#include<iostream>

using namespace std ;

class data

{

int acct\_no ;

float balance ;

public :

data (int a, float b)

{

acct\_no = a ;

balance = b ;

}

int display()

{

cout <<" Account Number = "<<acct\_no ;

cout<<" \n " ;

cout<<" Balance "<<balance ;

}

};

int main()

{

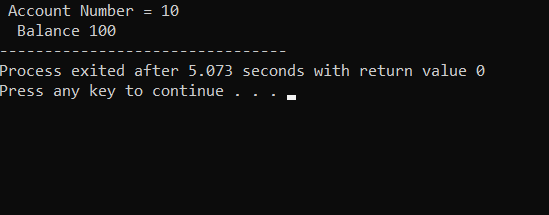
data d(10,100);

d.display ();

return 0 ;

}

**OUTPUT : --**



#include<iostream>

using namespace std ;

#include<conio.h>

class alpha

{

public :

static int count ;

alpha()

{

count ++;

cout<<"\n No.of Object Created =>"<<count;

}

~alpha()

{

cout<<"\n No of Object Destroyed=> "<<count ;

count --;

}

};

int alpha::count;

int main()

{

cout<<"\n\n ENTER MAIN \n ";

alpha A1,A2,A3,A4;

{

cout <<"\n\n ENTER THE BLOCK 1\n" ;

alpha A5;

}

{

cout<<"\n\n ENTER BLOCK 2\n";

alpha A6;

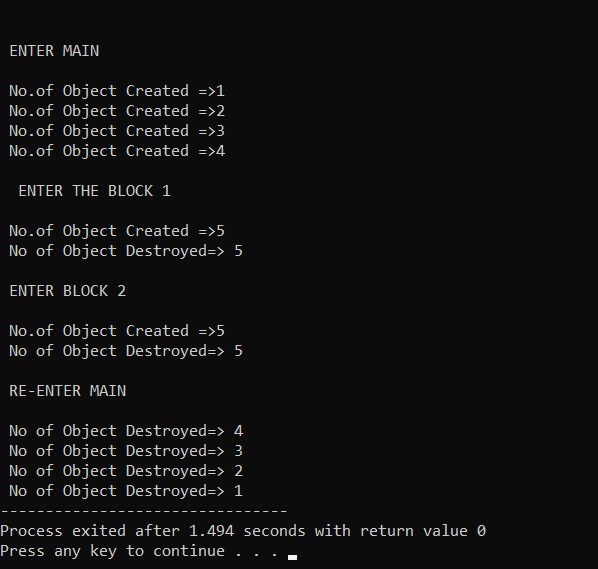
}

cout<<"\n\n RE-ENTER MAIN \n";

getch();

}

**OUTPUT : --**



#include<iostream>

using namespace std ;

#include<conio.h>

class Road

{

int km,m ;

public :

void getdata ()

{

cout<<"Enter kilometer =>";

cin>>km ;

cout<<"Enter meter =>";

cin>>m ;

}

void calculate (Road d1,Road d2)

{

Road d3;

d3.km = d1.km +d2.km;

d3.m = d1 .m + d2.m;

if(d3.m>1000)

{

d3.km = d3.km +1;

d3.m = d3.m -1000;

}

cout<<d3.km<<"."<<d3.m<<"km";

}

};

int main()

{

Road d1,d2,d3;

cout<<"Enter the Distance Between 1st and 2nd City :\n";

d1.getdata();

cout<<"\n\n Enter the Distance Between 1st and 2nd City :\n";

d2.getdata();

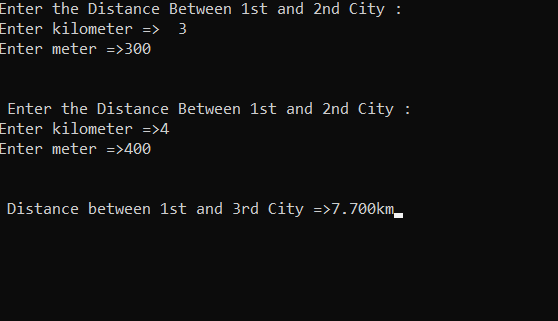
cout<<"\n\n Distance between 1st and 3rd City =>";

d3.calculate (d1,d2);

getch();

