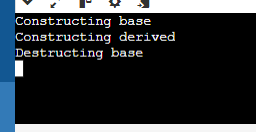
1. #include<iostream>
2. using namespace std;
3. class base
4. {
5. public:
6. base()
7. {cout<<"Constructing base \n"; }
8. ~base()
9. {cout<<"Destructing base \n"; }
10. };
11. class derived: public base{
12. public:
13. derived()
14. {cout<<"Constructing derived \n"; }
15. ~derived()
16. {cout<<"Destructing derived \n"; }
17. };
18. int main(void)
19. {
20. derived \*d = new derived();
21. base \*b = d;
22. delete b;
23. getchar();
24. return 0;
25. }

Output:



#include <iostream>

class B;

class A {

public:

void showB(B&);

};

class B{

private:

int b;

public:

B() { b = 0;}

friend void A::showB(B& x);

};

void A::showB(B& x)

{

std::cout<< "B::b = "<<x.b;

}

int main()

{

A a;

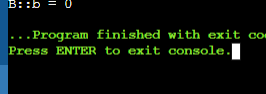
B x;

a.showB(x);

return 0;

}

Output



include <iostream>

class A {

private:

int a;

public:

A() { a = 0 ;}

friend class B;

};

class B{

private:

int b;

public:

void showA(A& x)

{

// since B is friend pf A, it can access

// private members of A

std::cout<<"A::a"<< x.a;

}

};

int main()

{

A a;

B b;

b.showA(a);

return 0;

}

Output:

