有如下C++代码：

struct A{

  void foo(){printf("foo");}

  virtual void bar(){printf("bar");}

  A(){bar();}

};

struct B:A{

  void foo(){printf("b\_foo");}

  void bar(){printf("b\_bar");}

};

那么

A \*p=new B;

p->foo();

p->bar();

输出为：

1. barfoob\_bar
2. foobarb\_bar
3. barfoob\_foo
4. foobarb\_fpp

解析：

A \*p=newB;// A类指针指向一个实例化对象B，

B类继承A类，先调用父类的无参构造函数，bar()输出bar，B类没有自己显示定义的构造函数。

p->foo();//执行B类里的foo()函数，因为foo不是虚函数，所以直接调用父类的foo函数，输出foo

p->bar();//执行B类的bar()函数, 该函数为虚函数，调用子类的实现，输出b\_bar