**SCHOOL OF ENGINEERING AND TECHNOLOGY**

**SRI PADMAVATI MAHILA VISVAVIDYALAYAM**

**Tirupati**

***Lab Record***

**Department of Computer Science and Engineering**

****

Name of the Student:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Roll Number:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Year:\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Semester:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Lab Name:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**SRI PADMAVATI MAHILA VISVAVIDYALAYAM,TIRUPATI**

**(Women’s University)**

**SCHOOL OF ENGINEERING & TECHNOLOGY**

****

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**Certificate**

**Roll Number:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Certified that is the bonafide work of**

**Mrs/Miss\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Of the year\_\_\_\_\_\_\_\_\_\_\_sem\_\_\_\_\_\_\_\_\_\_\_\_\_Branch\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**And has conducted \_\_\_\_\_\_\_\_\_\_\_\_\_\_out of\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ experiments in\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Laboratory and has undergone the course of Practical instructions during the Academic year\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Staff – In- charge Head of the Department**

**Date: Examiner**

**SCHOOL OF ENGINEERING AND TECHNOLOGY,SPMVV,TIRUPATI**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| INDEX | | | | | |
| **S.No.** | **Date** | **Name Of The Experiment** | **Page No.** | **Marks** | **Remarks** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**SCHOOL OF ENGINEERING AND TECHNOLOGY,SPMVV,TIRUPATI**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| INDEX | | | | | |
| **S.No.** | **Date** | **Name Of The Experiment** | **Page No.** | **Marks** | **Remarks** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**SCHOOL OF ENGINEERING AND TECHNOLOGY,SPMVV,TIRUPATI**

Arithmetic Operations

Experiment No:1 Roll no:1872035

Date: Page:

Aim: To find Arithmetic operations in C programming

Algorithm:

Step 1: Start.

Step 2: Declare a, b, add, sub, mul,div,mod as integers

Step 3: Read a, b values

Step 4: adda+b, suba-b, mula\*b, diva/b, moda%b

Step 5:print result

Step 6: Stop.

Source Code:

#include<stdio.h>

#include<conio.h>

Void main()

{

Int add,sub,mul,mod;

Float div;

Clrscr();

Printf(“enter a and b values”);

Scanf(“%d%d”,&a,&b);

Add=a+b;

Sub=a-b;

Mul=a\*b;

Div=(float)a/b;

Mod=a%b;

Printf(“\naddition oftwo numbers%d”,add);

Printf(“\ndifference of two nubers%d”,sub);

Printf(“\nproduct of two numbes%d”,mul);

Printf(“\ndivision of two numbers%f”,div);

Printf(“\nmodulus division of two numbers%d”,mod);

Getch();

}

Expected output:

Enter a and b values:100 5

Addition of two numbers 105

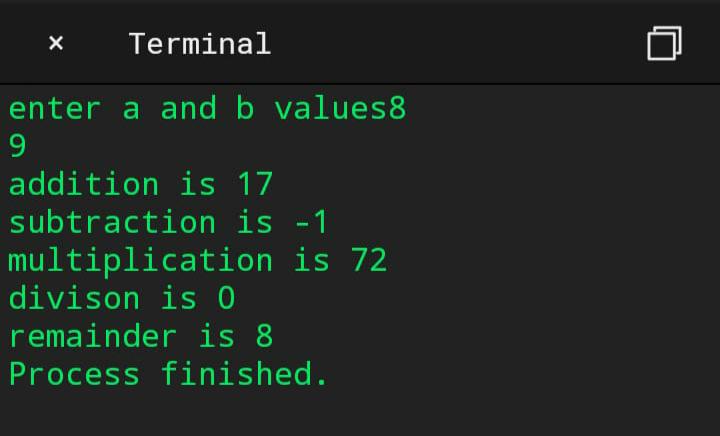
Difference of two numbers 95

Product of two numbers 500

Division of two numbers 20

Modulus division of two numbers 0

Actual Output:



Result: Above the program has been executed successfully.

Simple interest

Experiment no:2 Roll no:1872035

Date: Page:

Aim: To find the simple interest in C programming

Algorithm:

Step 1: Start.

Step 2: Declare amt, rate, time, si as float

Step 3: SI=amt\*rate\*time/100

include Step 5: display SI

Step 6: Stop

Source code:

#include<stdio.h>

#include<conio.h>

Void main()

{

Float amt,rate,time,si;

Clrscr();

Printf(“enter the amount”);

Scanf(“%f”,&amt);

Printf(“enter the rate”);

Scanf(“%f”,&rate);

Printf(“enter the time”);

Scanf(“%f”,&time);

Si=(amt\*rate\*time)/100;

Printf(“simple interest =%f”,si);

Getch();

}

EXPECTED OUTPUT:

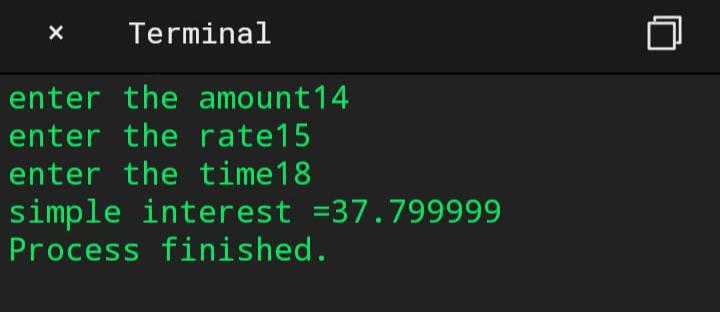
Enter the amount:10

Enter the rate:10

Enter the time:10

Si=10.000

ACTUAL OUTPUT:



RESULT: above the programme has been executed successfully.

Compound interest

Experiment No:3 Roll no:1872035

Date: Page:

Aim: To find the Compound interst in C programming

Algorithm:

Step 1: Start.

Step 2: Declare principle,time,rate and compound

Step 3: input principle, time, rate

Step 4: CI=p\*pow((1+R\100),t)

Step 5: display CI

Step 6: Stop

Source code:

#include<stdio.h>

#include<conio.h>

#include<math.h>

Int main()

{

Int p,t;

Float r,ci;

Printf(“enter principal amount\n”);

Scanf(“%d”,&p);

Printf(“enter rate of interest\n”);

Scanf(“%f”,&r);

Printf(“enter time period\n”);

Scanf(“%d”,&t);

Ci= p\*pow((1+r\100),t);

Printf(“compound interest is %f\n”,ci);

Return 0;

}

EXPECTED OUTPUT:

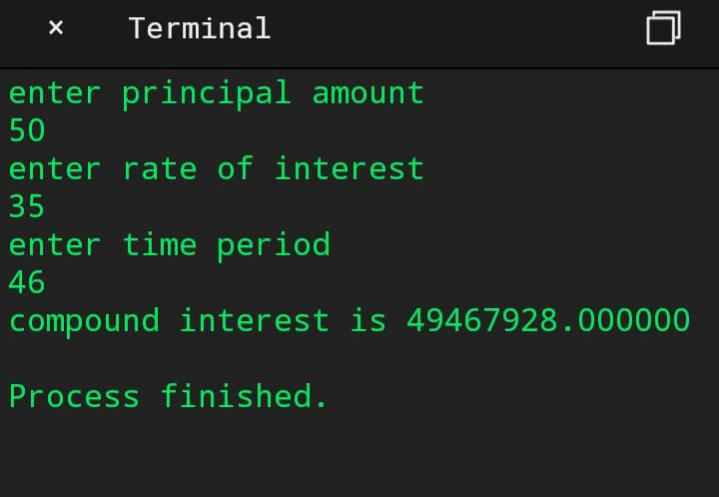
Enter principal amount=1000

Enter rate=5

Enter time period=3

Compound interest is 1157.28

ACTUAL OUTPUT:



RESULT: above the program has been executed successfully

AREA OF A CIRCLE

Experiment No:4 Roll no:1872035

Date: Page:

Aim: To find the area of circle in C programming

Algorithm:

Step 1: Start.

Step 2: Declare r, pi=3.14, n as float values

Step 3: Read r

Step 4:npi\*r\*r

Step 5:print area of the circle

Step 6: Stop

Source Code:

#include<stdio.h>

#include<conio.h>

#include<math.h>

Int main()

{

Int r;

Float pi=3.14,a;

clrscr();

printf(“enter r value”);

Scanf(“%d”,&r);

A=(pi\*r\*r);

Printf(“area of the circle %f”,a);

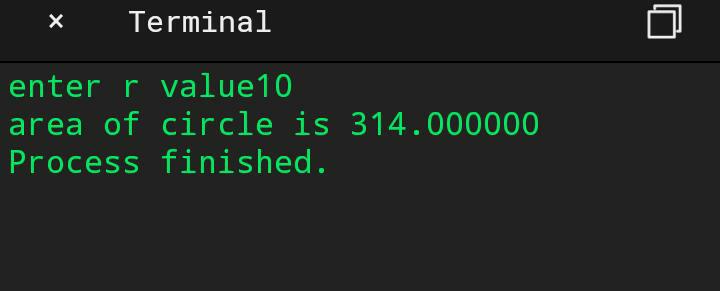
return 0;

}

Expected output:

Enter the radius:25

Area of circle 1962.5001

ACTUAL OUTPUT:

RESULT: Above programme has been executed successfully

To read in two integers and display one as percentage of the other.

Exp no:5. Roll no:1872035

Date: page no:

Aim: To write a c program to read in two integers and display one as a percentage of the other. Typically, your output should look like 20 is 50.00% of 40 assuming that the input numbers were 20 and 40 display the percentage correct to 2 decimals.

Algorithm:

Step 1: Start.

Step 2: Declare two integers and one float variable to store the output.

Step 3: check the condition if a<b or a>b and find out percentage a/b\*100 and convert it into float value.

Step 4: store the value and print it.

Step 5: Stop.

Source code:

#include<stdio.h>

#include<conio.h>

Void main()

{

Int a,b;

Float c,d,e;

Clrscr();

Printf(“enter a and b values:”);

Scanf(“%d %d”,&a,&b);

If(a<b)

{

C=(float)a/b\*100;

Printf(“%d is %.2f%% of %d”,a,c,b);

}

Else

{

C=(float)b/a\*100;

Printf(“%d is %.2f of %d”,b,c,a);

}

Getch();

}

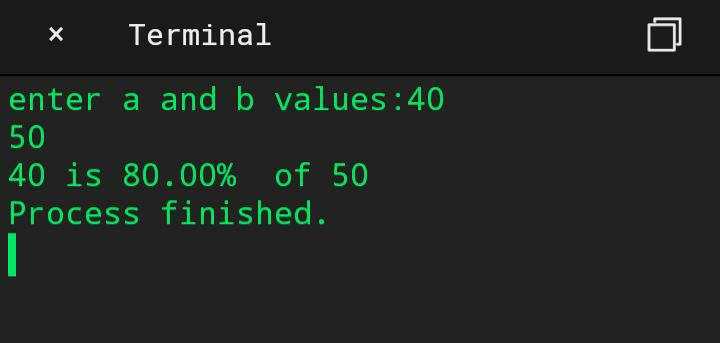
Expected output:

Enter a and b values:20

40

20 is 50.00% of 40

ACTUAL OUTPUT:



RESULT: Above programme has been executed successfully

To output the possible roots of a given set of coefficients.

Exp no:6. Roll no: 1872035

Date: page no:

Aim: To write a c program to output the possible roots of a given set of coefficients with appropriate messages.

Algorithm:

Step 1: Start.

Step 2: Read a, b, c as integer.

Step 3: declare root1, root2 as float.

Step 4: if(b\*b>4\*a\*c)

Print “the roots are real roots”

Assign root1 ((-b/2\*a) + sqrt(b\*b-4\*a\*c)/2\*a)

Assign root2((-b/2\*a) – sqrt(b\*b-4\*a\*c)/2\*a)

Print root1, root2

Else if(b\*b=4\*a\*c)

Print “the roots are repeated roots”

Assign root1-b/2\*a

Print root1

Else

Print “the roots are imaginary”

Step 5: Stop.

Source code:

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#include<conio.h>

Void main()

{

Float a, b, c, root1, root2;

Float realp, imagp;

Clrscr();

Printf(“Enter the values of a, b and c \n”);

Scanf(“%f %f %f”, &a, &b, &c);

/\* If a = 0, it is not a quadratic equation \*/

If (a == 0 || b == 0 || c == 0)

{

Printf(“Error: Roots cannot be determined \n”);

Exit(1);

}

Else

{

If (b\*b-4.0\*a\*c< 0)

{

Printf(“Imaginary Roots\n”);

Realp = -b / (2.0 \* a) ;

Imagp = sqrt(abs(b\*b-4.0\*a\*c)) / (2.0 \* a);

Printf(“Root1 = %f +i %f\n”, realp, imagp);

Printf(“Root2 = %f -i %f\n”, realp, imagp);

}

Else if (b\*b-4.0\*a\*c== 0)

{

Printf(“Roots are real and equal\n”);

Root1 = -b / (2.0 \* a);

Root2 = root1;

Printf(“Root1 = %f\n”, root1);

Printf(“Root2 = %f\n”, root2);

}

Else if (b\*b-4.0\*a\*c > 0 )

{

Printf(“Roots are real and distinct \n”);

Root1 =(-b + sqrt(b\*b-4.0\*a\*c)) / (2.0 \* a);

Root2 =(-b – sqrt(b\*b-4.0\*a\*c)) / (2.0 \* a);

Printf(“Root1 = %f \n”, root1);

Printf(“Root2 = %f \n”, root2);

}

}

Getch();

}

Expected output:

Enter the values of a, b and c 1

1

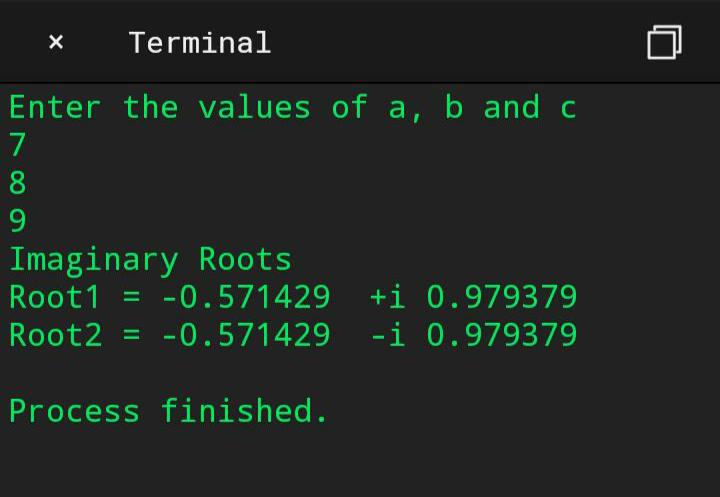
6

Roots are real and distinct

Root1=-2.000000

Root2=-3.000000.

ACTUAL OUTPUT:



RESULT: Above programme has been executed successfully

DATE MONTH YEAR

Experiment no: Roll no:1872035

Date: Page no:

Aim: To write a c program taken input of date as day, month, year and print the output of the following date.

Algorithm:

Step 1: Start.