}

**OUTPUT : --**



#include<iostream>

using namespace std ;

#include<string.h>

class AddString

{

public :

char s1[25] ,s2[25];

AddString(char str1[],char str2[])

{

strcpy(this ->s1,str1);

strcpy(this ->s2,str2);

}

int operator + ()

{

cout<<"\n Concatenation :"<<strcpy(s1,s2);

}

};

int main()

{

char str1[] = " Object Oriented ";

char str2[] = " Programming Language";

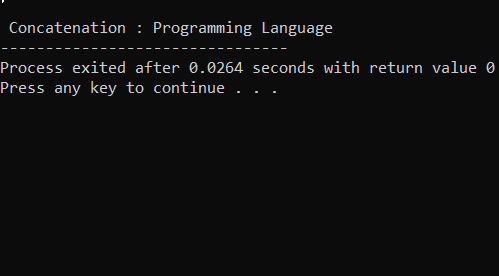
AddString a1(str1, str2);

+a1;

return 0 ;

}

**OUTPUT : --**



#include<iostream>

using namespace std ;

class Deposit

{

int rupee , paisa ;

public :

Deposit ()

{

rupee = 0 ;

paisa = 0 ;

}

int getdata ()

{

cout<<"\n Enter Rupee =>";

cin>>rupee ;

cout<<"\n Enter Paisa =>";

cin>>paisa ;

}

int putdata ()

{

cout<<rupee<<"."<<paisa ;

}

Deposit operator + (Deposit d)

{

Deposit t ;

t.rupee=rupee+d.rupee ;

t.paisa=paisa+d.paisa ;

if (t.paisa>100)

{

t.rupee=t.rupee+1 ;

t.paisa=t.paisa-100 ;

}

return t ;

}

};

int main()

{

int i ,n ;

Deposit oD[10],total ;

cout<<" Enter Total Number of depositor N=>" ;

cin>>n ;

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

{

cout<<"\n\n Enter the Deposited amount for "<<i+1<<"Depositor :" ;

oD[i].getdata() ;

cout<<"Balance of"<<i+1<<"Depositor =>" ;

oD[i].putdata() ;

}

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

{

total =total+oD[i];

}

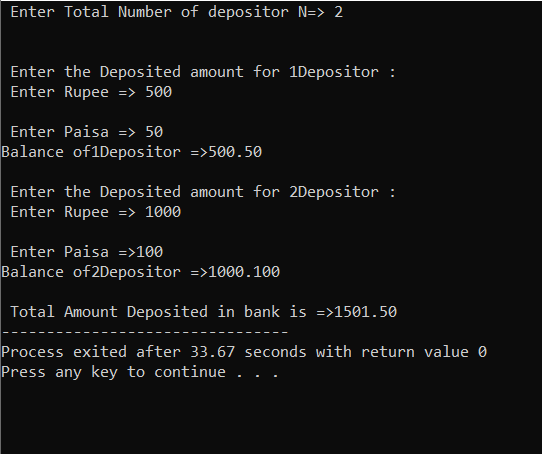
cout<<"\n\n Total Amount Deposited in bank is =>" ;

total.putdata() ;

return 0 ;

}

**OUTPUT : --**



#include<iostream>

using namespace std ;

class Event

{

int hours,minutes,seconds,total ;

public :

Event()

{

hours=minutes=seconds=total=0 ;

}

void getdata ()

{

cout<<" \n Enter the Hours " ;

cin>>hours ;

cout<<" \n Enter the Minutes " ;

cin>>minutes ;

cout<<" \n Enter the Seconds " ;

cin>>seconds ;

}

int putdata ()

{

cout<<hours<<":"<<minutes<<":"<<seconds ;

}

Event operator-(Event e2)

{

Event diff;

diff.hours=e2.hours-hours ;

diff.minutes=e2.minutes-minutes ;

diff.seconds=e2.seconds-seconds ;

if (diff.minutes<0)

{

diff.hours=diff.hours-1 ;

diff.minutes=diff.minutes+60 ;

}

if (diff.seconds<0)

{

diff.minutes=diff.minutes-1 ;

diff.seconds=diff.seconds+60 ;

}

return diff ;

}

};

int main()

{

Event oE1 ;

cout<<" \n Enter the time [24 Hour Format] of 1nd Event:" ;

oE1.getdata();

cout<<" 1st Event Time =>" ;

oE1.putdata() ;

Event oE2 ;

cout<<" \n Enter the time [24 Hour Format] of 2nd Event:" ;

oE2.getdata();

cout<<" 2nd Event Time =>" ;

oE2.putdata();

Event diff;

diff=oE1-oE2 ;

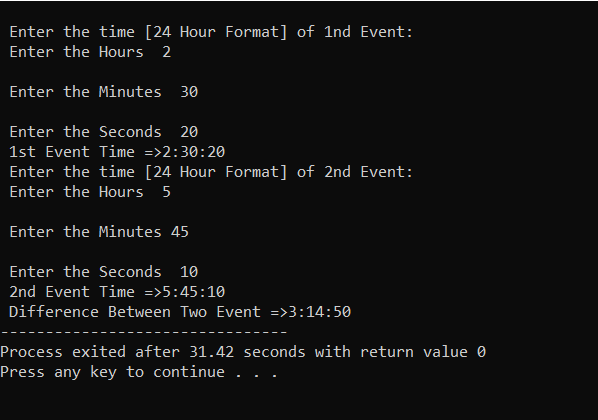
cout<<" \n Difference Between Two Event =>" ;

diff.putdata() ;

return 0 ;

}

**OUTPUT : --**



#include<iostream>

using namespace std ;

class B

{

int a ;

public :

int b ;

void get\_ab()

{

cout<<"\n Enter the value of A & B => " ;

cin>>a>>b ;

}

int get\_a ()

{

return a ;

}

void show\_a()

{

cout<<"\n A =>" <<a ;

}

};

class D : public B

{

int c ;

public :

void mul ()

{

c=b\*get\_a();

}

void display ()

{

cout<<" \n\n A=> "<<get\_a() ;

cout<<" \n B=> "<<b ;

cout<<" \n C=> "<<c ;

}

};

int main()

{

D d ;

d.get\_ab() ;

d.mul() ;

d.show\_a() ;

d.display() ;

d.b=20 ;

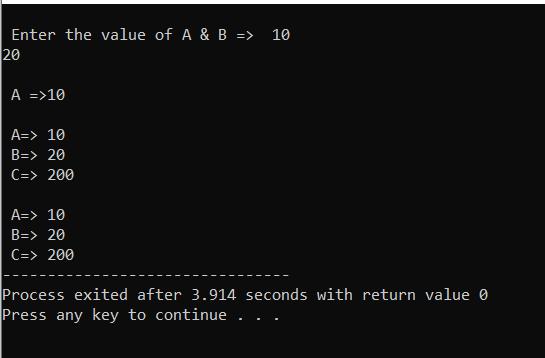
d.mul() ;

d.display() ;

return 0 ;

}

**OUTPUT : --**



#include<iostream>

using namespace std ;

class M

{

protected :

int m ;

public :

void get\_m(int x)

{

m=x ;

}

};

class N

{

protected :

int n ;

public :

void get\_n(int y)

{

n=y ;

}

};

class P: public M, public N

{

public :

void display ()

{

cout<<" \nm ="<<m ;

cout<<" \nn ="<<n ;

cout<<"\nm\*n "<<m\*n ;

