Object oriented programming with c++

1. #include<iostream>

using namespace std;

int l,b,h,a,v;

class area

{

public:

virtual void getdata()

{

a=l\*b;

cout<<"area="<<a;

}

};

class volume:public area

{

public:

void getdata()

{

cout<<"enter l,b,h values";

cin>>l>>b>>h;

v=l\*b\*h;

cout<<"volume="<<v;

}

};

main()

{

area \*j;

volume k;

j=&k;

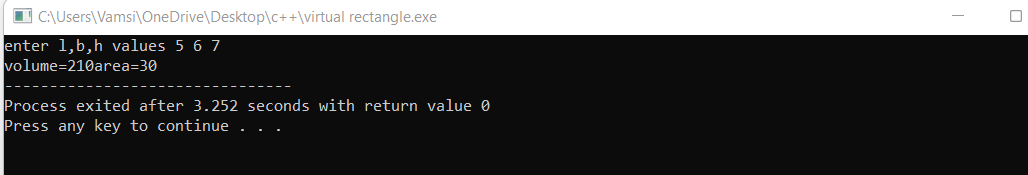
j->getdata();

area t;

t.getdata();

}

Output:



2.

#include<iostream>

using namespace std;

int roi;

class bank

{

public:

virtual void getrateofinterest()

{

cout<<"rate of interest of bank";

cin>>roi;

cout<<"roi="<<roi<<endl;

}

};

class sbi:public bank

{

public:

void getrateofinterest()

{

cout<<"rate of interest of sbi";

cin>>roi;

cout<<"rate of interest of sbi="<<roi<<endl;

}

};

class icic:public bank

{

public:

void getrateofinterest()

{

cout<<"enter rate of interest of icic";

cin>>roi;

cout<<"rate of interest of icic="<<roi<<endl;

}

};

class axis:public bank

{

public:

void getrateofinterest()

{

cout<<"enter rate of interest of axis";

cin>>roi;

cout<<"rate of interest of axis="<<roi<<endl;

}

};

main()

{

bank \*a;

sbi b;

a=&b;

a->getrateofinterest();

icic c;

a=&c;

a->getrateofinterest();

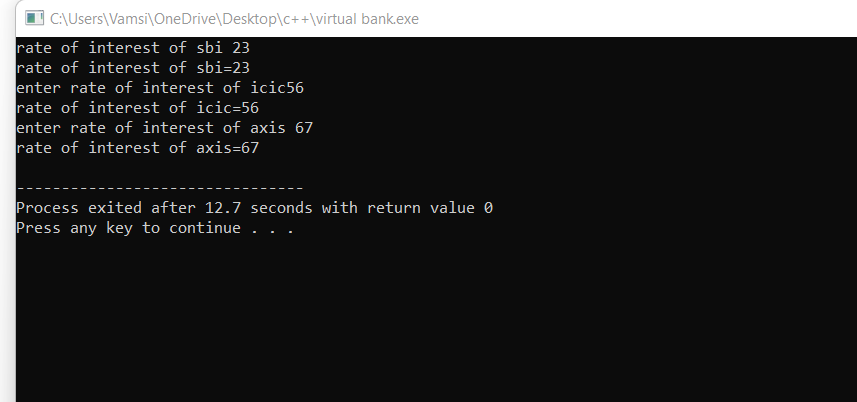
axis x;

a=&x;

a->getrateofinterest();

}

Output:



3.

#include<iostream>

using namespace std;

main()

{

int x=-1;

try

{

if(x<0)

{

throw x;

cout<<"x is small";

}

}

catch(int x)

{

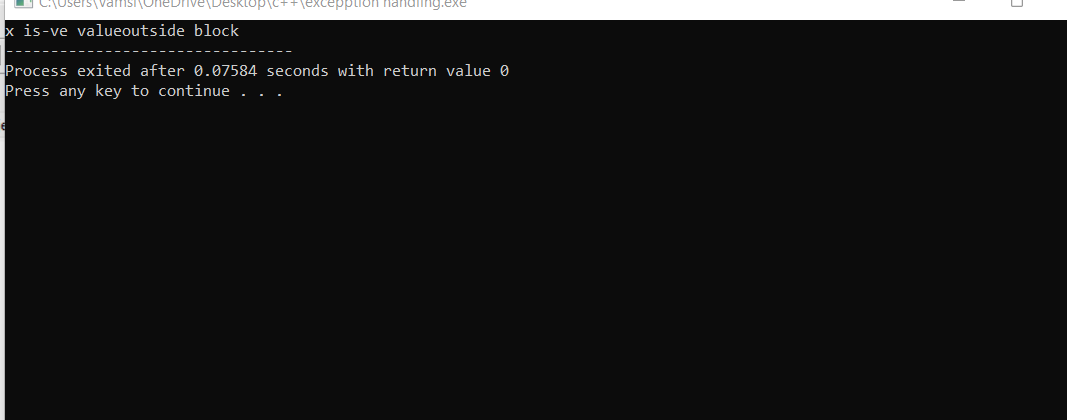
cout<<"x is-ve value";

}

cout<<"outside block";

}

Output:



4.