Step 2: Declare variables, day, month, year.

Step 3: enter date.

Step 4: check condition for even month or odd month.

Step 5: if it is odd month check condition

If(day==1&&day<=30) then day=day+1, else

Month=month+1&day=1.

Step 6: if it is even month but not equal to 2 then check condition if(day<=1&&day<=29)

Then day=day+1, else month=month+1 & day=1.

Step 7: if month==2 then check for leap year, if it is leap year then check condition

If(day==1&&day<=28) then day=day+1 else month=month+1 & day=1

Else check condition if(day==1&&day<=27) then day=day+1

Else month=month+1 & day=1.

Step 8: print date.

Step 9: Stop.

Source code:

#include<stdio.h>

#include<conio.h>

Void main()

{

Int mon,day,year;

Clrscr();

Printf(“enter date:”);

Scanf(“%d%d%d”,&day,&mon,&year);

If(mon==1||mon==3||mon==5||mon==7||mon==8||mon==10||mon==12)

{

If((day==1)&&(day<=30))

{

Day=day+1;

}

Else

{

Day=1;

Mon=mon+1;

}

}

If(mon==4||mon==6||mon==9||mon==11)

{

If((day==1)&&(day<=29))

{

Day=day+1;

}

Else

{

Day=1;

Mon=mon+1;

}

}

If(mon==2)

{

If((year%4)==0)

{

If((day==1)&&(day<=28))

{

Day=day+1;

}

Else

{

Day=1;

Mon=mon+1;

}

}

Else

{

If((day==1)&&(day<=27))

{

Day=day+1;

}

Else

{

Day=1;

Mon=mon+1;

}

}

}

If((day==31)&&(mon==12))

{

Year=year+1;

}

Printf(“the following date is %d:%d:%d”,day,mon,year);

Getch();

}

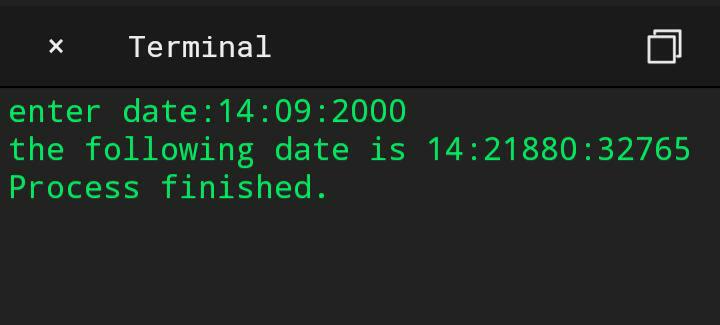
Expected output:

Enter date:

28 02 1992

The following date is 01:03:1992.

Actual output:



RESULT: Above programme has been executed successfully

Amicable or not

Experiment No:8 Roll no: 1872035

Date: page:

Aim: To write a c program to check whether given two numbers are amicable or not.

Algorithm:

Step 1: Start.

Step 2: declare num1, num2.

Step 3: initialize num1divisorsum=0 and num2divisorsum=0

Step 4: for(int i=1;i<num1;i++)

{

If(num1%i==0)

{

Num1divisorsum=num1divisorsum+I;

}

}

Step 5: for(int i=1;i<num2;i++)

{

If(num2%i==0) {

Num2divisorsum=nm2divisorsum+I;

}

}

Step 6: if((num1==num2divisorsum)&&(num2==num1divisorsum)) is true go to step 7, else

Go to step 8.

Step 7: print the two numbers are amicable.

St3p 8: print the two numbers are not amicable.

Step 9: Stop.

Source code:

#include<stdio.h>

#include<conio.h>

Int main()

{

Int I,num1,num2,num1divisorsum=0, num2divisorsum=0;

Clrscr();

Printf(“enter two numbers to check if amicable or not :”);

Scanf(“%d %d”,&num1,&num2);

For(i=1;i<num1;i++)

{

If(num1%i==0)

{

Num1divisorsum=num1divisorsum+I;

}

}

For(i=1;i<num2;i++)

{

If(num2%i==0)

{

Num2divisorsum=num2divisorsum+I;

}

} If((num1==num2divisorsum)&&(num2==num1divisorsum))

{

Printf(“%d and %d are amicable numbers\n”,num1,num2);

}

Else

{

Printf(“%d and %d are not amicable numbers\n”,num1,num2);

}

Getch();

Return 0;

}

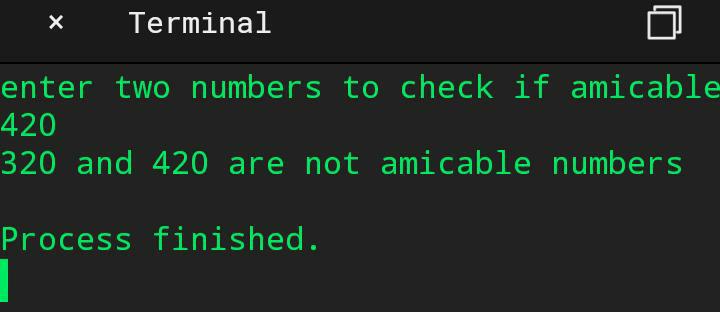
Expected output:

Enter two numbers to check if amicable or not:220

284

220 and 284 are amicable numbers.

Actual output:



RESULT: Above Programme has been executed successfully

STUDENTS MARK GRADING

Experiment No:9 Roll no:1872035

Date: Page no:

Aim: To write a c program for students mark grading.

Algorithm:

Step 1: Start.

Step 2: declare a,b,c,d,tot,avg.

Step 3: enter marks in all subjects.

Step 4: if(a>34&&b>34&&c>34&&d>34) is true then print student gets passed else go to

Step 5.

Step 5: print student is failed.

Step 6: calculate tot = a + b + c + d and avg =(float)tot/4.

Step 7: if(avg>=90) is true then print student got grade A else go to step 8.

Step 8: else if(avg>=80&&avg<90) then Print student got grade: A+.

Step 9: else if(avg>=70&&avg<80) then Print student got grade B.

Step 10: else if(avg>=60&&avg<70) then Print student got grade B+.

Step 11: else if(avg>=50&&avg<60) then Print student got grade C.

Step 12: else if(avg>=40&&avg<50) then Print student got grade D.

Step 13: else if(avg>=35&&avg<40) then Print student got grade E.

Step 14: else Print student got grade F.

Step 15: Print the average of a student.

Step 16: stop.

Source code:

#include<stdio.h>

#include<conio.h>

Void main()

{

Int a,b,c,d,tot;

Float avg;

Clrscr();

Printf(“enter marks in all subjects”);

Scanf(“%d %d %d %d”,&a,&b,&c,&d);

If(a>34&&b>34&&c>34&&d>34)

{

Printf(“student gets passed\n”);

}

Else

{

Printf(“student get failed\n”);

}

Tot=a+b+c+d;

Avg=(float)tot/4;

If(avg>=90)

{

Printf(“student got grade:’A’”);

}

Else if(avg>=80&&avg<90)

{

Printf(“student got grade:’A+’”);

}

Else if(avg>=70&&avg<80)

{

Printf(“student got grade:’B’”);

}

Else if(avg>=60&&avg<70)

{

Printf(“student got grade:’B+’”);

}

Else if(avg>=50&&avg<60)

{

Printf(“student got grade:’C’”);

}

Else if(avg>=40&&avg<50)

{

Printf(“student got grade:’D’”);

}

Else if(avg>=35&&avg<40)

{

Printf(“student got grade:’E’”);

}

Else

{

Printf(“student got grade:’F’”);

}

Printf(“\nthe average of a student is %f”,avg);

Getch();

}

Expected output:

Enter marks in all subjects

90

80

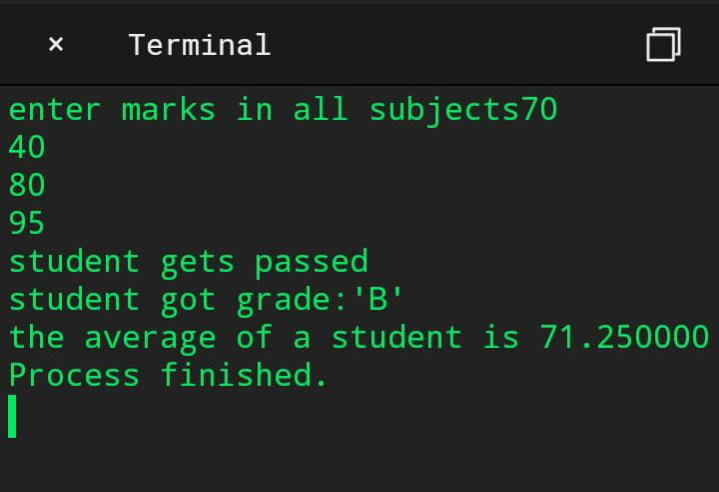
78

85

Student get passed

Student got grade A++

The average of student is 83.25

ACTUAL OUTPUT:

RESULT: above programme has been executed successfully

Palindrome or not

Experiment No:10 Roll no: 1872035

Date: page no:

Aim: To write a c program that takes an integer number as input, check whether it is palindrome or not and output the reverse of the same with suitable messages.

Algorithm:

Step 1: Start.

Step 2: declare n, reversed integer, remainder and original integer.

Step 3: check the condition while(n!=0) then calculate remainder=n%10,

Reversed integer =reversed integer\*10+remainder,n=n/10.

Step 4: if (original integer==reversed integer)

Print integer is a palindrome else go to step5

Step 5: print integer is not a palindrome.

Step 6: Stop.

Source code:

#include<stdio.h>

#include<conio.h>

Int main()

{

Int n,reversedInteger=0,remainder,originalInteger;

Clrscr();

Printf(“enter an integer:”);

Scanf(“%d”,&n);

originalInteger=n;

while(n!=0)

{

Remainder=n%10;

reversedInteger=reversedInteger\*10+remainder;

n=n/10;

}

If(originalInteger==reversedInteger)

Printf(“Num: %d,Reverse: %d,is a palindrome”,originalInteger,reversedInteger);

Else

Printf(“Num: %d,Reverse: %d,not a palindrome”,originalInteger,reversedInteger);

Getch();

Return 0;

}

Expected output:

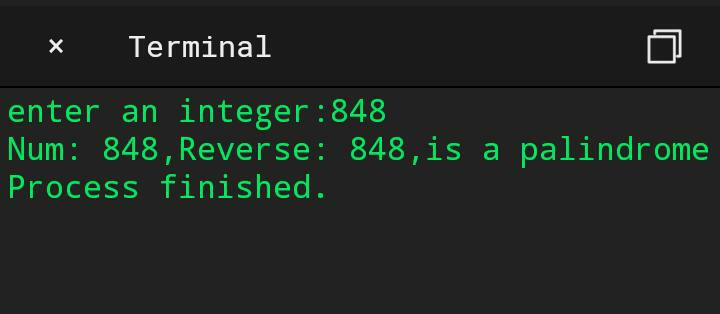
Enter an integer 12321

Num:12321

Reverse:12321

It is a palindrome.

Actual output:



RESULT: Above programme has been executed successfully

Income tax

Experiment No: 11 Roll no:1872035

Date: page:

Aim: To write a c program for the problem, Assume that the united states of America uses the following income tax code formula for their annual income: First US $5000 of income: 0% tax Next US $10,000 0f income: 10% of tax, next US $20,000 of income:15% tax. An amount of above US $35,000: 20% of tax. For example, somebody earning US $38,000 annually would owe US $5000\*0.00+10,000\*0.10+20,000\*0.15+30,000\*0.20, which comes to US $4600.Write a program that uses a loop to input the income and calculate and repeat the owed tax amount.

Algorithm:

Step 1: Start.

Step 2: declare float income tax=0.

Step 3: enter your income.

Step 4: check the condition

If(income>35000) then calculate

Income=income-35000

Tax=(10,000\*0.10)+(20,000\*0.15)+(income\*0.20);

Print your tax is: else go to step 5.

Step 5: else if(income>20000 && income<=35000)

Income=income-15000

Tax=(10,000\*0.10)+(income\*0.15) and print your tax is: else go to step 6

Step 6: else if(income>10000 && income<=20000)

Income=income-10000

Tax=(1000\*0.10)+(income\*0.15)

Print your tax is: else go to step7

Step 7: else if(income>5000 && income<=10000)

Income=income-5000

Tax=income\*0.10, print your tax is: else go to step 8

Step 8: print you have no tax

Step 9:income

Source code:

#include<stdio.h>

#include<conio.h>

Void main()

{

Float income,tax=0;

Clrscr();

Printf(“\nenter your income:US$ “);

Scanf(“%f”,&income);

If(income>35000)

{

Income=income-35000;

Tax=(10000\*0.10)+(20000\*0.15)+(income\*0.20);

Print(“\nyour tax is US$ %f”,tax);

}

Else if(income>20000 && income<=35000)

{

Income=income-15000;

Tax=(10000\*0.10)+(income\*0.15);

Printf(“\nyour tax is US$ %f”,tax);

}

Else if(income>10000 && income<=20000)

{

Income=income-15000;

Tax=(10000\*0.10)+(income\*0.15);

Printf(“\your tax is US$ %f”,tax);

}

Else if(income>5000 && income< =10000)

{

Income=income-5000;

Tax=income\*0.10;

Printf(“\n your tax is US$ %f”,tax);

}

Else

{

Printf(“\n you have no tax”);

}

Getch();

}

Expected output:

Enter your income US $23000

Your tax is US $5600

ACTUAL OUTPUT:



RESULT : ABOVE PROGRAMME HAS BEEN EXECUTED SUCCESSFULLY

Hailstone sequence

Experiment No:12 Roll no:

Date: page no:

Aim: To write a c program to generate hailstone sequence.

Algorithm:

Step 1: Start.

Step 2: declare a function to generate next number.

Step 3: declare I = num.

Step 4: if (i%2 == 0) is true then calculate i=i/2 else go to step 5

Step 5: calculate i=3\*i+1.

Step 6: Stop.

Source code:

#include<stdio.h>

#include<conio.h>

Int getnextnumber(int num)

{

Int i=num;

If(i%2==0)

{

I=i/2;

}

Else

{

I=3\*i+1;

}

Return(i);

}

Void getHailstone(int num)

{

Int a=num;

If(a==1)

{

Printf(“\t%i”,a);

}

Else

{

Printf(“\t%i”,a);

getHailstone(getnextnumber(a));

}

}

Int main()

{

Int num;

Clrscr();

Printf(“the following sequence of numbers will be generated for n= “);

Scanf(“\t%i”,&num);

getHailstone(num);

getch();

return 0;

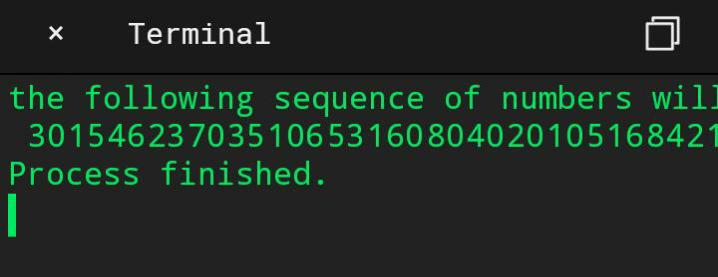
}

Expected output:

The following sequence of numbers will be generated for n=15

15 46 23 70 35 106 53 160 80 40 20 10 5 16 8 4 2 1

Actual output:



RESULT: Above programme has been executed successfully

Perfect number

Experiment No: 13 Roll no:1872035

Date: Page:

Aim: To write a c program to check whether a given number is perfect number or not.

Algorithm:

Step 1: start.

Step 2: declare number,rem,sum=0,i.

Step 3: enter a number.

Step 4: for(i=1;i<=(number-1);i++)

Rem=number%i;

If(rem==0) is true go to step 5.

Step 5: sum=sum+i.

Step 6: if(sum==number) is true go to step 7 else go to step 8.

Step 7: print entered number is a perfect number.

Step 8: print entered number is not a perfect number.

Step 9: stop.

Source code:

#include<stdio.h>

#include<conio.h>

Int main()

{

Int number,rem,sum=0,I;

Clrscr();

Printf(“enter a number\n”);

Scanf(“%d”,&number);

For(i=1;i<=(number-1);i++)

{

Rem=number%i;

If(rem==0)

{

Sum=sum+I;

}

}

If(sum==number)

{

Printf(“entered number is perfect number”);

}

Else

{

Printf(“entered number is not a perfect number”);

}

Getch();

Return 0;

}

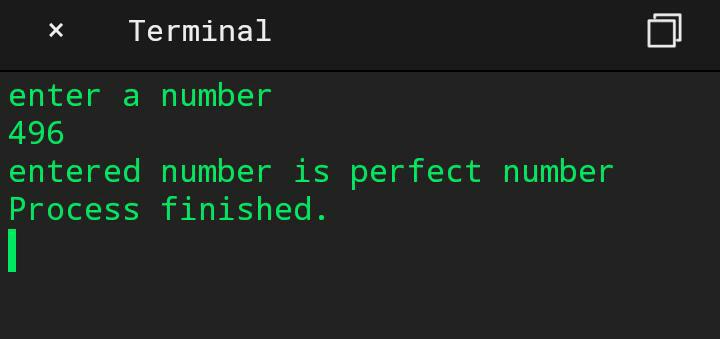
Expected output:

Enter a number

28

Entered number is a perfect number.

Actual output:



RESULT: Above programme has been executed successfully

Prime numbers in the give range

Experiment No: 13 Roll no:1872035

Date: page:

Aim: To write a c program to check whether a given number is prime number in the given range.

Algorithm:

Step1:start

Step2:Initialize I, j, n, count=0 as integers

Step3:Read n value

Step4:for(i=1;i<=n;i++)

Count=0

For(j=1;j<=n;j++)

If(i%j==0)

Count++

If(count==2)

Step 5:print result

Step 6:Stop.

Source code:

#include<stdio.h>

#include<conio.h>

Void main()

{

Int I,j,n,count=0;

Printf(“enter the range of prime numbers”);

Scanf(“%d”,&n);

Printf(“the prime numbers are:”);

For(i=1;i<=n;i++)

{

Count=0;

For(j=1;j<=n;j++)

{

If(i%j==0)

{

Count++;

}

}

If(count==2)

{

Printf(“%d”,i);

}

}

Return 0;

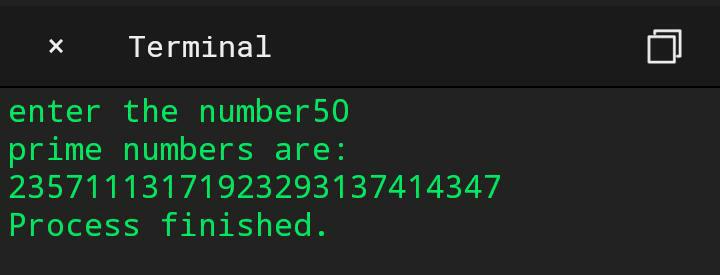
}

Expected output:

Enter the range of prime numbers 30

The prime numbers are 2 3 5 7 11 13 17 19 23 29

Actual output:



RESULT: Above programme has been executed successfully

Call By Value

Experiment No: 15 Roll no:1872035

Date: page no:

Aim: To determine the swapping of two numbers using call by value

Algorithm:

Step1: start

Step 2: initializing a and b values

Step 3:printing the a ,b values using swapping

Step 4: calling the function swap by passing the a, b values

Step 5 swapping the a, b values using temp

Step 6:printing the a, b values after swapping the values in the function

Step 7:printing the a, b values after swapping the values in the main

Step8:stop

Source code:

#include<stdio.h>

#include<conio.h?

Void swap(int , int);

Int main()

{

Int a = 50;

Int b = 60;

Printf(“Before swapping the values in main a = %d, b = %d\n”,a,b);

Printf(“After swapping values in main a = %d, b = %d\n”,a,b);

Void swap (int a, int b)

{

Int temp;

Temp = a; A=b; B=temp;

Printf(“After swapping values in function a = %d, b = %d\n”,a,b);

Getch();

}

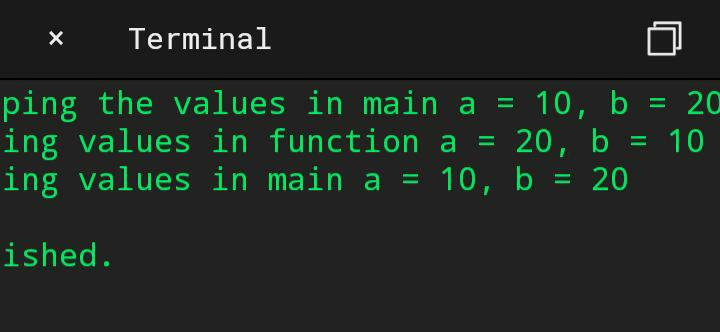
Expected Output:

Before swapping the values in main a = 50, b = 60

After swapping values in function a = 60, b = 50

After swapping values in main a = 50, b = 60

Actual Output



RESULT: Above programme has been executed successfully

Call by reference

Experiment No:16 Roll no:1872035

Date: page no:

Aim:To determine the swapping of two numbers using call by reference

Algorithm:

Step1: start

Step 2: initializing a=10 B and b=20

Step 3:printing the a,b values using swapping

Step 4: calling the function swapp by passing the a,b values tem, \*a, \*b

Step 5 swapping the a, b values using temp

Step 6:printing the a,b values after r swapping the values in the function

Step 7:printing the a,b values after swapping the values in the main

Step8:stop.

Source code:

#include<stdio.h>

#include<conio.h>

Void swap(int \*, int \*);

Int main()

{

Int a = 10;

Int b = 20;

Printf(“Before swapping the values in main a = %d, b = %d\n”,a,b);

Printf(“After swapping values in main a = %d, b = %d\n”,a,b);

Void swap (int \*a, int \*b)

{

Int temp;

Temp = \*a;

\*a=\*b;

\*b=temp;

Printf(“After swapping values in function a = %d, b = %d\n”,\*a,\*b);

Getch();

}

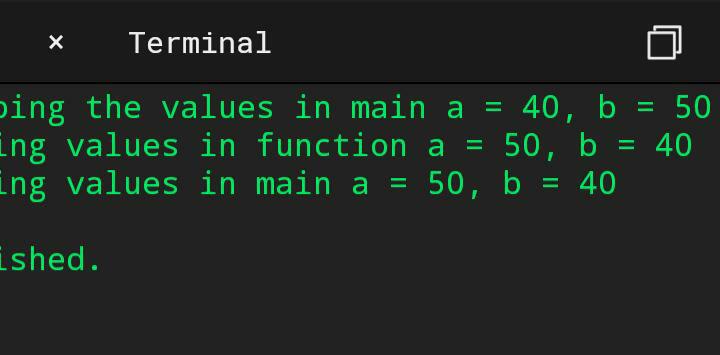
Expected output

Before swapping the values in main a = 100, b = 200

After swapping values in function a = 200, b = 100

After swapping values in main a = 200,b = 100

ACTUAL OUTPUT:



RESULT: Above programme has been executed successfully

Armstrong numbers

Experiment No: 17 Roll no:1872035

Date: page no:

Aim: To write a c program to find all amstrong numbers in range of 0 and 999.

Algorithm:

Step 1: Start.

Step 2: declare num , I , rem, cube, n, sum =0.

Step 3: enter range from o to 999

Step 4: for(i=0;i<=n;i++)

{

Num=I;

Sum=0;

While(num!=0)

{

Rem=num%10;

Cube=pow(rem,3);

Sum=sum+cube;

Num=num/10;

}

If ( num == I ) is true go to step 5

Step 5: print amstrong numbers are:

Step 6: Stop.

Source code:

#include<stdio.h>

#include<math.h>

#include<conio.h>

Void main()

{

Int n,I,rem,cube,num,sum=0;

Clrscr();

Printf(“\n enter the range from 0 to:”);

Scanf(“%d”,&n);

Printf(“\n Amstsrong numbers are:”);

For(i=0;i<=n;i++)

{

Num=I;

Sum=0;

While(num!=0)

{

Rem=num%10;

Cube=pow(rem,3);

Sum=sum+cube;

Num=num/10;

}

If(sum==i)

Printf(“\t%d”,i);

}

Getch();

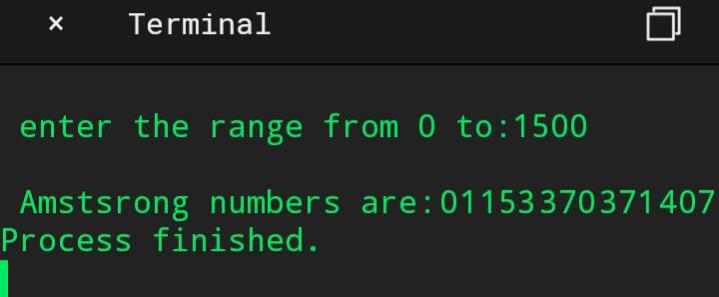
}

Expected output:

Enter the range from 0 to :999

Amstrong numbers are:0 1 153 370 371 407

Actual output:



RESULT: Above programme has been executed successfully

Matrix operations

Experiment No:18 Roll no:1872035

Date: page no:

Aim: To write a c program to perform addition, subtraction, multiplication of matrices, transpose of a matrix, to find whether a given matrix is magic square or not, inverse of a matrix and to print upper and lower triangle of matrix using switch case.

Algorithm:

Step 1: Start.

Step 2: declare c,func1(),func2(),func3(),func4(),func5(),func6() and func7().

Step 3: use switch case to perform seven operations.

Step 4: to perform addition operation enter your choice as 1.

Step 5: to perform subtraction operation enter your choice as 2.

Step 6: to perform multiplication operation enter your choice as 3.

Step 7: to perform transpose of a matrix enter your choice as 4.

Step 8: find whether a given matrix is magic square or not, enter your choice as 5.

Step 9: to perform inverse of a matrix, enter your choice as 6.

Step 10: to print upper and lower triangle of matrix, enter your choice as 7.

Step 11: Stop.

Source code:

#include<stdio.h>

#include<conio.h>

Void main()

{

Int c;

Void func1();

Void func2();

Void func3();

Void func4();

Void func5();

Void func6();

Void func7();

Clrscr();

Printf(“\n-:Matrix Manipulation functions :”);

Printf(“\n--------------------------“);

Printf(“\n Matrix addition:1”);

Printf(“\n matrix subtraction:2”);

Printf(“\n matrix multiplication:3”);

Printf(“\n transpose of a matrix matrix :4”);

Printf(“\n matrix is magic square or not:5”);

Printf(“\n inverse of a matrix:6”);

Printf(“\n print the upper and lower triangle of matrix:7”);

Printf(“\n enter your choice :”);

Scanf(“%d”,&c);

Switch©

{

Case 1:func1();

Break;

Case 2:func2();

Break;

Case 3:func3();

Break;

Case 4:func4();

Break;

Case 5:func5();

Break;

Case 6:func6();

Break;

Case 7:func7();

Break;

Default:

Printf(“\n invalid choice”);

}

Getch();

}

Void func1()

{

Int I, j, a[10][10], b[10][10], c[10][10];

Int row1, col1, row2, col2;

Printf(“\nEnter the number of Rows of a: “);

Scanf(“%d”, &row1);

Printf(“\nEnter the number of Cols of a : “);

Scanf(“%d”, &col1);

Printf(“\nEnter the number of Rows of b : “);

Scanf(“%d”, &row2);

Printf(“\nEnter the number of Columns of b : “);

Scanf(“%d”, &col2);

If (row1 != row2 || col1 != col2) {

Printf(“\nOrder of two matrices is not same “)

}

Else

{

Printf(“\n enter first matrix:\n”);

For (I = 0; I < row1; i++) {

For (j = 0; j < col1; j++) {

Scanf(“%d”, &a[i][j]);

}

}

Printf(“\n enter second matrix:\n”);

For (I = 0; I < row2; i++)

For (j = 0; j < col2; j++) {

Scanf(“%d”, &b[i][j]);

}

For (I = 0; I < row1; i++)

For (j = 0; j < col1; j++) {

C[i][j] = a[i][j] + b[i][j];

}

Printf(“\nThe Addition of two Matrices is : \n”);

For (I = 0; I < row1; i++) {

For (j = 0; j < col1; j++) {

Printf(“%d\t”, c[i][j]);

}

Printf(“\n”);

}

}

}

Void func2()

{

Int I, j, a[10][10], b[10][10], c[10][10];

Int row1, col1, row2, col2;

Printf(“\nEnter the number of Rows of a: “);

Scanf(“%d”, &row1);

Printf(“\nEnter the number of Cols of a : “);

Scanf(“%d”, &col1);

Printf(“\nEnter the number of Rows of b : “);

Scanf(“%d”, &row2);

Printf(“\nEnter the number of Columns of b : “);

Scanf(“%d”, &col2);

If (row1 != row2 || col1 != col2) {

Printf(“\nOrder of two matrices is not same “);

}

Else

{

For (I = 0; I < row1; i++) {

For (j = 0; j < col1; j++) {

Printf(“Enter the Element a[%d][%d] : “, I, j);

Scanf(“%d”, &a[i][j]);

}

}

For (I = 0; I < row2; i++)

For (j = 0; j < col2; j++) {

Printf(“Enter the Element b[%d][%d] : “, I, j);

Scanf(“%d”, &b[i][j]);

}

For (I = 0; I < row1; i++)

For (j = 0; j < col1; j++) {

C[i][j] = a[i][j] – b[i][j];

}

Printf(“\nThe substraction of two Matrices is : \n”);

For (I = 0; I < row1; i++) {

For (j = 0; j < col1; j++) {

Printf(“%d\t”, c[i][j]);

}

Printf(“\n”);

}

}

}

Void func3()

{

Int m, n, p, q, c, d, k, sum = 0;

Int first[10][10], second[10][10], multiply[10][10];

Printf(“Enter number of rows and columns of first matrix\n”);

Scanf(“%d%d”, &m, &n);

Printf(“Enter elements of first matrix\n”);

For (c = 0; c < m; c++)

For (d = 0; d < n; d++)

