation of a1[] of size 10.

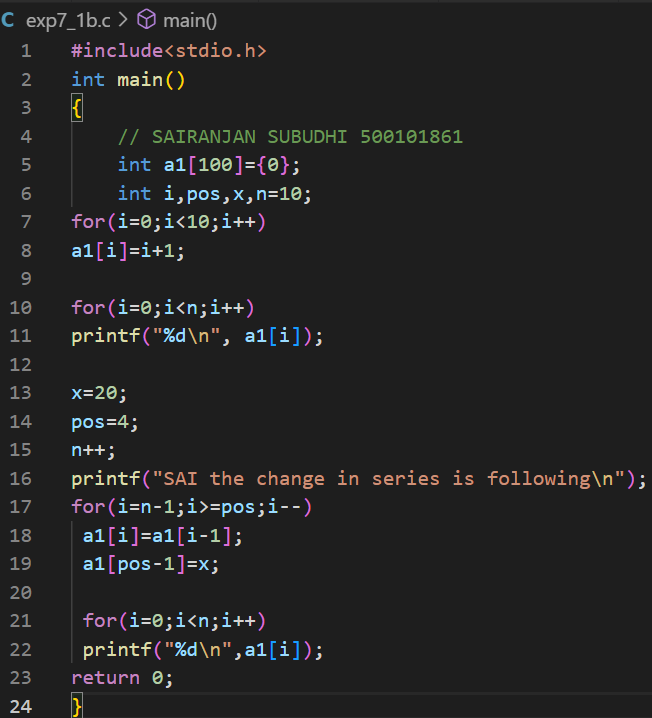
STEP4: Then, for loop is secondly used for entering value in the array a1[].

STEP5: Then, for loop is used for third time for changing the position of element within the array a1[].

STEP6: Then, for loop is used again for entering new element in the array a1[].

STEP7: END

CODE SNIPPET:



OUTPUT SNIPPET:



3. **Convert uppercase string to lowercase using for loop.**

**ALGORITHM**:

STEP1: START

STEP2: Enter the string.

STEP3: for(b=0;a[b];b++)

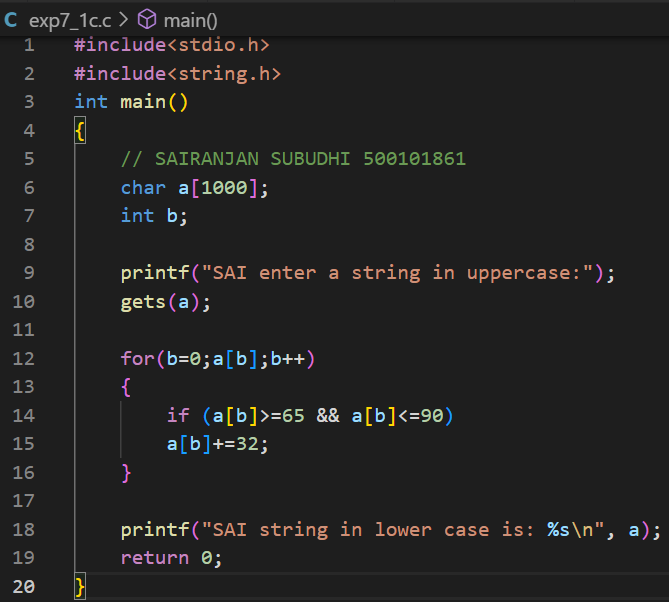
STEP4: if(a[b]>=65 && a[b]<=90)

STEP5: a[b]+=32

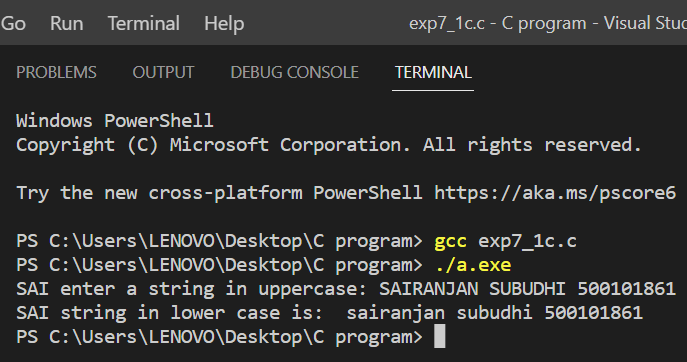
STEP6: Output = a

STEP7: END

CODE SNIPPET:



OUTPUT SNIPPET:



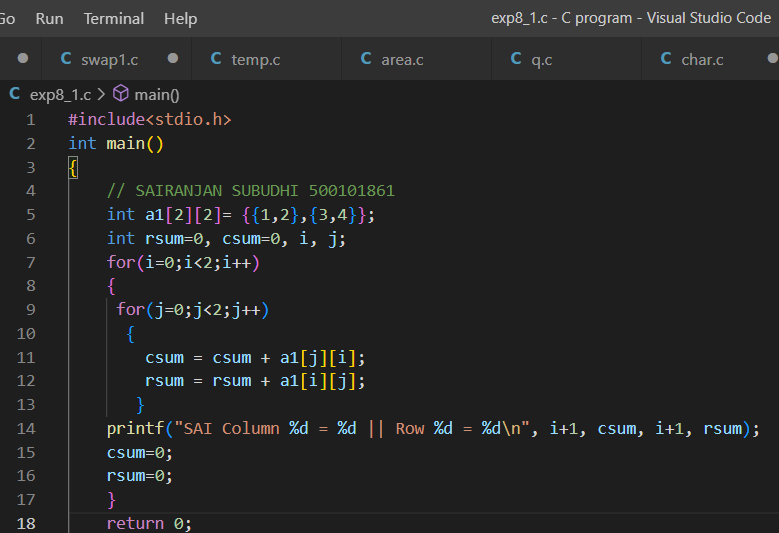
**EXPERIMENT 8-10**

Q8. Write algorithm and C program, compile, execute and test the code with Linux C compiler with suitable test cases.

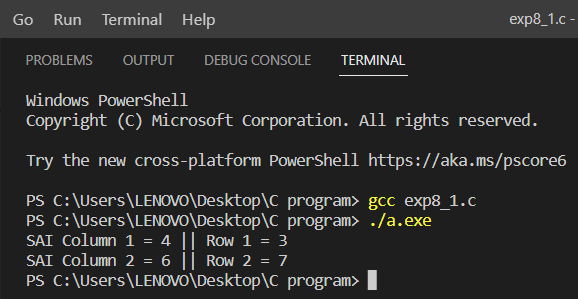
1. Find the sum of rows and columns of matrix of given order.
2. Count how many even numbers are there in a given integer 2D array. [Hint: Linear Search]
3. Store ‘n’ integers in an 1D array in ascending or descending order. Search for a number with binary search technique.

1.

**CODE SNIPPET:**

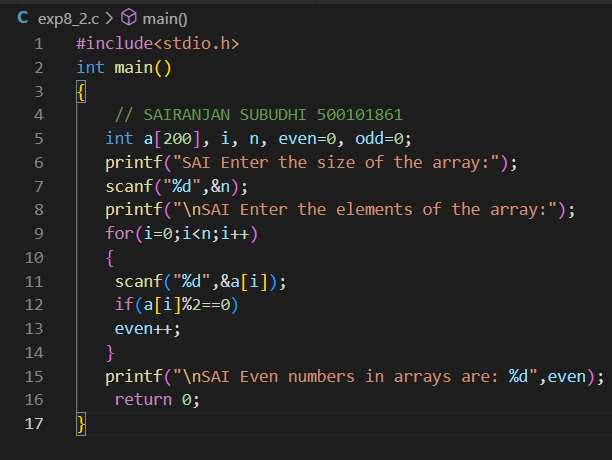


**OUTPUT SNIPPET:**

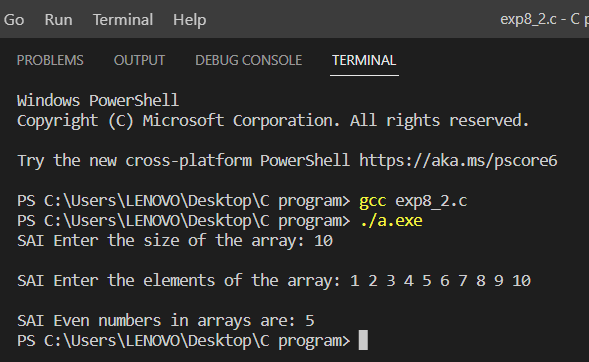


2.

**CODE SNIPPET:**



**OUTPUT SNIPPET:**

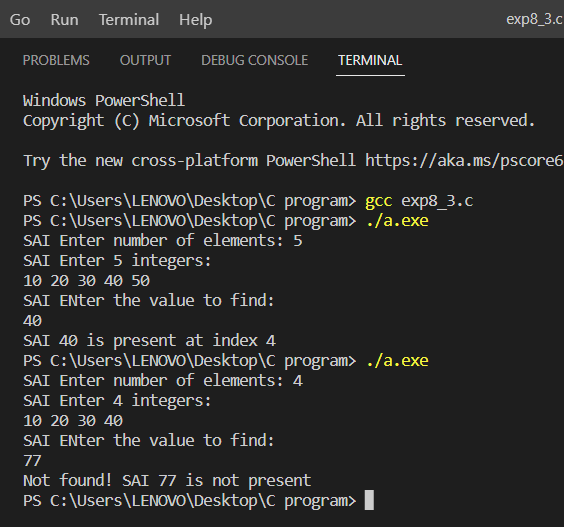


3.

**CODE SNIPPET:**



**OUTPUT SNIPPET:**

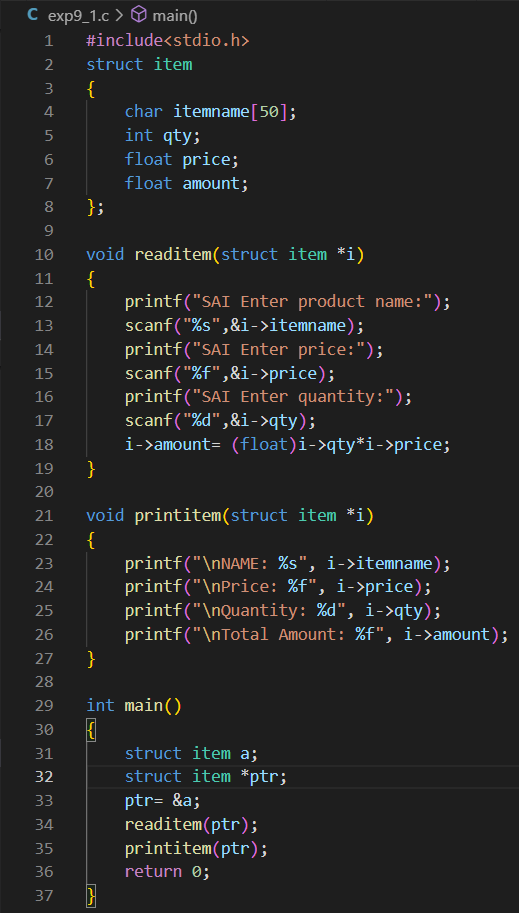


Q9. Write algorithm and C program, compile, execute and test the code with Linux C compiler with suitable test cases.

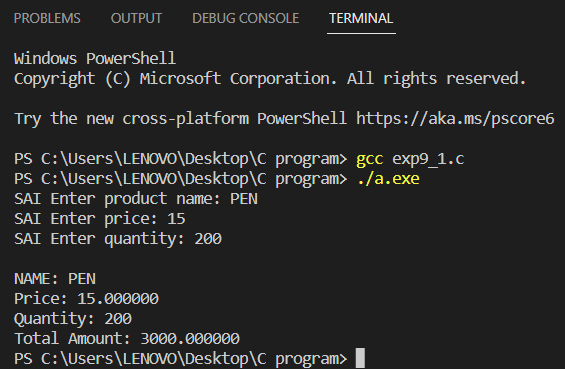
1. Design a structure ‘product’ to store the details of the product purchased like product name, price per unit, number of quantities purchased, and amount spent. Get the name, price per unit, and number of quantities of the product purchased. Calculate the amount spent on the product and then display all the details of the procured product using structure pointers.
2. Design a structure ‘student\_record’ to store student details like name, SAP ID, enrollment number, date of registration and date of birth. The element date of registration is defined using another structure ‘date’ to store date details like day, month, and year. Get data of ‘n’ students and then print the entered values [Hint: Use concept of Nested structures and Array of Structures.]
3. Design a union ‘product’ to store the details of the product purchased like product name, price per unit, number of quantities purchased, and amount spent. Get the name, price per unit, and number of quantities of the product purchased. Calculate the amount spent on the product and then display all the details of the procured product using union pointers.

