```
1 class X;
2
3 class Y : public X {
4     int b;
5 };
6
7 class X {
8     int a;
9 };
```

What might happen if you try to compile this code?

```
1  struct Foo {
2      Foo(int a, int b) : a_(a), b_(b) {}
3      int a_;
4      int b_;
5  };
6
7  struct Bar : Foo {
8      Bar(int a, int b, int c) : a_(a), b_(b), c_(c) {}
9      int c_;
10 };
```

This code does not compile. Why not? How to fix?

```
 1 #include <iostream>
 2
 3 template<typename T> void p(T x) { std::cout << x; }
 4
 5 struct Foo {
 6     Foo() { p(1); }
 7     ~Foo() { p(2); }
 8 };
 9
10 struct Bar : Foo {
11     Bar() { p(3); }
12     ~Bar() { p(4); }
13 };
14
15 struct Gaz : Bar {
16     Gaz() { p(5); }
17     ~Gaz() { p(6); }
18 };
19
20 int main() {
21     p('-');
22     Foo f;
23     p('-');
24     Bar b;
25     p('-');
26     Gaz g;
27     p('-');
28 }
```

What will this code print out?

```
 1 #include <iostream>
 2
 3 template<typename T> void p(T x) { std::cout << x; }
 4
 5 struct A {
 6     A() { p(1); }
 7     ~A() { p(2); }
 8 };
 9
10 struct B : A {
11     B() { p(3); }
12     ~B() { p(4); }
13 };
14
15 struct C : A, B {
16     C() { p(5); }
17     ~C() { p(6); }
18 };
19
20 struct D {
21     D() { p(7); }
22     ~D() { p(8); }
23 };
24
25 struct E : D, B, C {
26     E() { p(9); }
27     ~E() { p(0); }
28 };
29
30 int main() {
31     E e;
32     p('-');
33 }
```

What might happen if you try to compile, link and run this code?

```
 1 class A {
 2     // ...
 3 };
 4
 5 class B : private A {
 6     // ...
 7 };
 8
 9 class C : protected A {
10     // ...
11 };
12
13 class D : public A {
14     // ...
15 };
16
17 int main() {
18     A * a1 = new B();
19     A * a2 = new C();
20     A * a3 = new D();
21 }
```

Consider this code. What might happen when you try to compile this code? What does private inheritance mean? When should you use protected inheritance?

```
 1 #include <iostream>
 2
 3 template<typename T> void p(T x) { std::cout << x; }
 4
 5 class A {
 6     char * s;
 7 public:
 8     A() { p('A'); s = new char[1024]; }
 9     ~A() { p('a'); delete[] s; }
10 };
11
12 class Foo {
13     A a;
14 public:
15     virtual void print() = 0;
16 };
17
18 class Bar : public Foo {
19     A a;
20 public:
21     virtual void print() { p(1); };
22 };
23
24 int main() {
25     p('-');
26     Foo * f = new Bar;
27     p('-');
28     f->print();
29     p('-');
30     delete f;
31     p('-');
32 }
```

What will this code print out? Please criticize.

```
 1 #include <iostream>
 2
 3 template<typename T> void p(T x) {
 4     std::cout << x;
 5 }
 6
 7 struct A {
 8     A() { p('A'); }
 9     ~A() { p('a'); }
10 };
11
12 struct B : A {
13     B() { p('B'); }
14     ~B() { p('b'); }
15 };
16
17 class Foo {
18 public:
19     virtual A * create() const {
20         p(1);
21         return new A();
22     }
23 };
24
25 class Bar : public Foo {
26 public:
27     virtual B * create() const {
28         p(2);
29         return new B();
30     }
31 };
```

```
33 int main() {
34     const Foo & f1 = Foo();
35     A * a1 = f1.create();
36     delete a1;
37     p('-');
38
39     const Foo & f2 = Bar();
40     A * a2 = f2.create();
41     delete a2;
42     p('-');
43
44     const Bar & f3 = Bar();
45     B * a3 = f3.create();
46     delete a3;
47     p('-');
48 }
```

What might happen if you try to compile, link and run this program?

```
 1 #include <iostream>
 2
 3 struct A {
 4     void func() { std::cout << 'A'; }
 5 };
 6
 7 struct B {
 8     void func() { std::cout << 'B'; }
 9 };
10
11 struct C {
12     void run() { std::cout << 'C'; }
13 };
14
15 template<typename T> class Foo {
16 public:
17     Foo(const T & t) : value(t) { }
18     void run();
19 private:
20     T value;
21 };
22
23 template<typename T> void Foo<T>::run() {
24     value.func();
25 }
```

```
27 int main() {
28     A a;
29     Foo<A> fa(a);
30     fa.run();
31
32     B b;
33     Foo<B> fb(b);
34     fb.run();
35
36     C c;
37     Foo<C> fc(c);
38     c.run();
39 }
```

What might happen if you try to compile, link and run this program?

```
 1 #include <iostream>
 2
 3 template<typename T, int maxsize> class Foo {
 4     T vector_[maxsize];
 5 public:
 6     int size() {
 7         return sizeof(vector_) / sizeof(T);
 8     }
 9     // ...
10 };
11
12 int main() {
13     Foo<int, 4> a;
14     std::cout << a.size();
15
16     int sz=2;
17     Foo<char, sz> b;
18     std::cout << b.size();
19 }
```

What might happen if you try to compile, link and run this program?

```
 1 #include <iostream>
 2
 3 template<typename T, T threshold> bool gt(T t) {
 4     std::cout << t << '>' << threshold;
 5     return t > threshold;
 6 }
 7
 8 int main() {
 9     std::cout << std::boolalpha << "=" << gt<int,5>(3);
10 }
```

What might happen if you try to compile, link and run this program?

```
 1 #include <iostream>
 2 #include <sstream>
 3
 4 template<typename T> bool istrue(T t) {
 5     std::stringstream s;
 6     s << t;
 7     return s.str() == "42";
 8 }
 9
10 int main() {
11     std::cout << std::boolalpha;
12
13     int t = 42;
14     std::cout << istrue(t) << std::endl;
15
16     float f = 43;
17     std::cout << istrue(f) << std::endl;
18
19     char * s = "42";
20     std::cout << istrue(s) << std::endl;
21
22     char c = 42;
23     std::cout << istrue(c) << std::endl;
24 }
```

What might happen if you try to compile, link and run this program?

```
 1 #include <iostream>
 2 #include <sstream>
 3
 4 template<typename T> void foo(T t) {
 5     std::cout << 'a';
 6 }
 7
 8 void foo(float t) {
 9     std::cout << 'b';
10 }
11
12 void foo(long t) {
13     std::cout << 'c';
14 }
15
16 int main() {
17     foo(42);
18     foo("42");
19     foo(42L);
20     foo(42.3);
21 }
```

What might happen if you try to compile, link and run this program?

```
 1 #include <iostream>
 2
 3 template<typename T> void foo(T t) {
 4     std::cout << 1;
 5 }
 6
 7 template<typename T> void foo(T * t) {
 8     std::cout << 2;
 9 }
10
11 template<> void foo(char) {
12     std::cout << 4;
13 }
14
15 int main() {
16     int a;
17     foo(a);
18     foo(&a);
19     char b;
20     foo(b);
21     foo(&b);
22 }
```

What might happen if you try to compile, link and run this program?