}

};

int main()

{

P oP ;

oP.get\_m(10) ;

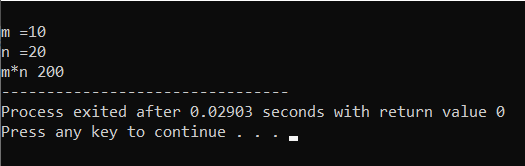
oP.get\_n(20) ;

oP.display() ;

return 0;

}

**OUTPUT : --**



#include<iostream>

using namespace std ;

class student

{

protected :

int roll\_number ;

public :

void get\_number (int a)

{

roll\_number = a ;

}

void put\_number ()

{

cout<<" \n Roll Number :"<<roll\_number ;

}

};

class test : public student

{

protected :

float sub1 , sub2 ;

public :

int get\_marks( float x, float y)

{

sub1 = x ;

sub2 = y ;

}

int put\_marks ()

{

cout<<" \n Marks in Sub1 ="<<sub1 ;

cout<<" \n Marks in Sub2 ="<<sub2 ;

}

};

class result : public test

{

float total ;

public :

int display ()

{

total = sub1+sub2 ;

put\_number() ;

put\_marks () ;

cout<<" \n Total =" <<total ;

}

};

int main()

{

result stud1 ;

stud1.get\_number(1) ;

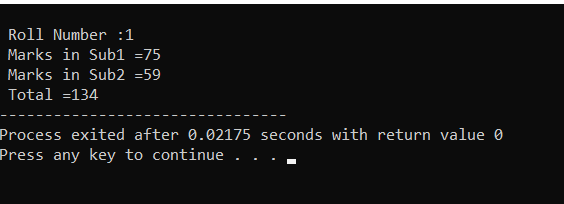
stud1.get\_marks(75,59) ;

stud1.display() ;

return 0 ;

}

**OUTPUT : --**



#include<iostream>

using namespace std ;

class M

{

protected :

int m ;

public :

void get\_m(int x)

{

m=x ;

}

};

class N

{

protected :

int n ;

public :

void get\_n(int y)

{

n=y ;

}

};

class P: public M, public N

{

public :

void display ()

{

cout<<" \nm ="<<m ;

cout<<" \nn ="<<n ;

cout<<"\nm\*n "<<m\*n ;

}

};

int main()

{

P oP ;

oP.get\_m(10) ;

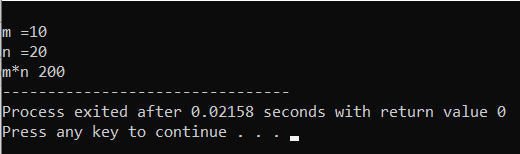
oP.get\_n(20) ;

oP.display() ;

return 0;

}

**OUTPUT :--**



#include<iostream>

using namespace std ;

class Side

{

protected :

int L ;

public :

void set\_values()

{

cout <<" \n Enter the number => " ;

cin>>L ;

}

};

class Square : public Side

{

public :

void sq()

{

cout<<" Square of "<<L<< " is " <<L\*L ;

}

};

class Cube : public Side

{

public :

void cub()

{

cout <<" Cube of "<<L<<" is "<<L\*L\*L ;

}

};

int main()

{

Square oS ;

oS.set\_values() ;

oS.sq() ;

Cube oC ;

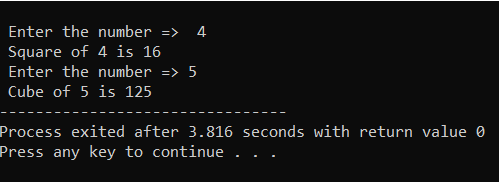
oC.set\_values() ;

oC.cub() ;

return 0 ;

}

**OUTPUT :--**



#include<iostream>

using namespace std ;

class Base

{

public :

void display ()

{

cout <<" \n Display Base " ;

}

virtual show ()

