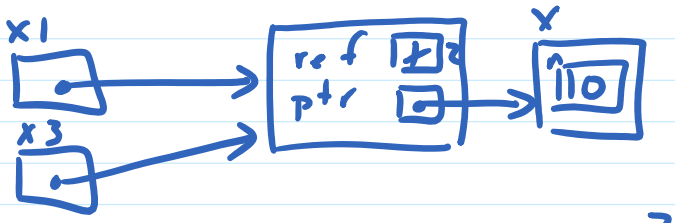
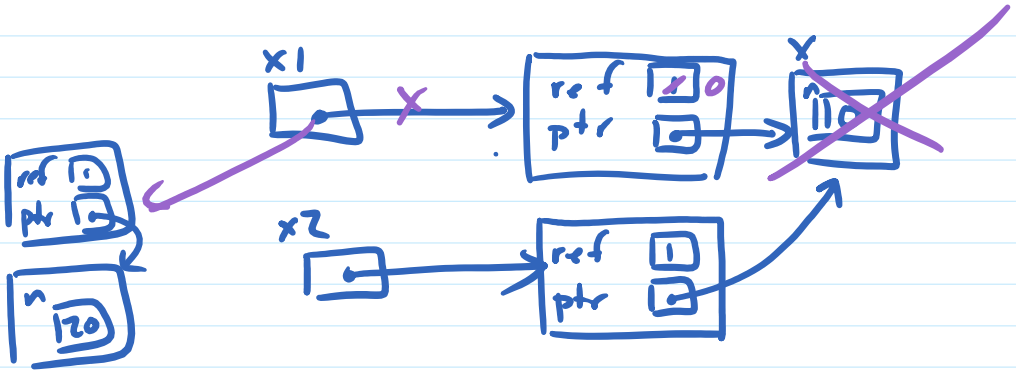
```
class X
{
public:
  X(int n) { this->n = n; }
  ~X() { }
  int get() const { return n; }
  std::shared_ptr<X> sharedPointer() { return std::shared_ptr<X>(this); }

private:
  int n;
};

int main(int argc, char **argv)
{
  std::shared_ptr<X> x1 = std::make_shared<X>(10);
  std::shared_ptr<X> x2 = x1->sharedPointer();

  std::cout << "x1=" << x1->get() << ", x2=" << x2->get() << std::endl;
  x1 = std::make_shared<X>(20);
  std::cout << "x1=" << x1->get() << ", x2=" << x2->get() << std::endl;
}
```



x3 = x1

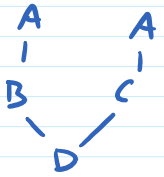


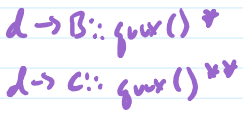
Model : managing data

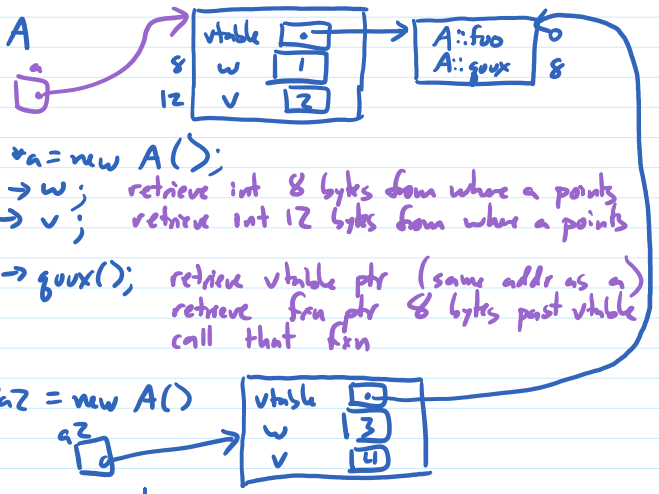
View : displays data

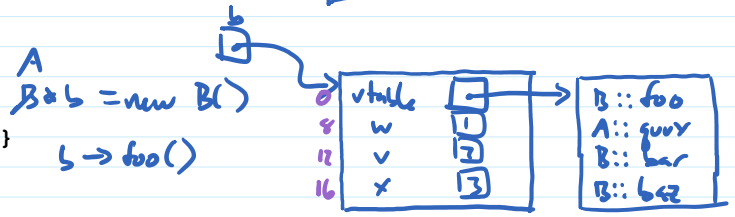
Controller : handles user input

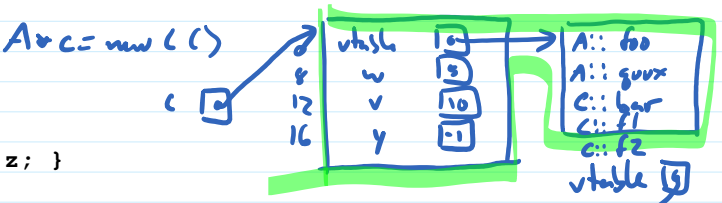
```
class A
{
  public:
    virtual void foo();
    virtual void quux();
  private:
    int w, v;
};
class B : public A
{
  public:
    void foo();
    virtual void bar();
    virtual void baz();
  private:
    int x;
};
class C : public A
{
  public:
    virtual void bar();
     virtual void f1() { f2(); }
     virtual void f2();
  private:
    int y;
};
class D : public B, public C
{
  public:
    virtual void baz();
     virtual void f2() { cout << z; }
  private:
    int z;
};
```

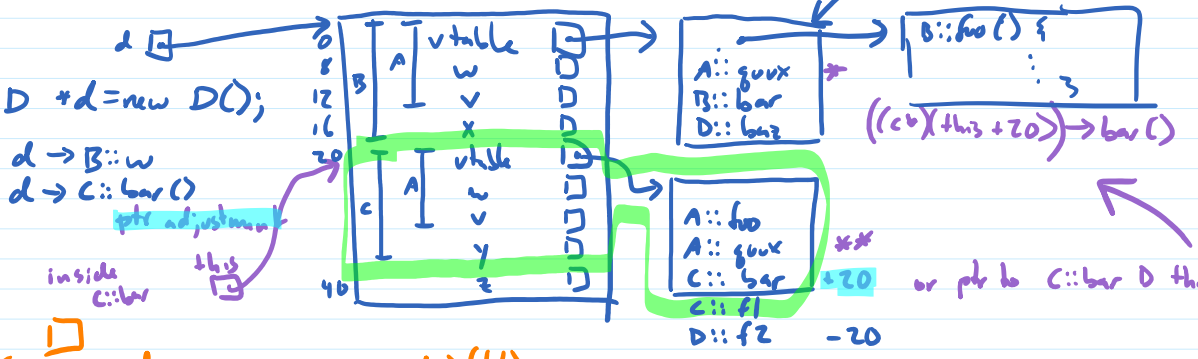


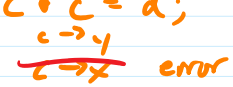




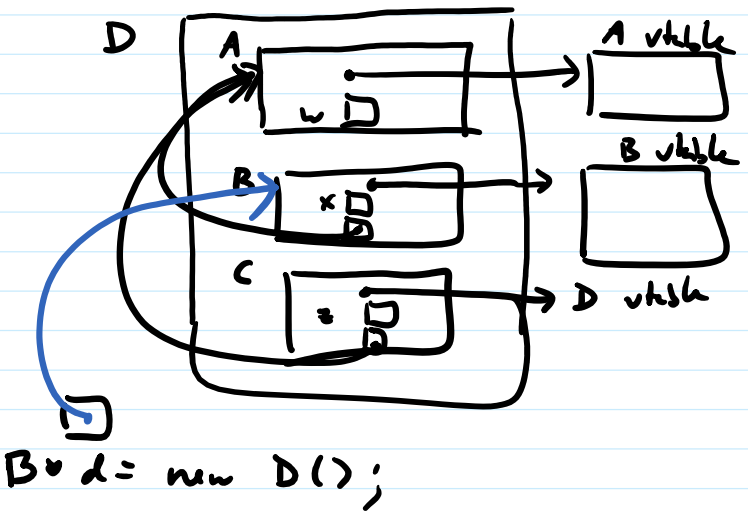








```
class A
{
  public:
    virtual void foo();
  private:
    int w;
};
class B : public virtual A
{
  public:
    void foo();
    virtual void bar();
    virtual void baz();
  private:
    int x;
};
class C : public virtual A
{
  public:
    virtual void bar();
  private:
    int y;
};
class D : public B, public C
{
  public:
    virtual void baz();
  private:
    int z;
};
```

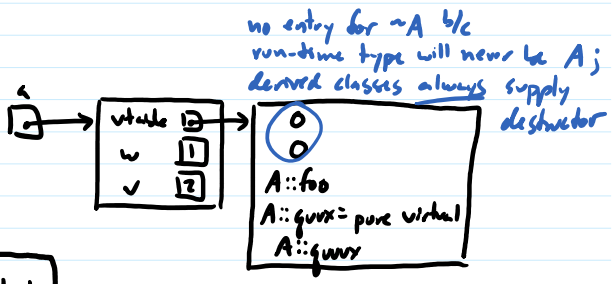


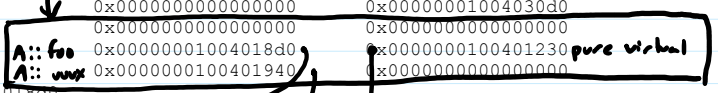
B* d = new D();

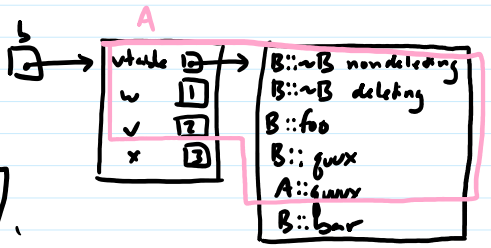
d → x get int from 8 bytes past d
d → w get ptr from 12 bytes past d
go 8 bytes past that ptr to get w

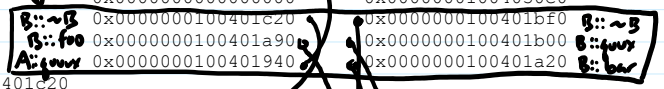
```
#include <iostream>

class A
{
public:
  A() : w(1), v(2) {}
  virtual ~A() {}

  virtual void foo() {
    std::cout << "A::foo " << this << " " << w << std::endl;
  }

  virtual void quux() = 0;

protected:
  virtual void quuux(int a) {
    std::cout << "A::quuux " << this << " " << w << " " << a << std::endl;
  }

private:
  int w, v;
};

class B : public A
{
public:
  B() { x = 3; }

  void foo() {
    std::cout << "B::foo " << this << " " << x << std::endl;
  }

  virtual void bar() {
    std::cout << "B::bar "  << this << " " << x << std::endl;
  }

  void quux() {
    std::cout << "B::quux "  << this << " " << x << std::endl;
  }

private:
  int x;
};

class C : public A
{
public:
  C() { y = 4; }

  virtual void baz() {
    std::cout << "C::baz "  << this << " " << y << std::endl;
    foo();
    quux();
  }
private:
  int y;
};

class D : public B, public C
{
public:
  D() { z = 5; }

  virtual void quux() {
    std::cout << "D::quux "  << this << " " << z << std::endl;
    std::cout << "D::B::A::w = " << *((int *)(((char *)this) + 8)) << std::endl;
  }

private:
  int z;
};

typedef void(*method)(A*, int);

int main(int argc, char **argv)
{
  B *b = new B();
  D *d = new D();

  // static calls
  d->B::foo();
  d->C::foo();

  // polymorphic calls
  ((B*)d)->foo();
  ((C*)d)->foo();
```

```
  // no pointer adjustment
  d->bar();

  // pointer adjustment before calls (equivalent to ((C*)d)->baz())
  d->baz();
  d->quux();

  // error: quuux is protected...
  //d->B::quuux();

  //...but we can get a pointer to it from the vtable
  method *vtable = *((method **)d);
  vtable[4]((B*)d, 99);

  delete d;
}

/*
An instance of A (as seen partway through the construction of an instance of D)
0x00000001004019de      6         A() : w(1), v(2) {}
(gdb) print *this
$2 = {_vptr.A = 0x1004031b0 <vtable for A+16>, w = 1, v = 2}
(gdb) x/4xw this
0x6000394f0: 0x004031b0      0x00000001      0x00000001      0x00000002
(gdb) x/8xg 0x1004031a0
0x1004031a0 <_ZTV1A>:        0x0000000000000000      0x00000001004030d0
0x1004031b0 <_ZTV1A+16>:     0x0000000000000000      0x0000000000000000
0x1004031c0 <_ZTV1A+32>:     0x00000001004018d0      0x0000000100401230
0x1004031d0 <_ZTV1A+48>:     0x0000000100401940      0x0000000000000000
(gdb) x/1i 0x00000001004018d0
   0x1004018d0 <A::foo()>:       push   %rbp
(gdb) x/1i 0x0000000100401230
   0x100401230 <__cxa_pure_virtual>:
                                 jmpq   *0x6ff2(%rip)
(gdb) x/1i 0x0000000100401940
   0x100401940 <A::quuux(int)>:  push   %rbp


An instance of B
(gdb) x/8xw b
0x6000394f0: 0x004031f0      0x00000001      0x00000001      0x00000002
0x600039500: 0x00000003      0x00000006      0x00000053      0x00000000
(gdb) x/8xg 0x1004031e0
0x1004031e0 <_ZTV1B>:        0x0000000000000000      0x00000001004030e0
0x1004031f0 <_ZTV1B+16>:     0x0000000100401c20      0x0000000100401bf0
0x100403200 <_ZTV1B+32>:     0x0000000100401a90      0x0000000100401b00
0x100403210 <_ZTV1B+48>:     0x0000000100401940      0x0000000100401a20
(gdb) x/14i 0x0000000100401c20
   0x100401c20 <B::~B()>:        push   %rbp
   0x100401c21 <B::~B()+1>:      mov    %rsp,%rbp
   0x100401c24 <B::~B()+4>:      sub    $0x20,%rsp
   0x100401c28 <B::~B()+8>:      mov    %rcx,0x10(%rbp)
   0x100401c2c <B::~B()+12>:     lea    0x15bd(%rip),%rcx        # 0x1004031f0 <_ZTV1B+16>
   0x100401c33 <B::~B()+19>:     mov    0x10(%rbp),%rax
   0x100401c37 <B::~B()+23>:     mov    %rdx,(%rax)
   0x100401c3a <B::~B()+26>:     mov    0x10(%rbp),%rax
   0x100401c3e <B::~B()+30>:     mov    %rax,%rcx
   0x100401c41 <B::~B()+33>:     callq  0x100401a00 <A::~A()>
   0x100401c46 <B::~B()+38>:     nop
   0x100401c47 <B::~B()+39>:     add    $0x20,%rsp
   0x100401c4b <B::~B()+43>:     pop    %rbp
   0x100401c4c <B::~B()+44>:     retq
(gdb) x/13i 0x0000000100401bf0
   0x100401bf0 <B::~B()>:        push   %rbp
   0x100401bf1 <B::~B()+1>:      mov    %rsp,%rbp
   0x100401bf4 <B::~B()+4>:      sub    $0x20,%rsp
   0x100401bf8 <B::~B()+8>:      mov    %rcx,0x10(%rbp)
   0x100401bfc <B::~B()+12>:     mov    0x10(%rbp),%rcx
   0x100401c00 <B::~B()+16>:     callq  0x100401c20 <B::~B()>
   0x100401c05 <B::~B()+21>:     mov    $0x18,%edx
   0x100401c0a <B::~B()+26>:     mov    0x10(%rbp),%rcx
   0x100401c0e <B::~B()+30>:     callq  0x100401260 <_ZdlPvm>
   0x100401c13 <B::~B()+35>:     nop
   0x100401c14 <B::~B()+36>:     add    $0x20,%rsp
   0x100401c18 <B::~B()+40>:     pop    %rbp
   0x100401c19 <B::~B()+41>:     retq
(gdb) x/14i 0x100401a00
   0x100401a00 <A::~A()>:        push   %rbp
   0x100401a01 <A::~A()+1>:      mov    %rsp,%rbp
   0x100401a04 <A::~A()+4>:      mov    %rcx,0x10(%rbp)
   0x100401a08 <A::~A()+8>:      lea    0x17a1(%rip),%rax        # 0x1004031b0 <_ZTV1A+16>
   0x100401a0f <A::~A()+15>:     mov    0x10(%rbp),%rcx
   0x100401a13 <A::~A()+19>:     mov    %rdx,(%rax)
   0x100401a16 <A::~A()+22>:     nop
   0x100401a17 <A::~A()+23>:     pop    %rbp
   0x100401a18 <A::~A()+24>:     retq
(gdb) x/1i 0x0000000100401a90
0x100401a90 <B::foo()>:              push   %rbp
(gdb) x/1i 0x0000000100401b00
0x100401b00 <B::quux()>:             push   %rbp
```