Scanf(“%d”, &first[c][d]);

Printf(“Enter number of rows and columns of second matrix\n”);

Scanf(“%d%d”, &p, &q);

If (n != p)

Printf(“The matrices can’t be multiplied with each other.\n”);

Else

{

Printf(“Enter elements of second matrix\n”);

For (c = 0; c < p; c++)

For (d = 0; d < q; d++)

Scanf(“%d”, &second[c][d]);

For (c = 0; c < m; c++) {

For (d = 0; d < q; d++) {

For (k = 0; k < p; k++) {

Sum = sum + first[c][k]\*second[k][d];

}

Multiply[c][d] = sum;

Sum = 0;

}

}

Printf(“Product of the matrices:\n”);

For (c = 0; c < m; c++) {

For (d = 0; d < q; d++)

Printf(“%d\t”, multiply[c][d]);

Printf(“\n”);

}

}

}

Void func4()

{

Int m, n, c, d, matrix[10][10], transpose[10][10];

Printf(“Enter the number of rows and columns of matrix\n”);

Scanf(“%d%d”, &m, &n);

Print(“Enter elements of the matrix\n”);

For (c = 0; c < m; c++)

For(d = 0; d < n; d++)

Scanf(“%d”, &matrix[c][d]);

For (c = 0; c < m; c++)

For( d = 0 ; d < n ; d++ )

Transpose[d][c] = matrix[c][d];

Printf(“Transpose of the matrix:\n”);

For (c = 0; c < n; c++) {

For (d = 0; d < m; d++)

Printf(“%d\t”, transpose[c][d]);

Printf(“\n”);

}

}

Void func5()

{

Int size = 3;

Int matrix[3][3]; // = {{4,9,2},{3,5,7},{8,1,6}};

Int row, column = 0;

Int sum, sum1, sum2;

Int flag = 0;

Printf(“\nEnter matrix : “);

For (row = 0; row < size; row++) {

For (column = 0; column < size; column++)

Scanf(“%d”, &matrix[row][column]);

}

Printf(“Entered matrix is : \n”);

For (row = 0; row < size; row++) {

Printf(“\n”);

For (column = 0; column < size; column++) {

Printf(“\t%d”, matrix[row][column]);

}

}

//For diagonal elements

Sum = 0;

For (row = 0; row < size; row++) {

For (column = 0; column < size; column++) {

If (row == column)

Sum = sum + matrix[row][column];

}

}

//For Rows

For (row = 0; row < size; row++) {

Sum1 = 0;

For (column = 0; column < size; column++) {

Sum1 = sum1 + matrix[row][column];

}

If (sum == sum1)

Flag = 1;

Else

{

Flag = 0;

Break;

}

}

//For Columns

For (row = 0; row < size; row++) {

Sum2 = 0;

For (column = 0; column < size; column++) {

Sum2 = sum2 + matrix[column][row];

}

If (sum == sum2)

Flag = 1;

Else {

Flag = 0;

Break;

}

}

If (flag == 1)

Printf(“\nMagic square”);

Else

Printf(“\nNo Magic square”);

}

Void func6()

{

Int mat[3][3], I, j;

Float determinant = 0;

Printf(“Enter elements of matrix row wise:\n”);

For(I = 0; I < 3; i++)

For(j = 0; j < 3; j++)

Scanf(“%d”, &madeterminant

Printf(“\nGiven matrix is:”);

For(I = 0; I < 3; i++){

Printf(“\n”);

For(j = 0; j < 3; j++)

Printf(“%d\t”, mat[i][j]);

}

//finding determinant

For(I = 0; I < 3; i++)

Determinant = determinant + (mat[0][i] \* (mat[1][(i+1)%3] \* mat[2][(i+2)%3] – mat[1][(i+2)%3] \* mat[2][(i+1)%3]));

Printf(“\n\ndeterminant: %f\n”, determinant);

Printf(“\nInverse of matrix is: \n”);

For(I = 0; I < 3; i++){

For(j = 0; j < 3; j++)

Printf(“%.2f\t”,((mat[(j+1)%3][(i+1)%3] \* mat[(j+2)%3][(i+2)%3]) – (mat[(j+1)%3][(i+2)%3] \* mat[(j+2)%3][(i+1)%3]))/ determinant);

Printf(“\n”);

}

}

Void func7()

{

Int a[10][10],I,j,row\_size,col\_size;

Printf(“\nEnter the row and column size of matrix”);

Scanf(“%d %d”,&row\_size,&col\_size);

Printf(“\nEnter the matrix elements->\n”);

For(i=0;i<row\_size;i++)

For(j=0;j<col\_size;j++)

Scanf(“%d”,&a[i][j]);

Printf(“\nThe entered matrix is\n”);

For(i=0;i<row\_size;i++)

Printf(“\n”);

For(j=0;j<col\_size;j++)

Printf(“%d\t”,a[i][j]);

}

Printf(“\nThe lower diagonal elements of the matrix is\n”);

For(i=0;i<row\_size;i++){

Printf(“\n”);

For(j=0;j<col\_size;j++){

If(i>j)

Printf(“%d\t”,a[i][j]);

}

}

Printf(“\nThe upper diagonal elements of the matrix is\n”);

For(i=0;i<row\_size;i++){

Printf(“\n”);

For(j=0;j<col\_size;j++){

If(i<j)

Printf(“%d\t”,a[i][j]);

Else

Printf(“ \t”);

}

}

}

Expected output:

-:Matrix Manipulation functions :

--------------------------

Matrix addition:1

Matrix subtraction:2

Matrix multiplication:3

Transpose of a matrix matrix :4

Matrix is magic square or not:5

Inverse of a matrix:6

Print the upper and lower triangle of matrix:7

Enter your choice :1

Enter the number of Rows of a: 2

Enter the number of Cols of a : 2

Enter the number of Rows of b : 2

Enter the number of Columns of b : 2

Enter first matrix:

1 2 3 4

Enter second matrix:

1 2 3 4

The Addition of two Matrices is :

2 4

6 8

--------------------------

Matrix addition:1

Matrix subtraction:2

Matrix multiplication:3

Transpose of a matrix matrix :4

Matrix is magic square or not:5

Inverse of a matrix:6

Print the upper and lower triangle of matrix:7

Enter your choice :2

Enter the number of Rows of a: 1

Enter the number of Cols of a : 2

Enter the number of Rows of b : 1

Enter the number of Columns of b : 2

Enter the Element a[0][0] : 6

Enter the Element a[0][1] : 4

Enter the Element b[0][0] : 3

Enter the Element b[0][1] : 7

The substraction of two Matrices is :

3 -3

-:Matrix Manipulation functions :

--------------------------

Matrix addition:1

Matrix subtraction:2

Matrix multiplication:3

Transpose of a matrix matrix :4

Matrix is magic square or not:5

Inverse of a matrix:6

Print the upper and lower triangle of matrix:7

Enter your choice :3

Enter number of rows and columns of first matrix

2 2

Enter elements of first matrix

1 2 3 4

Enter number of rows and columns of second matrix

2 2

Enter elements of second matrix

3 4 5 6

Product of the matrices:

13 16

29 36

-:Matrix Manipulation functions :

--------------------------

Matrix addition:1

Matrix subtraction:2

Matrix multiplication:3

Transpose of a matrix matrix :4

Matrix is magic square or not:5

Inverse of a matrix:6

Print the upper and lower triangle of matrix:7

Enter your choice :4

Enter the number of rows and columns of matrix

4 2

Enter elements of the matrix

8 5 7 4

9 3 6 2

Transpose of the matrix:

8 7 9 6

5 4 3 2

-:Matrix Manipulation functions :

--------------------------

Matrix addition:1

Matrix subtraction:2

Matrix multiplication:3

Transpose of a matrix matrix :4

Matrix is magic square or not:5

Inverse of a matrix:6

Print the upper and lower triangle of matrix:7

Enter your choice :5

Enter matrix : 2 7 6

9 5 1

4 3 8

Entered matrix is :

2 7 6

9 5 1

4 3 8

Magic square

Matrix addition:1

Matrix subtraction:2

Matrix multiplication:3

Transpose of a matrix matrix :4

Matrix is magic square or not:5

Inverse of a matrix:6

Print the upper and lower triangle of matrix:7

Enter your choice :6

Enter elements of matrix row wise:

6 7 8 9

1 3 4 5

1 3 4 6

Given matrix is:

6 7 8

9 1 3

4 5 1

Determinant: 265.000000

Inverse of matrix is:

-0.05 0.12 0.05

0.01 -0.10 0.20

0.15 -0.01 -0.22

Matrix multiplication:3

Transpose of a matrix matrix :4

Matrix is magic square or not:5

Inverse of a matrix:6

Print the upper and lower triangle of matrix:7

Enter your choice :7

Enter the row and column size of matrix2 2

Enter the matrix elements->

3 4

2 3

The entered matrix is

3 4

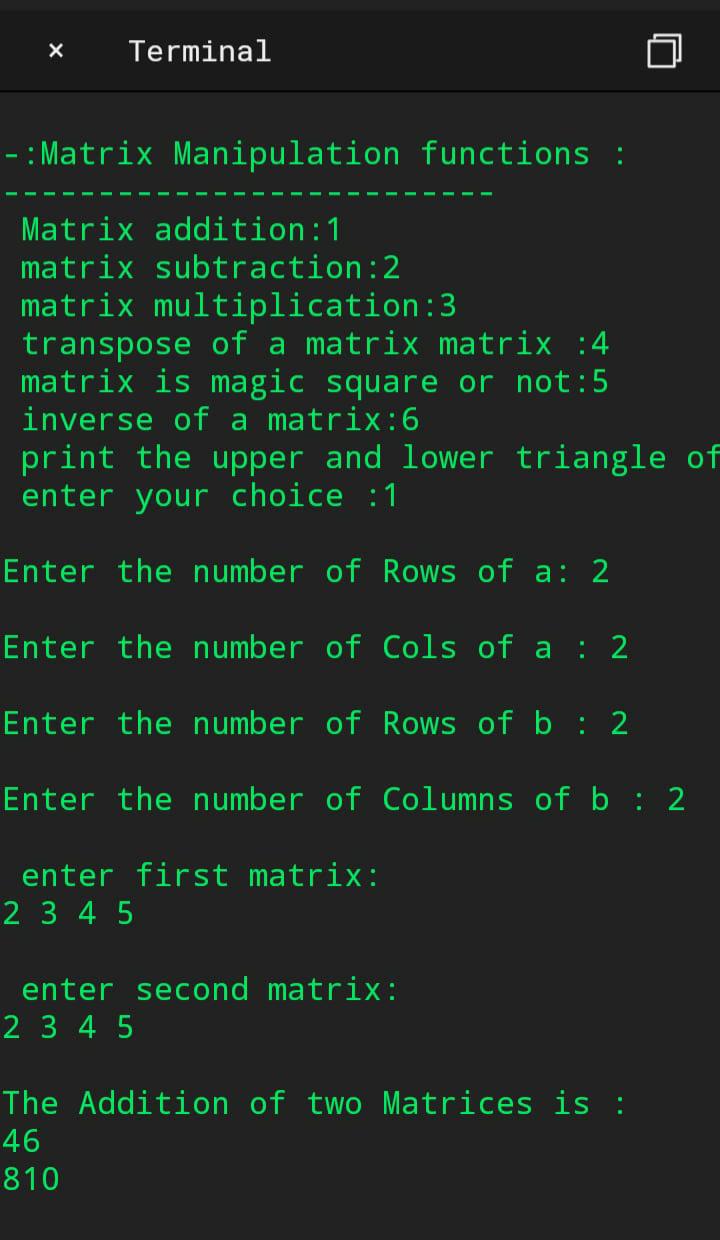
2 3

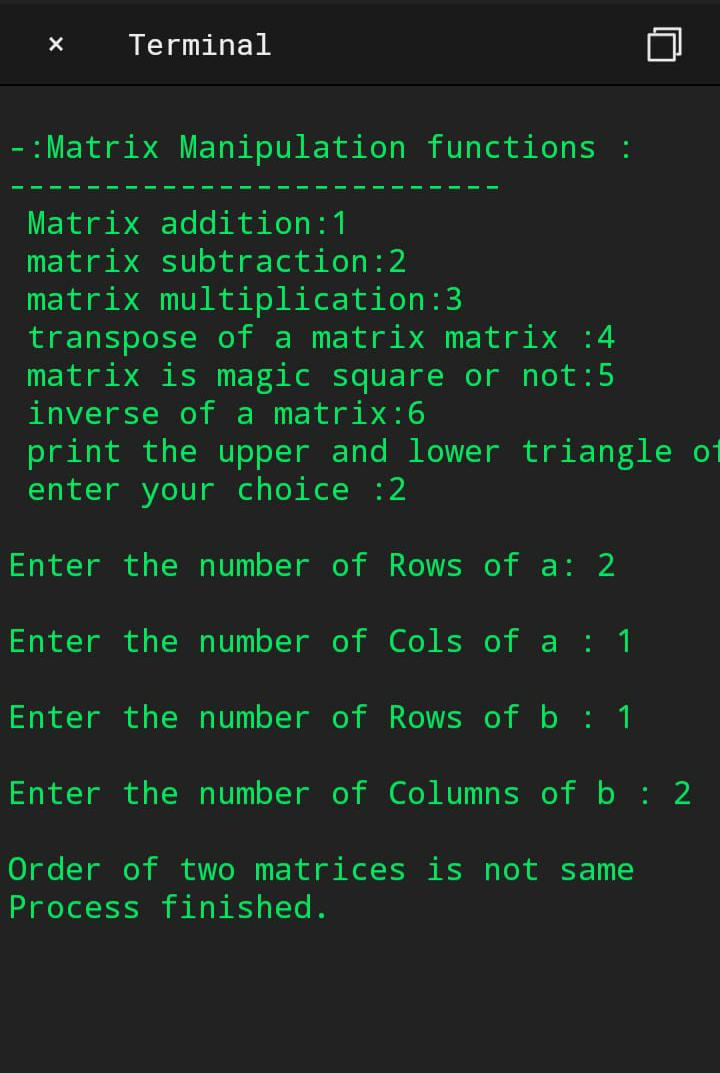
The lower diagonal elements of the matrix is

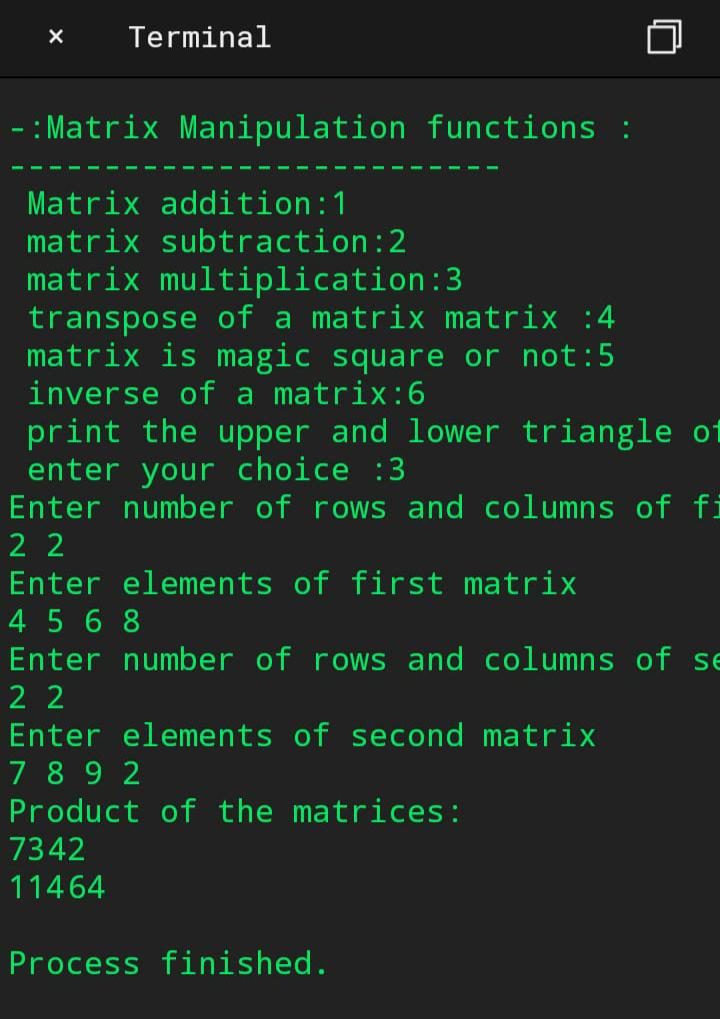
2

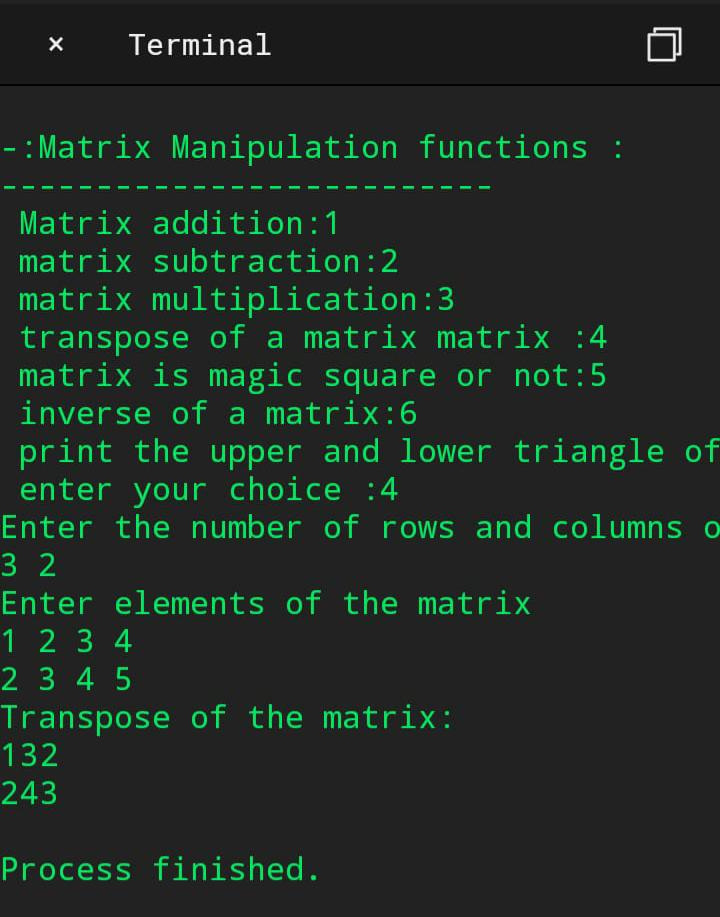
The upper diagonal elements of the matrix is 4

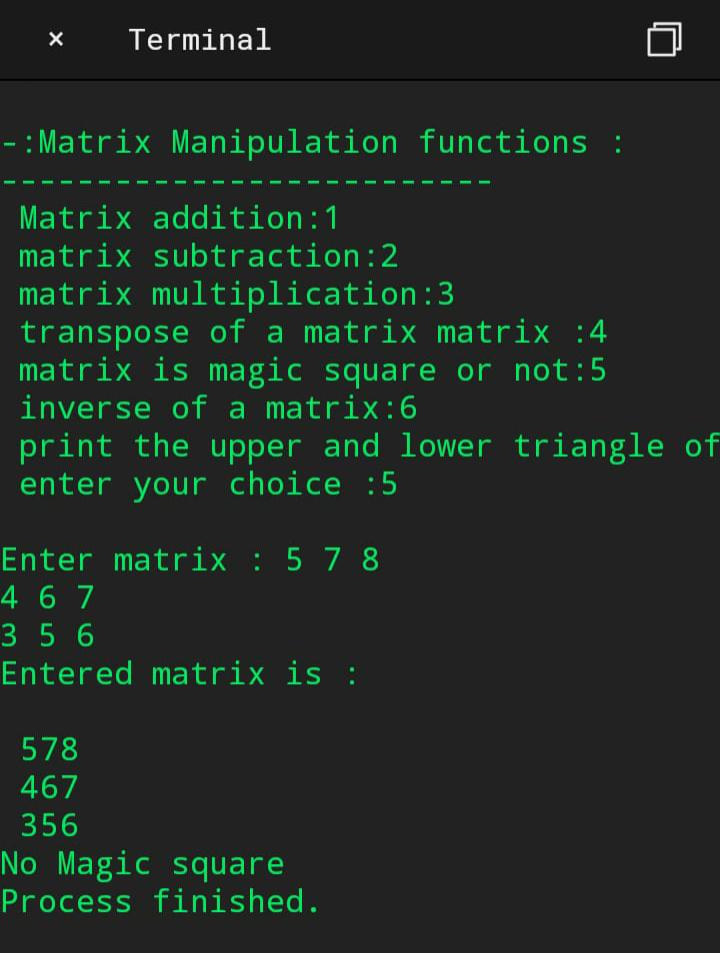
Actual output:

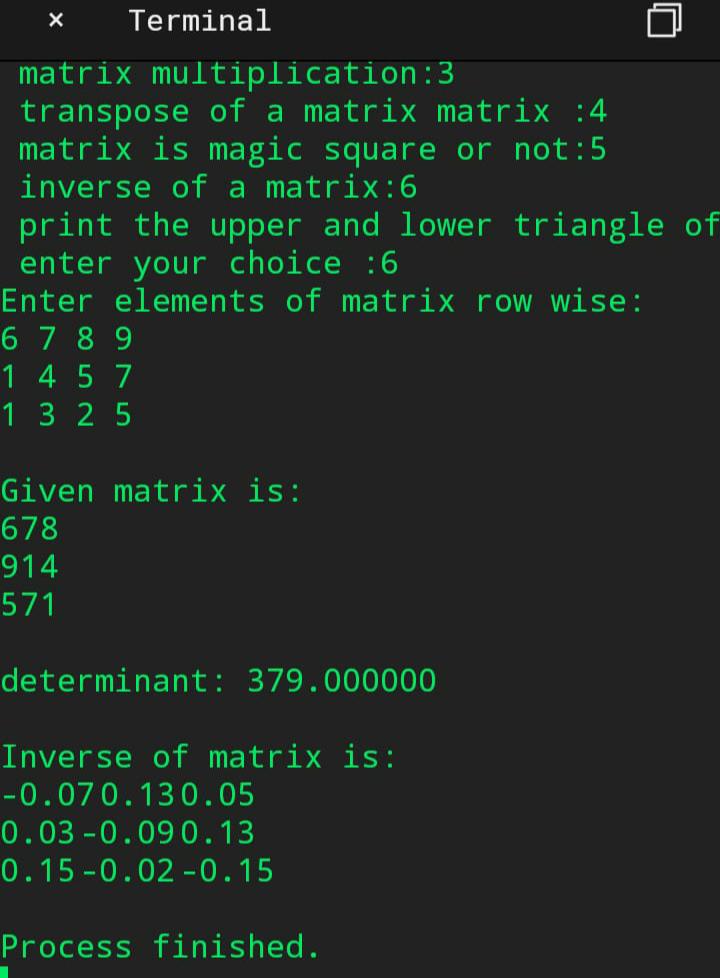
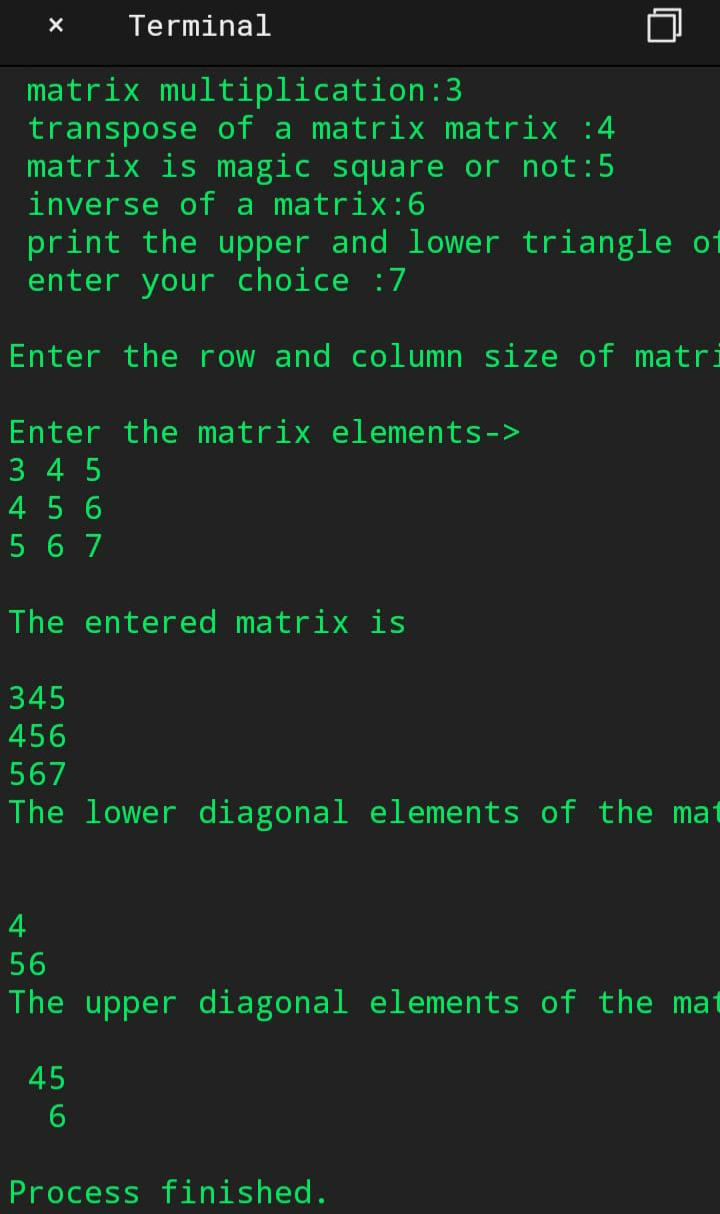












RESULT: Above programme has been executed successfully

Sorting of strings

Experiment No: 19 Roll no:1872035

Date: page:

Aim: To sort a set of strings in ascending alphabetical order

Algorithm:

Step1: start

Step 2:initialize I, j, count as integers, get strongIi]

Step3:Read count

Step4:for(i=0;i<=count;i++), get str[i],

For(i=0;i<=count;i++),

For(j=i+1;j<=count;j++),

If(strcmp(str[i],str[j])>0)

Strcpy(temp,str[i]);

Strcpy(str[i],str[j]);

Strcpy(str[j],temp);

Step4:Stop

Source Code:

#include <stdio.h>

#include <stdlib.h>

Int main()

{

Int I,j,count;

Char str[25][25],temp[25];

Puts(“how many strings you are going to enter?”);

Scanf(“%d”,&count);

Puts(“enter strings one by one: “);

For(i=0;i<=count;i++)

Gets(str[i]);

For(i=0;i<=count;i++)

For(j=i+1;j<=count;j++)

{

If(strcmp(str[i],str[j])>0)

{

Strcpy(temp,str[i]);

Strcpy(str[i],str[j]);

Strcpy(str[j],temp);

}

}

Printf(“order of sorted strings”);

For(i=0;i<=count;i++)

Puts(str[i]);

Return 0;

}

Expected output:

How many strings you are going to enter: 4

Enter strings one by one:

Prathyusha

Jhansi

Manasa

Sruthi

Order of sorted strings:

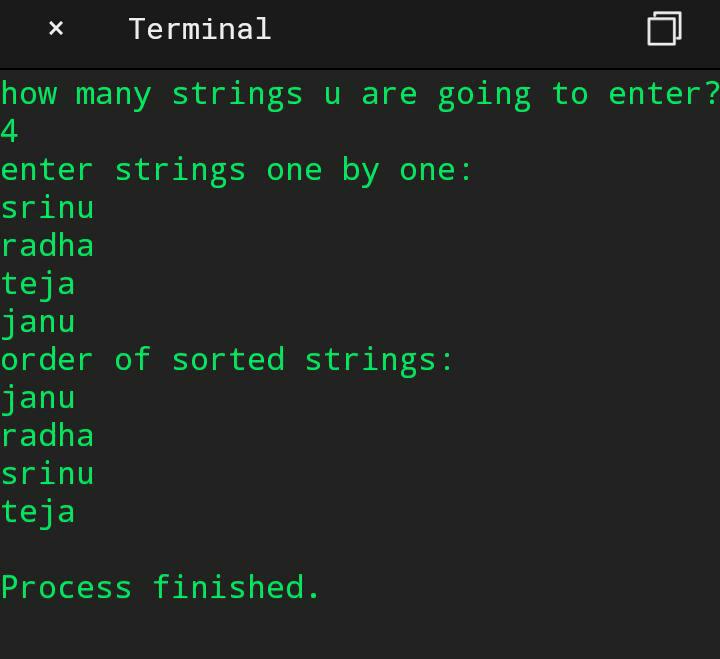
Jhansi

Manasa

Prathyusha

Sruthi

Actual output:



Result: Hence the program has been executed successfully.

Bank account details of a customer

Experiment No:20 Roll no:1872035

Date: page no:

Aim: To write a C++ program to define a class represent a bank account the following Members as data members: a)name of the depositor b)account number c)withdrawal Amount d)balance amount in the account. Member functions: a)to assign initial values b)to deposit an amount c)to withdraw an amount after checking balance d)to display name and balance.

Algorithm:

Step 1: start

Step 2: declare name, account number, account type, balance in class bank.

Step 3: use void assign() to declare all the details like name, account number, initial balance.

Step 4: use void deposit() to deposit amount and bal=bal+ deposit and print the total balance.

Step 5: use void withdraw() to withdraw amount. Enter how much amount to be withdraw

If(amount>bal) then print there is no sufficient amount else bal=bal-with amount

Display the final balance.

Step 6: use void display() to print all the details.

Step 7: In main() we have to create a object for class bank.

Step 8: Create a choice list. Using switch case call the member functions of the class.

Step 9: Stop.