{

cout <<" \n Show Base " ;

}

};

class Derive: public Base

{

public :

int display()

{

cout << " \n Display Derive " ;

}

int show ()

{

cout <<" \n Show Derive " ;

}

};

int main ()

{

Base oB ;

Derive oD ;

Base \* bptr ;

cout <<" \n bptr point to the Base " ;

bptr = & oB ;

bptr -> display () ;

bptr -> show () ;

cout <<" \n\n bptr point to the Derive ";

bptr = &oD ;

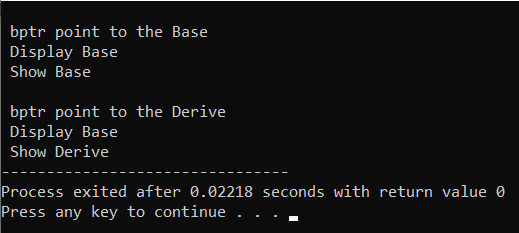
bptr ->display () ;

bptr ->show ();

return 0 ;

}

**OUTPUT : --**



#include<iostream>

using namespace std ;

class Base

{

public :

void display ()

{

cout<<" \n display Base " ;

}

virtual void show ()= 0 ;

};

class Drive:public Base

{

public :

void display()

{

cout <<" \n Display Drive " ;

}

void show ()

{

cout<<" \n Show Drive " ;

}

};

int main()

{

Drive oD ;

Base\*bptr ;

cout<<" \n\n bptr point to the Drive " ;

bptr=&oD ;

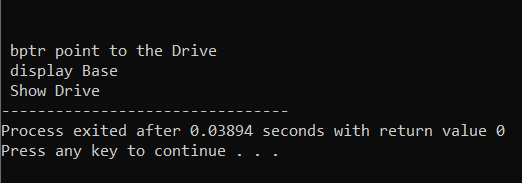
bptr->display () ;

bptr->show() ;

return 0 ;

}

**OUTPUT : --**



#include<iostream>

using namespace std ;

int main()

{

int x ;

cout<<" Enter the Number =>" ;

cin >>x ;

try

{

if (x==0)

throw 'x' ;

if (x>1)

throw 1 ;

if(x<0)

throw -1.0 ;

}

catch (char c)

{

cout <<" Cought Zero Error " ;

}

catch (int i)

{

cout <<" Cought Positive Number " ;

}

catch (double d)

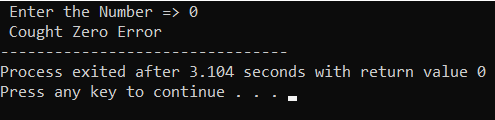
{

cout <<" Cought Negative Number " ;

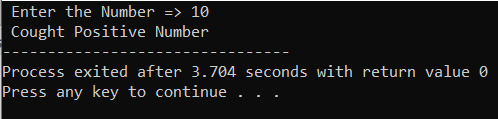
}

}

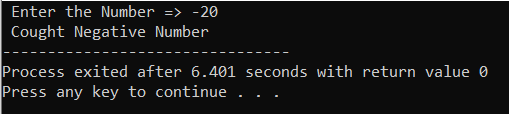
**OUTPUT 1 ) :--**



**OUTPUT 2) :--**



**OUTPUT 3) :--**



#include<iostream>

using namespace std ;

int main()

{

int a, b ;

cout<<" Enter the Two Number =>" ;

cin>>a>>b ;

try

{

if(b!=0)

cout<<a<<"/"<<b<<"="<<a/b ;

else

throw (b) ;

}

catch (int x)

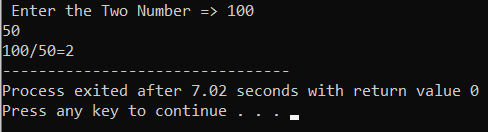
{

cout <<" Exception Caught Divide By Zero " ;

}

}

**OUTPUT 1) : --**



**OUTPUT 2) : --**