1.

**CODE SNIPPET:**

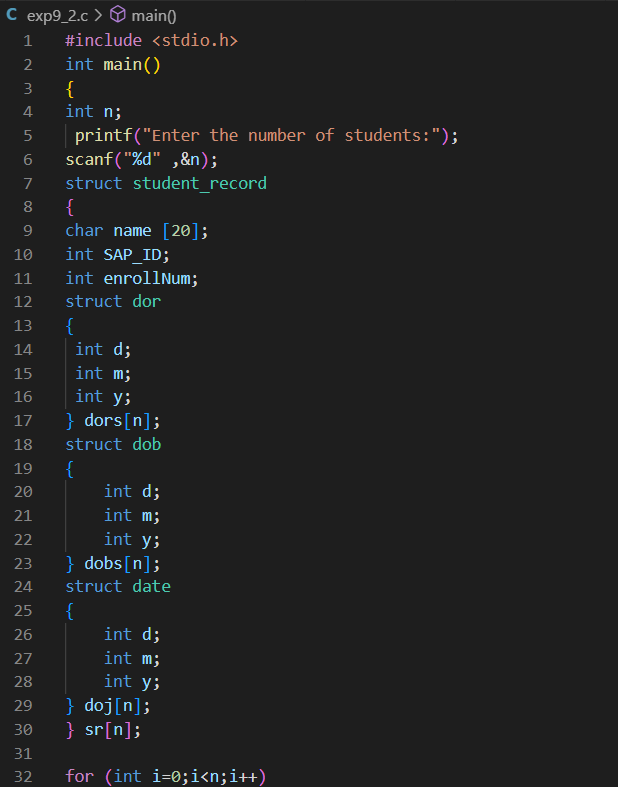


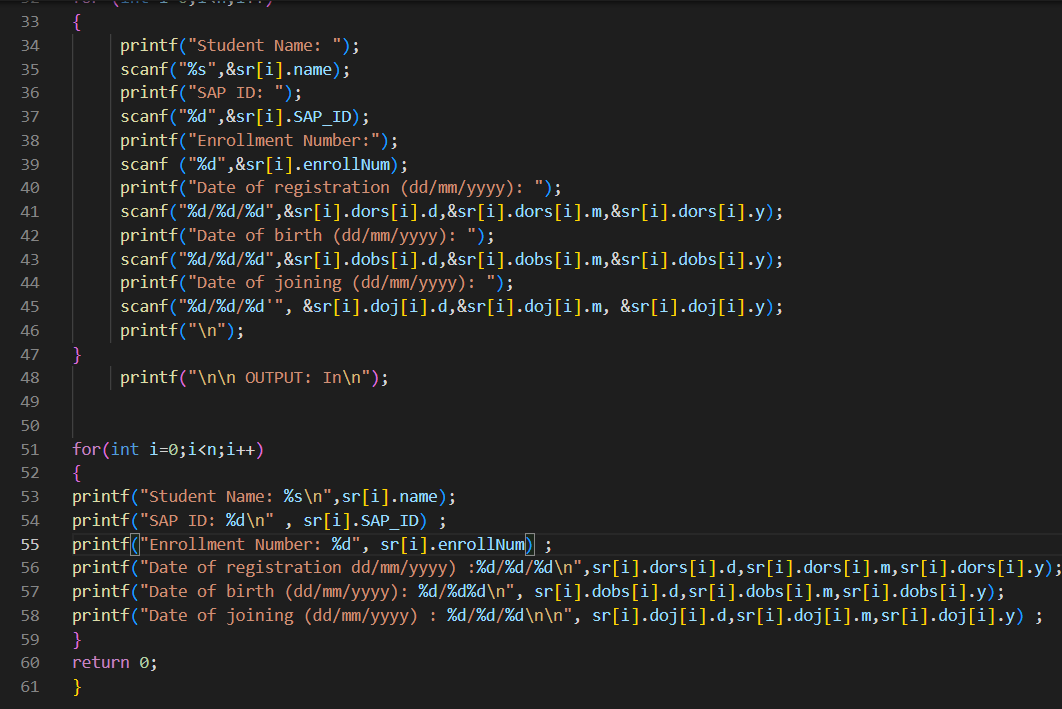
**OUTPUT SNIPPET:**



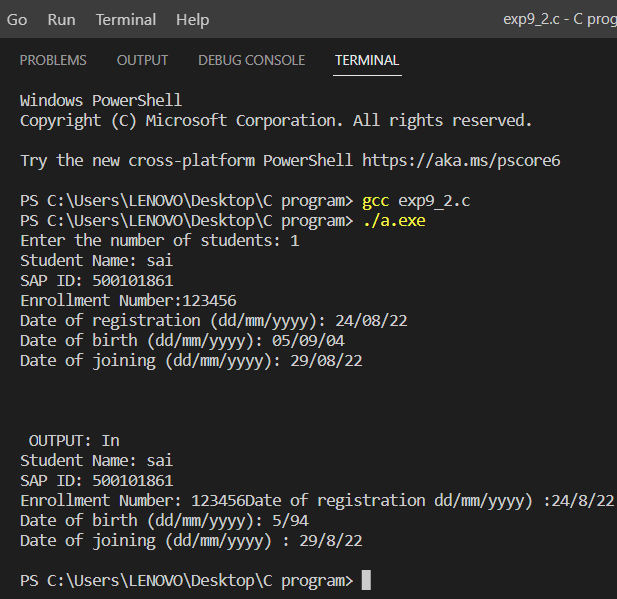
**2.**

**CODE SNIPPET:**



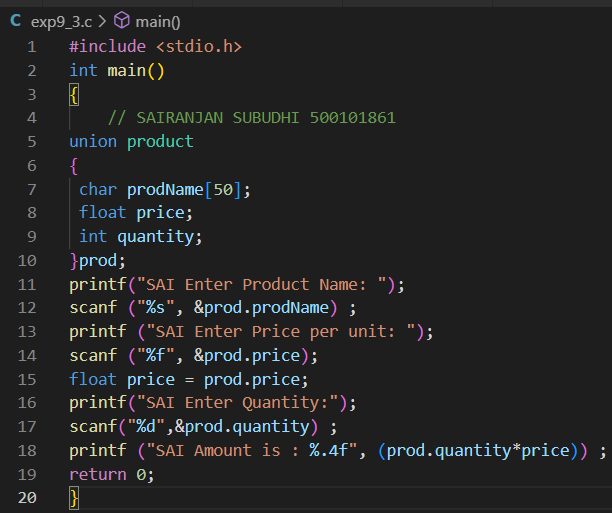


**OUTPUT SNIPPET:**

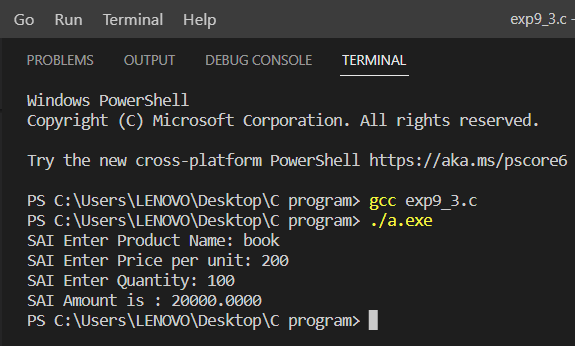


**3.**

**CODE SNIPPET:**



**OUTPUT SNIPPET:**



**Q10.** Write algorithm and C program, compile, execute and test the code with Linux C compiler with suitable test cases.

1. Design a structure ‘subject’ to store the details of the subject (subject name, subject code, etc.). Using structure pointer allocate dynamic memory to structure, and obtain details of ‘n’ subjects with for loop. [Hint: Try to incorporate this in Activity 9.2 with nested structures]

**CODE SNIPPET:**



**OUTPUT SNIPPET:**