Source code:

#include<iostream.h>

#include<conio.h>

Class bank

{

Char name[20],actype[20];

Int acno,bal;

Public:

Void initialbal();

Void deposit();

Void withdraw();

Void display();

};

Void bank::initialbal()

{

Cout<<”Enter name:”;

Cin>>name;

Cout<<”Enter A/C no:”;

Cin>>acno;

Cout<<”Enter A/C type:”;

Cin>>actype;

Cout<<”Enter opening balance:”;

Cin>>bal;

}

Void bank::deposit()

{

Int d=0;

Cout<<”Enter deposit amount:”;

Cin>>d;

Bal+=d;

Cout<<”Deposit balance=”<<bal;

}

Void bank::withdraw()

{

Int w;

Cout<<”Balance amount=”<<bal;

Cout<<”\nEnter withdraw amount:”;

Cin>>w;

If(w<=bal)

{

Bal=bal-w;

Cout<<”After withdrawal…Balance=”<<bal;

}

Else

Cout<<”Insufficient balance!”;

}

Void bank::display()

{

Cout<<”\nDETAILS”;

Cout<<”\nName:”<<name;

Cout<<”\nA/C no:”<<acno;

Cout<<”\nA/C type:”<<actype;

Cout<<”\nBalance:”<<bal;

}

Void main()

{

Int ch;

Bank obj;

Clrscr();

Cout<<”\nOpening the account\n”;

Obj.initialbal();

Do

{

Cout<<”\n\nCHOICE LIST”;

Cout<<”\n1.To deposit”;

Cout<<”\n2.To withdraw”;

Cout<<”\n3.To display all the details”;

Cout<<”\n4.EXIT”;

Cout<<”\nEnter your choice:”;

Cin>>ch;

Switch(ch)

{

Case 1:obj.deposit();

Break;

Case 2:obj.withdraw();

Break;

Case 3:obj.display();

Break;

Case 4:goto end;

Break;

Default:cout<<”\nInvalid option”;

}

}while(1);

End:

Getch();

}

Expected output:

Opening the account

Enter name:Ram

Enter A/C no:1203

Enter A/C type:Savings

Enter opening balance:1000

CHOICE LIST

1.To deposit

2.To withdraw

3.To display all the details

4.EXIT

Enter your choice:1

Enter deposit amount:500

Deposit balance:1500

CHOICE LIST

1.To deposit

2.To withdraw

3.To display all the details

4.EXIT

Enter your choice:2

Balace amount:1500

Enter withdraw amount:300

After withdrawal…Balance =1200

CHOICE LIST

1.To deposit

2.To wwithdra

3.To display all the details

4.EXIT

Enter your choice:3

DETAILS

Name:Ram

A/C no:1203

A/C type:Savings

Balance:1200

CHOICE LIST

1.To deposit

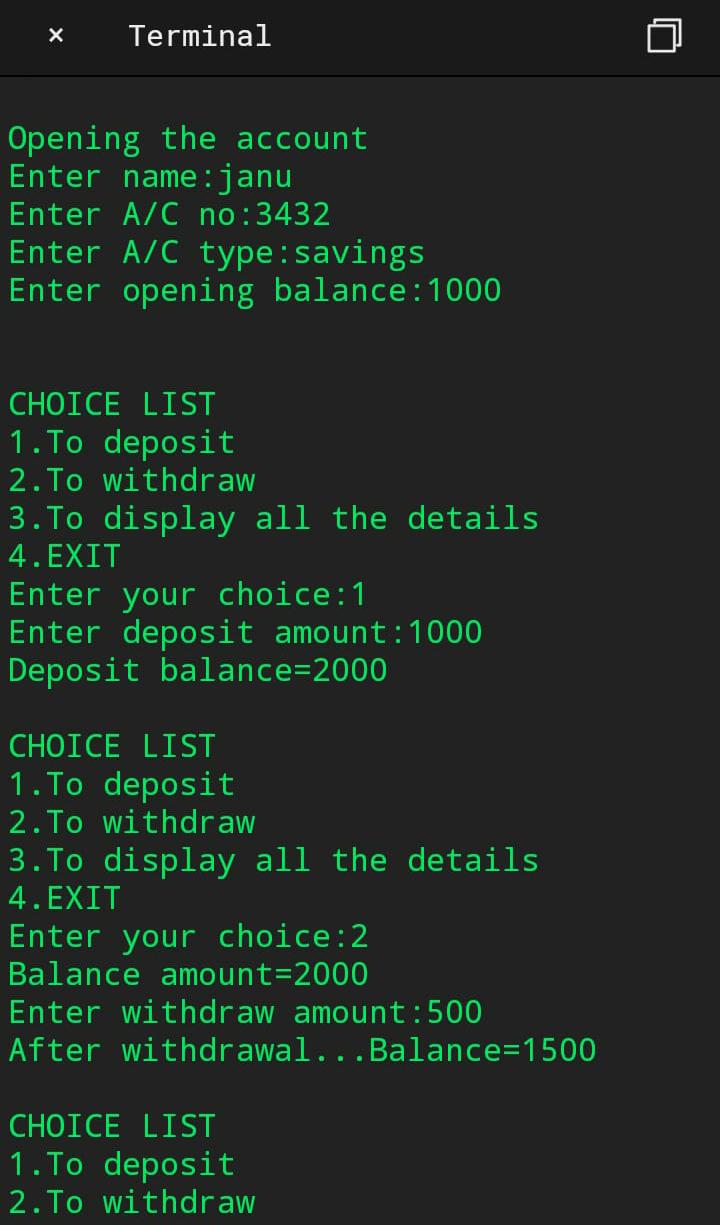
2.To withdraw

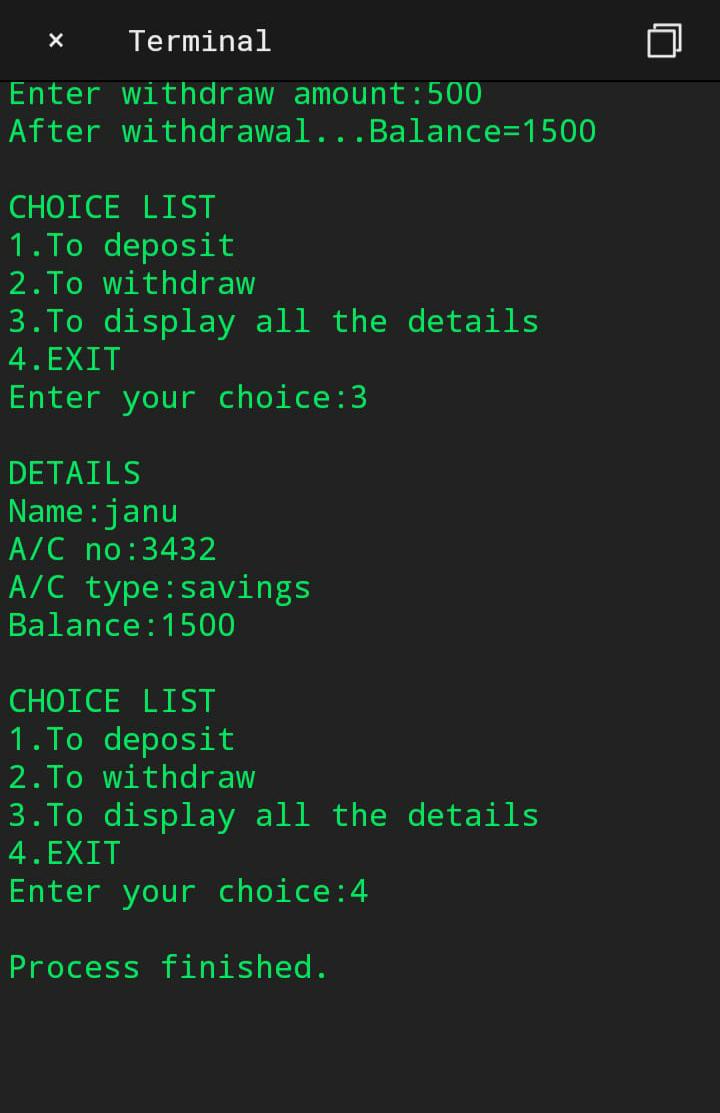
3.To display all the details

4.EXIT

Enter your choice:4

Actual output:





RESULT: Above programme has been executed successfully

Bank account details of N customers

Experiment No:21 Roll no:1872035

Date: page:

Aim: To write a C++ program for handling n number of account holders using array of objects.

Algorithm:

Step 1:start

Step 2:Create a class employee

Step 3:declare id, name,age, salary.

Step 4:use void getdata() function to read employee id, name, age, salary

Step 5:use void putdata() function to print employee id, name, age, salary.

Step 6:In main() create array of objects E[30] for the class employee.

Step 7:Call getdata() and putdata() using for loops.

Step 8:stop.

Source code:

#include<iostream.h>

#include<conio.h>

Class employee

{

Int id,age;

Char name[25];

Long salary;

Public:

Void getdata()

{

Cout<<”\nEnter employee id:”;

Cin>>id;

Cout<<”Enter employee name:”;

Cin>>name;

Cout<<”Enter employee age:”;

Cin>>age;

Cout<<”Enter employee salary:”;

Cin>>salary;

}

Void putdata()

{

Cout<<”\nId:”<<id;

Cout<<”\nName:”<<name;

Cout<<”\nAge:”<<age;

Cout<<”\nSalary:”<<salary;

}

};

Void main()

{

Int I,n;

Employee E[30];

Clrscr();

Cout<<”\nEnter no.of employees:”;

Cin>>n;

For(i=0;i<n;i++)

{

Cout<<”\nEnter details of employee “<<i+1;

E[i].getdata();

}

For(i=0;i<n;i++)

{

Cout<<”\nDetails of employee “<<i+1;

E[i].putdata();

}

Getch();

}

Expected output:

Enter no of employees:3

Enter details of employee 1

Enter employee id:872061

Enter employee name:Kavya

Enter employee age:45

Enter employee salary:45000

Enter details of employee 2

Enter employee id:1972089

Enter employee name: sravya

Enter employee age:43

Enter employee salary:salar

Enter details of employee 3

Enter employee id:1872323

Enter employee name: Bhavya

Enter employee age:42

Enter employee salary:35000

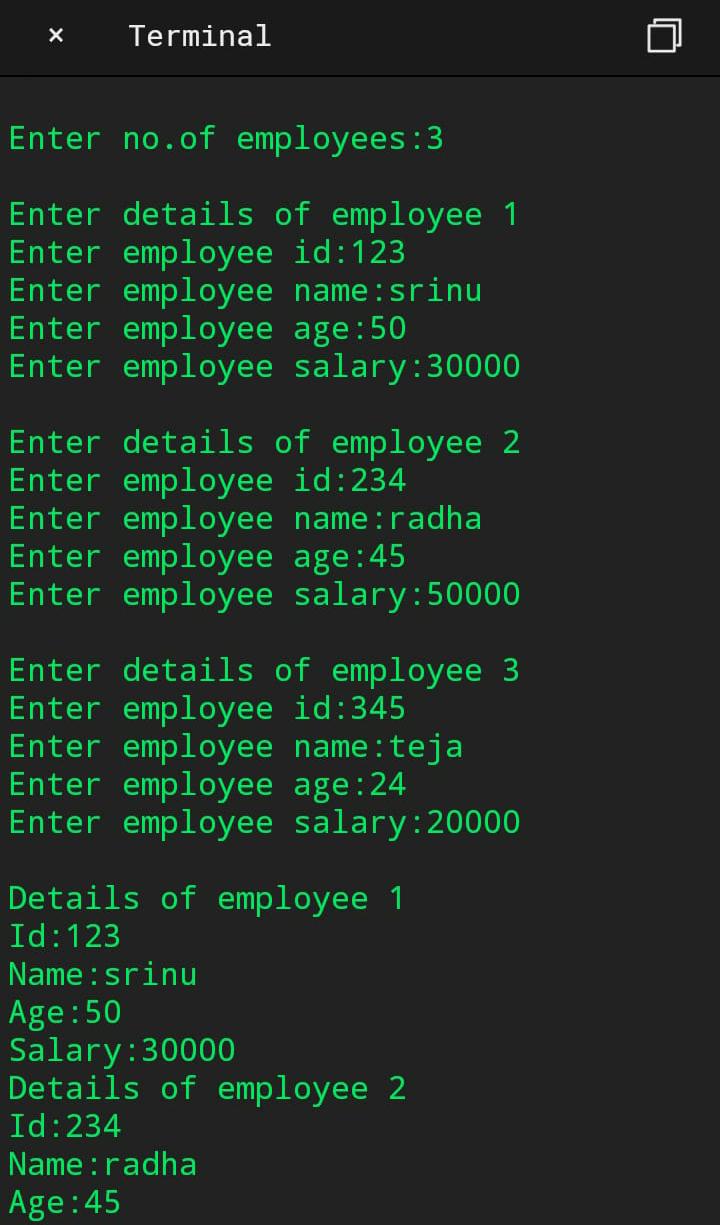
Enter details of employee 4

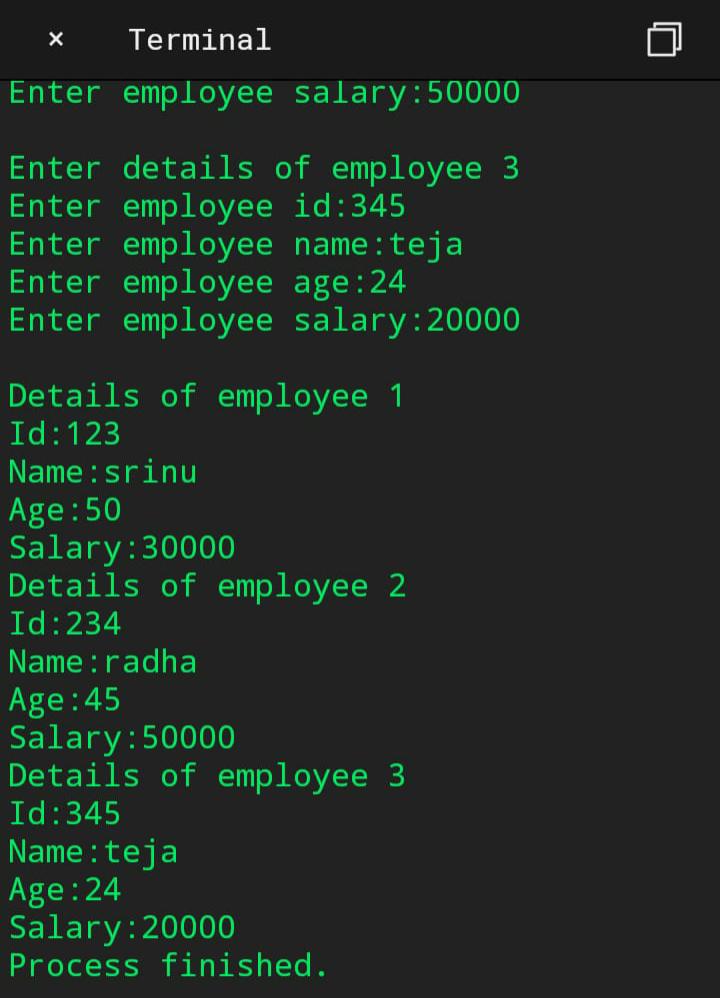
Enter employee id:1872112

Enter employee name: Navya

Enter employee age:40

Enter employee salary:3000

Actual output:



RESULT: ABOVE PROGRAMME HAS BEEN EXECUTED SUCCESSFULLY

To compute area for types of triangle

Experiment No:22 Roll no:1872035

Date: page:

Aim: To write a C++ program to compute area of right angle triangle, equilateral triangle,

Isosceles triangle using function overloading concept.