#include<iostream>

#include<exception>

using namespace std;

struct myexception:public exception

{

const char \*what()const throw()

{

return "c++ exception";

}

};

main()

{

try

{

throw myexception();

}

catch(myexception&e)

{

std::cout<<"myexception caught"<<std::endl;

std::cout<<e.what()<<std::endl;

}

catch(std::exception e)

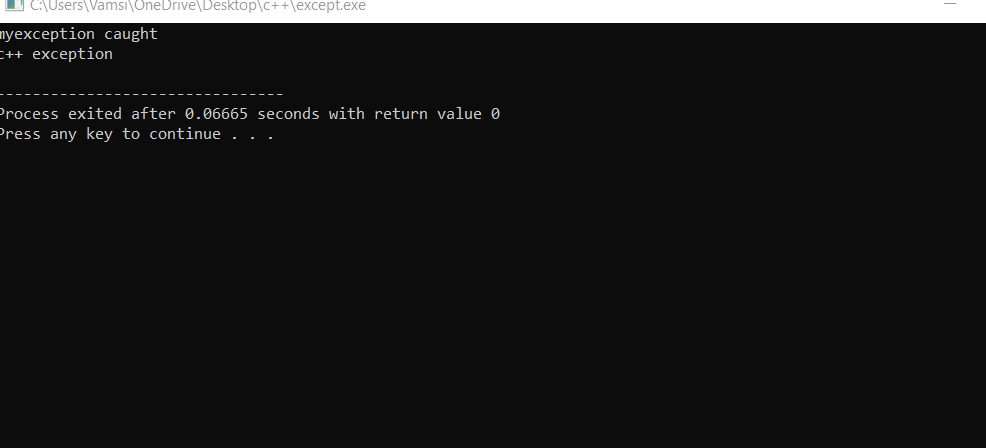
{

//other errors

}

}

Output:



5.

#include<iostream>

using namespace std;

main()

{

try

{

throw 10;

}

catch(char \*exp)

{

cout<<"caught";

}

catch(int a)

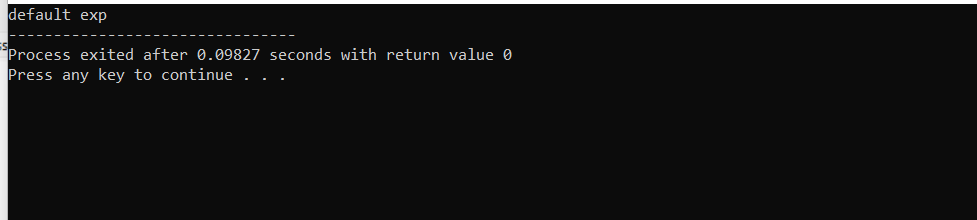
{

cout<<"default exp";

}

}

Output:



6. #include<iostream>

using namespace std;

double division(int a,int b)

{

if(b == 0)

{

throw "division by zero";

}

return (a/b);

}

main()

{

int x = 50,y = 00;

double z = 0;

try

{

z = division(x,y);

cout<<z<<endl;

}

catch (const char\*msg)

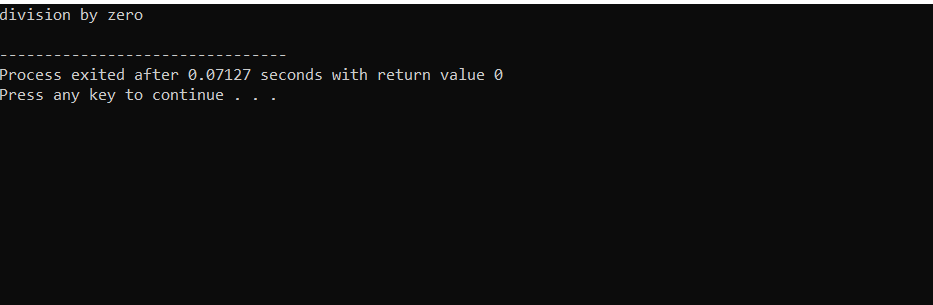
{

cerr<<msg<<endl;

}

}

Output:



7. #include<iostream>

using namespace std;

class A

{

public:

virtual void getdata()

{

cout<<"A";

}

};

class B:public A

{

public:

void getdata()

{

cout<<"B"<<endl;

}

};

main()

{

B b;

b.getdata();

b.getdata();

}