Algorithm:

Step 1:start

Step 2:declare int a, b, h, m, n and float area,area1,area2 inside a class.

Step 3:use void tri(int a) function for calculating area of equilateral triangle using

Area=((sqrt(3))/4)\*(a\*a).

Step 4:use void tri(int b, int h) function to declare two sides of right angle triangle and

Area1=(0.5)\*(float)(b\*h).

Step 5:use void tri(float l, float m, float n) to declare sides of isosceles triangle and

Area2=(l)\*(m\*n) and print area of isosceles triangle.

Step 6:in main() to declare int a, b, h, n, m, l and create objects for above classes.

Step 7:stop.

Source code:

#include<iostream.h>

#include<conio.h>

#include<math.h>

Class over

{

Public:

Int a, b, h, m, n;

Float area, area1, area2;

Void tri(int a)

{

Cout<<”enter the side of equilateral triangle\nOnly one side:”;

Cin>>a;

Area=((sqrt(3))/4)\*(a\*a);

Cout<<”area of equilateral triangle is”<<area;

}

Void tri(int b, int h)

{

Cout<<”\nenter two sides of right angle triangle:”;

Cin>>b>>h;

Area1=(0.5)\*(float)(b\*h);

Cout<<”\narea of right angled triangle is:”<<area1;

}

Void tri(float l, float m, float n)

{

Cout<<”\nenter the sides of isosceles triangle:”;

Cin>>m>>n;

L=0.5;

Area2=(l)\*(m\*n);

Cout<<”\narea of isoceles triangle=”<<area2;

}

};

Int main()

{

Int a, b, h, m, n, l;

Clrscr();

Over abc;

Abc.tri(a);

Abc.tri(b, h);

Abc.tri(l, m, n);

Getch();

Return 0;

}

Expected output:

Enter the sideof equilateral triangle: 15

Area of equilateral triangle is12.9903811

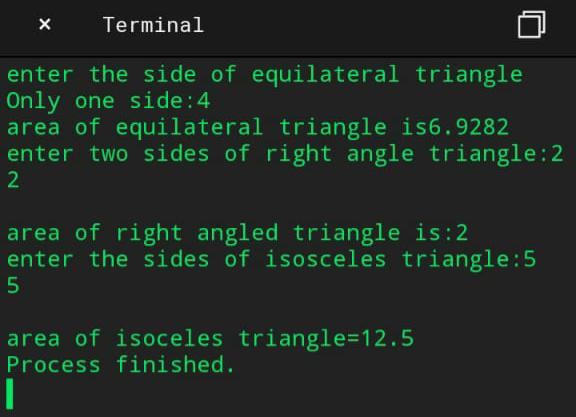
Enter two sides of right angle triangle:12 15

Area of right angled triangle is:90

Enter the sides of isosceles triangle:10 2

Area of isosceles triangle=10

Actual outpu:



RESULT: Above programme has been executed successfully

To swap two integers using friend functions

Experiment No: 23 Roll no:1872035

Date: Page:

Aim: To write a C++ program to swap the values of two integer members of different classes using friend functions.

Algorithm:

Step 1:start.

Step 2: using class A we have to declare a number and write a friend function.

Step 3: using class B we have to declare a second number and write a friend function.

Step 4:write a swap(),inside this function we have to print before swapping and f=obj1.s

S=obj2.m,m=f and print the values after swapping

Step 5:inside main() create objects.

Step 6:stop.

Source code:

#include<iostream.h>

#include<constream.h>

Class B;

Class A

{

Int s;

Public:

A()

{

Cout<<”enter a number”;

Cin>>s;

}

Friend void swap(A,B);

};

Class B

{

Int m;

Public:

B()

{

Cout<<”enter a number”;

Cin>>m;

}

Friend void swap(A,B);

};

Void swap(A obj1,B obj2)

{

Intm,s,f;

Cout<<”before swapping:”;

Cout<<”\ns=”<<obj1.s;

Cout<<”\nm=”<<obj2.m;

F=obj1.s;

S=obj2.m;

M=f;

Cout<<”\nafter swapping values are:”;

Cout<<”\ns=”<<s;

Cout<<”\nm=”<<m;

}

Void main()

{

Clrscr();

A obj1;

B obj2;

Swap(obj1,obj2);

Getch();

}

Expected output:

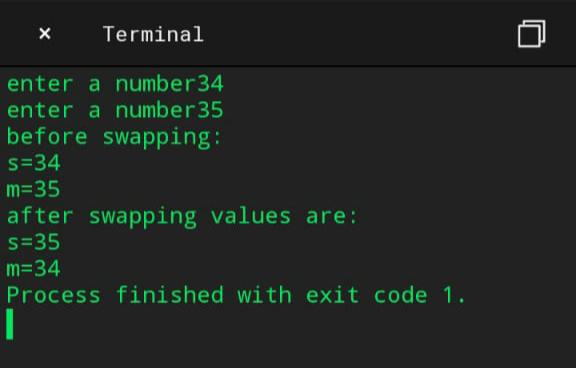
Enter a number100

Enter a number200

Before swapping: s=100 m=200

After swapping values are: s=200 m=100

Actual output:



RESULT: Above programme has been executed successfully

To compare and concatenate two strings

Experiment No:24 Roll no:1872035

Date: Page:

Aim: To write a C++ program to define a class string and overload to compute two strings and + Operator for concatenation of two strings.

Algorithm:

Step 1:start

Step 2:declare char m[20],n[20],res[20] inside a class

Step 3:print enter two strings and int result=strcmp(m,n)

Step 4:check condition if(result==0) is true the print strings are equal else print strings strings

Are not equal.

Step 5:use void operate() to print string concatenation is and char\* res=strcat(m, n).

Step 6:using void main() we have to create objects for classes.

Step 7:stop.

Source code:

#include<iostream.h>

#include<conio.h>

#include<string.h>

Class fun

{

Public:

Char m[20],n[20],res[100];

Void com()

{

Cout<<”enter two strings:”;

Cin>>m>>n;

Int result=strcmp(m,n);

If(result==0)

{

Cout<<”strings are equal\n”;

}

Else

{

Cout<<”strings are not equal\n”;

}

}

Void operate()

{

Cout<<”string concatenation is:”;

Char\* res=strcat(m,n);

Cout<<res;

}

};

Void main()

{

Clrscr();

Fun f1;

F1.com();

F1.operate();

Getch();

}

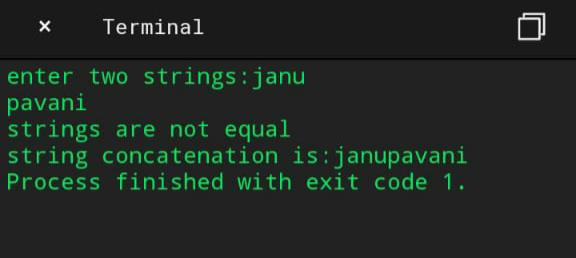
Expected output:

Enter two strings:Swarna Latha

Strings are not equal

String concatenation is:swarnaLatha

Actual Output:



RESULT: Above programme has been executed successfully

To print examination results of a student using inheritance

Experiment No: 25 Roll no:1872035

Date: Page:

Aim: To write a C++ program to consider an example of declaring the examination result .

Design three classes student, exam and result. The student has data members such as

Roll no, name. Create the class exam by inheriting the student class .the exam

Classadds Data members representing the marks scored in 5 subjects. Derive the

Result from exam And it has own data members like total, avg.

Algorithm:

Step 1:start.

Step 2:declare int rollno, char\* name inside class student.

Step 3:print enter name of the student and enter roll name read name and roll no.

Step 4:declare int m,n,s,h,k,tot and float avg inside class exam public student.

Step 5:using void compute() to print enter marks in five subjects and compute

Tot=(m+n+s+h+k);

Avg=(float)tot/5;

Step 6:next declare class result :public exam use void res() to print the average of a student.

Step 7: check condition if(avg>=35 && m>35 && n>35 && s>35 && h>35 && k>35) is true then

Print student gets passed else student gets failed and print average.

Step 8: in main() we have to create objects for above classes.

Step 9:stop.

Source code:

#include<iostream.h>

#include<conio.h>

Class student

{

Public:

Int rollno;

Char\* name;

Void details()

{

Cout<<”enter name of the student:”;

Cin>>name;

Cout<<”enter roll number:”;

Cin>>rollno;

}

};

Class exam:public student

{

Public:

Int m, n, s, h, k, tot;

Float avg;

Void compute()

{

Cout<<”enter marks in five subjects:”;

Cin>>m>>n>>s>>h>>k;

Tot=(m+n+s+h+k);

Avg=(float)tot/5;

}

};

Class result:public exam

{

Public:

Void res()

{

Cout<<”the average of a student is:”<<avg;

If(avg>=35&&m>35&&n>35&&s>35&&h>35&&k>35)

{

Cout<<”\nthe student gets passed”;

}

Else

{

Cout<<”\nthe student gets failed”;

}

}

};

Void main()

{

Clrscr();

Result r;

r.details();

r.compute();

r.res();

getch();

}

Expected output:

Enter name of the student: Janani

Enter roll number:1872035

Enter marks in five subjects:90

80

90

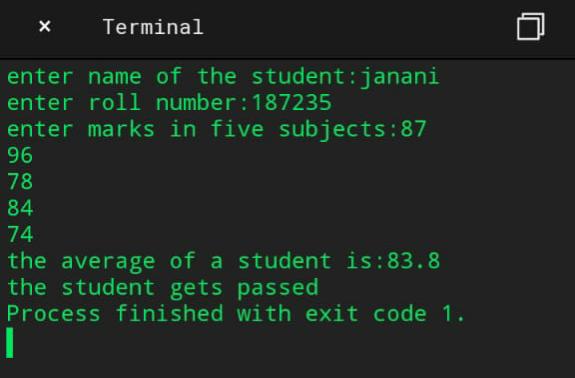
95

75

The average of a student is:85

The student gets passed

ACTUAL OUTPUT:



Result: Hence the program has been executed successfully

STACK USING ARRAYS

Experiment No: 26 Roll no:1872035

Date: Page no:

Aim: To write a Check program of stacks using arrays

Algorithm:

Step1:Start

Step2:Initialize stacks[100], choice, n, top, x, I as integers

Step3:Taking the inputs from the user

Step4:Calling the function according the requirement

Step5:Printing the result

Step6: Stop.

Source code:

#include<stdio.h>

Int stack[100],choice,n,top,x,I;

Void push(void);

Void pop(void);

Void display(void);

Int main()

{

//clrscr();

Top=-1;

Printf(“\n Enter the size of STACK[MAX=100]:”);

Scanf(“%d”,&n);

Printf(“\n\t STACK OPERATIONS USING ARRAY”);

Printf(“\n\t--------------------------------“);

Printf(“\n\t 1.PUSH\n\t 2.POP\n\t 3.DISPLAY\n\t 4.EXIT”);

Do

{

Printf(“\n Enter the Choice:”);

Scanf(“%d”,&choice);

Switch(choice)

{

Case 1:

{

Push();

Break;

}

Case 2:

{

Pop();

Break;

}

Case 3:

{

Display();

Break;

}

Case 4:

{

Printf(“\n\t EXIT POINT “);

Break;

}

Default:

{

Printf (“\n\t Please Enter a Valid Choice(1/2/3/4)”);

}

}

}

While(choice!=4);

Return 0;

}

Void push()

{

If(top>=n-1)

{

Printf(“\n\tSTACK is over flow”)

}

Else

{

Printf(“ Enter a value to be pushed:”);

Scanf(“%d”,&x);

Top++;

Stack[top]=x;

}

}

Void pop()

{

If(top<=-1)

{

Printf(“\n\t Stack is under flow”);

}

Else

{

Printf(“\n\t The popped elements is %d”,stack[top]);

Top--;

}

}

Void display()

{

If(top>=0)

{

Printf(“\n The elements in STACK \n”);

For(i=top; i>=0; i--)

Printf(“\n%d”,stack[i]);

Printf(“\n Press Next Choice”);

}

Else

{

Printf(“\n The STACK is empty”);

}

}

Expected output:

Enter the size of STACK[MAX=100]:4

STACK OPERATIONS USING ARRAY

--------------------------------

1.PUSH

2.POP

3.DISPLAY

4.EXIT

Enter the Choice:1

Enter a value to be pushed:7

Enter the Choice:1

Enter a value to be pushed:6

Enter the Choice:1

Enter a value to be pushed:

Enter the Choice:2

The popped elements is7

Enter the Choice:3

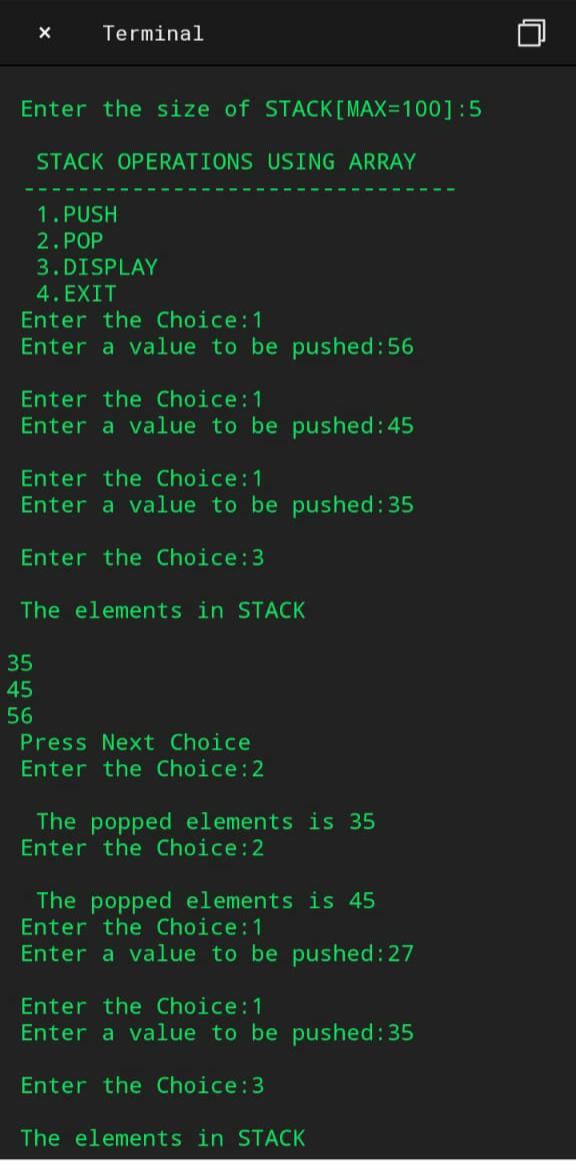
The elements in STACK

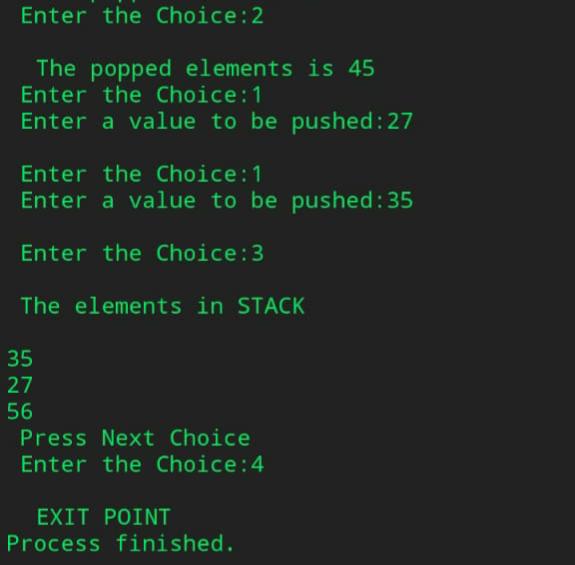
4 6

Press Next Choice

Enter the Choice:4

EXIT POINT

ACTUAL OUTPUT:



RESULT: Above programme has been executed successfully

STACK USING LINKED LIST

Experiment No: 27 Roll no:1872035

Date: Page no:

Aim: To write a Check program of stack using linked list

Algorithm:

Step1:Start

Step2:Initiaze the head ,next

Step3:Taking the inputs from the user

Step4:Calling the function according the requirement

Step5:Printing the result

Step6: Stop

Source code:

#include <stdio.h>

#include <stdlib.h>

Void push();

Void pop();

Void display();

Struct node

{

Int val;

Struct node \*next;

};

Struct node \*head;

Void main ()

{

Int choice=0;

Printf(“\n\*\*\*\*\*\*\*\*\*Stack operations using linked list\*\*\*\*\*\*\*\*\*\n”);

Printf(“\n----------------------------------------------\n”);

While(choice != 4)

{

Printf(“\n\nChose one from the below options…\n”);

Printf(“\n1.Push\n2.Pop\n3.Show\n4.Exit”);

Printf(“\n Enter your choice \n”);

Scanf(“%d”,&choice);

Switch(choice)

{

Case 1:

{

Push();

Break;

}

Case 2:

{

Pop();

Break;

}

Case 3:

{

Display();

Break;

}

Case 4:

{

Printf(“Exiting….”);

Break;

}

Default:

{

Printf(“Please Enter valid choice “);

}

};

}

}

Void push ()

{

Int val;

Struct node \*ptr = (struct node\*)malloc(sizeof(struct node));

If(ptr == NULL)

{

Printf(“not able to push the element”);

}

Else

{

Printf(“Enter the value”);

Scanf(“%d”,&val);

If(head==NULL)

{

Ptr->val = val;

Ptr -> next = NULL;

Head=ptr;

}

Else

{

Ptr->val = val;

Ptr->next = head;

Head=ptr;

}

Printf(“Item pushed”);

}

}

Void pop()

{

Int item;

Struct node \*ptr;

If (head == NULL)

{

Printf(“Underflow”);

}

Else

{

Item = head->val;

Ptr = head;

Head = head->next;

Free(ptr);

Printf(“Item popped”);

}

}

Void display()

{

Int I;

Struct node \*ptr;

Ptr=head;

If(ptr == NULL)

{

Printf(“Stack is empty\n”);

}

Else

{

Printf(“Printing Stack elements \n”);

While(ptr!=NULL)

{

Printf(“%d\n”,ptr->val);

Ptr = ptr->next;

}

}

}

OUT PUT:

\*\*\*\*\*\*\*\*\*Stack operations using linked list\*\*\*\*\*\*\*\*\*

Chose one from the below options…

1.Push

2.Pop

3.Show

4.Exit

Enter your choice

1

Enter the value2

Item pushed

Chose one from the below options…

1.Push

2.Pop

3.Show

4.Exit

Enter your choice

1

Enter the value4

Item pushed

Chose one from the below options…

1.Push

2.Pop

3.Show

4.Exit

Enter your choice

1

Enter the value6

Item pushed

Chose one from the below options…

1.Push

2.Pop

3.Show

4.Exit

Enter your choice

2

Item popped

Chose one from the below options…

1.Push

2.Pop

3.Show

4.Exit

Enter your choice

3

Printing Stack elements

4 2

Chose one from the below options…

1.Push

2.Pop

3.Show

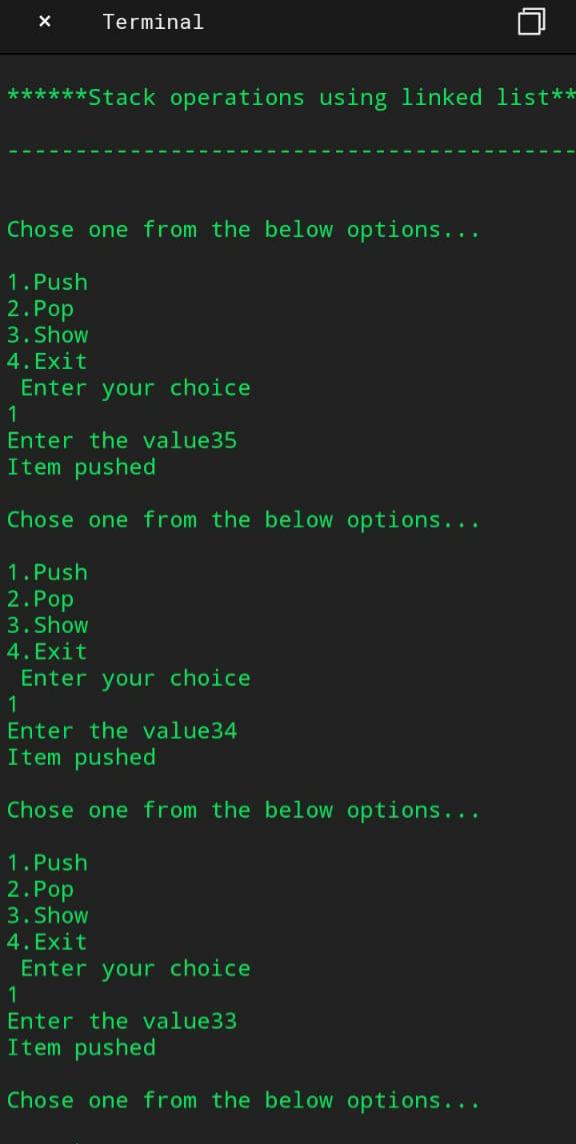
4.Exit

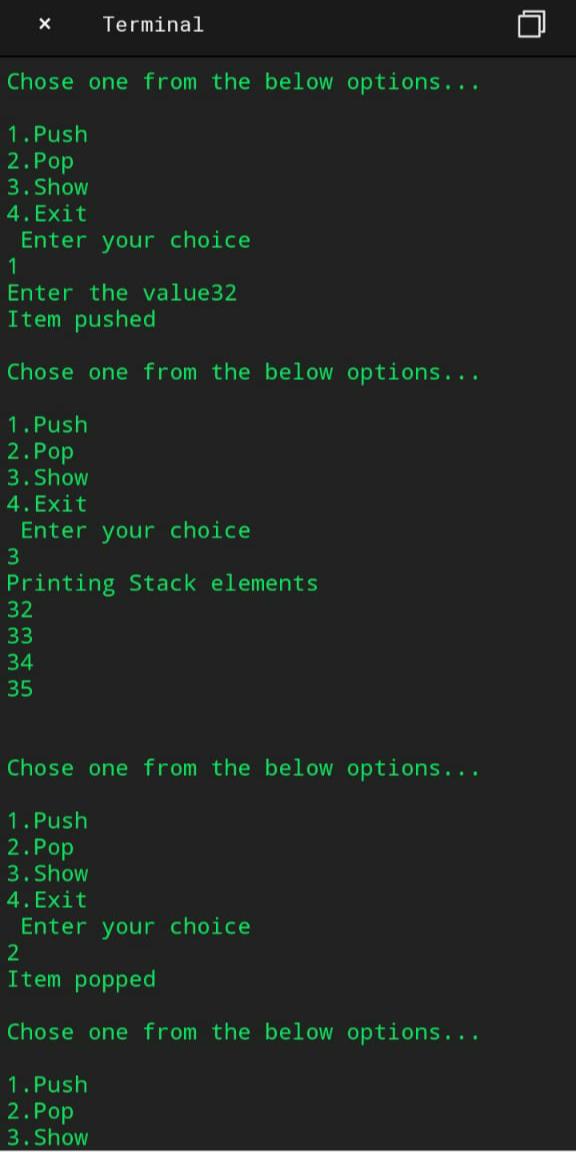
Enter your choice

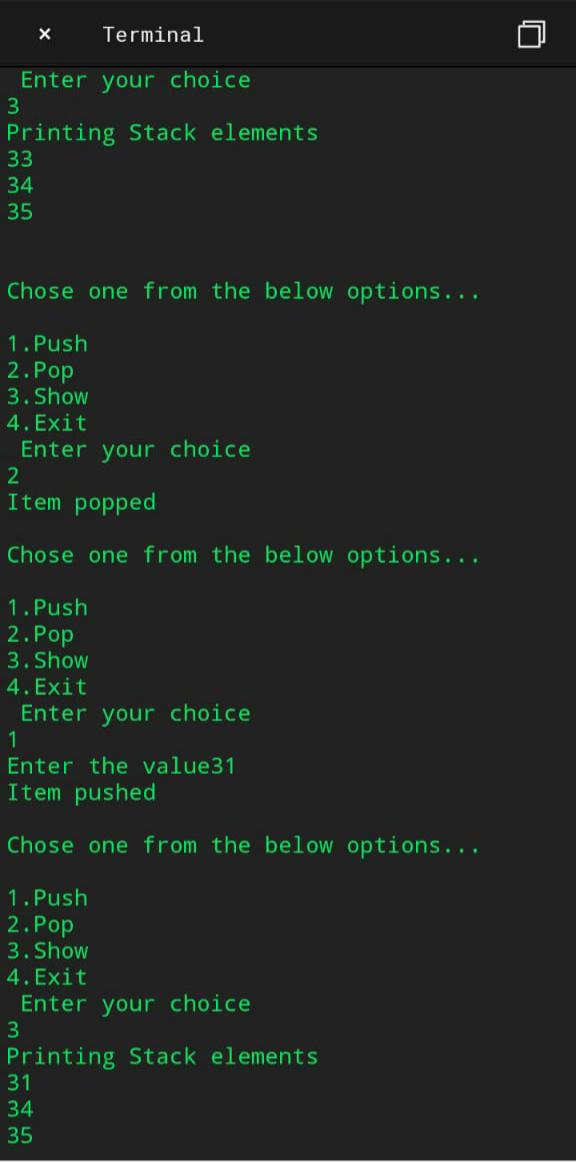
4

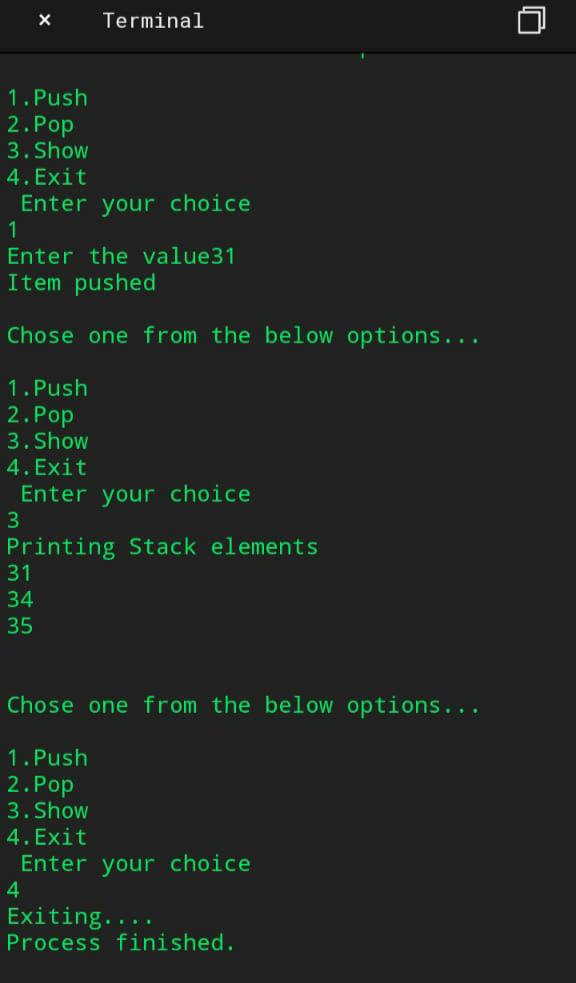
Exiting….Press any key to continue . . .

ACTUAL OUTPUT:









RESULT: ABOVE PROGRAMME HAS BEEN EXECUTED SUCCESSFULLY

BUBBLE SORTING

Experiment No:28 Roll no:1872035

Date:

Aim: To write a program for sorting using bubble sorting.

Algorithm:

Step1:Start

Step2:Initialize a[10], I, j, temp, n as integers

Step3:Read n Value fo

Step 4:for(i=0;i<n;i++)

Temp=a[i]

A[i]=a[j]

A[j]=temp

Step 5:Printing the result

Step 6:Stop.

Source code:

#include<stdio.h>

Int main()

{

Int a[10],I,j,temp,n;

Printf(“\n Enter the no.of Elements to Sort: \n”);

Scanf(“%d”,&n);

Printf(“\n Enter the Elements : \n”);

For(i=0; i<n; i++)

{

Scanf(“%d”,&a[i]);

}

For(i=0; i<n; i++)

For(j=i+1; j<n; j++)

{

If(a[i]>a[j])

{

Temp=a[i];

A[i]=a[j];

A[j]=temp;

}

}

For(i=0; i<n; i++)

{

Printf(“%d\t”,a[i]);

}

Return 0;

}

Expected output:

Enter the no.of Elements to Sort:4

Enter the Elements :

5

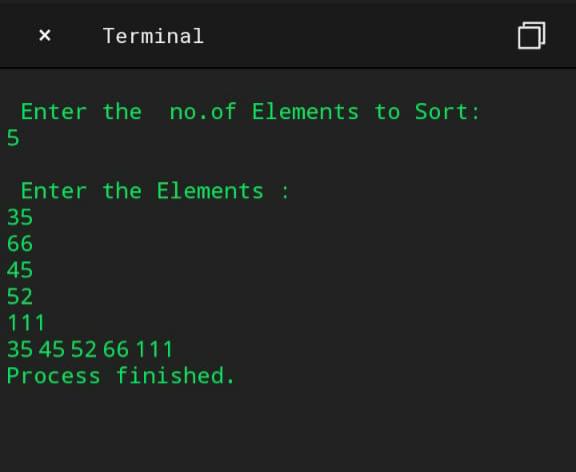
-5

-20

20

-20 -5 5 20

Actual output:



RESUlT : ABOVE PROGRAMME HAS BEEN EXECUTED SUCCESSFUL