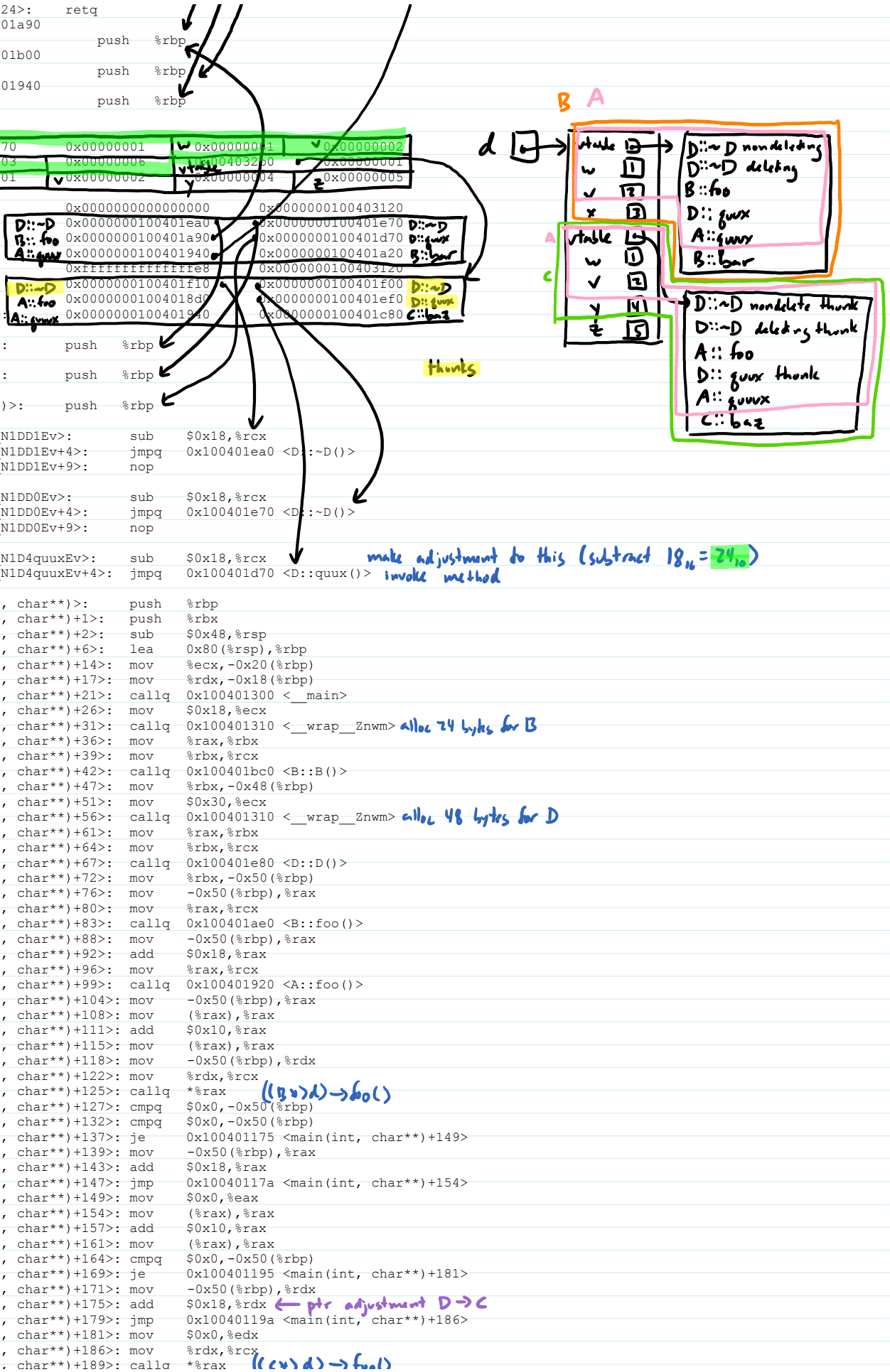
derived class destroyed; set vtable to B vtable

chained to non-deleting A destructor

call non-deleting destructor...

... then delete

```
   0x100401a18 <A::~A()+24>:    retq
(gdb) x/1i 0x0000000100401a90
0x100401a90 <B::foo()>:              push   %rbp
(gdb) x/1i 0x0000000100401b00
0x100401b00 <B::quux()>:             push   %rbp
(gdb) x/1i 0x0000000100401940
0x100401a20 <B::bar()>:              push   %rbp

(gdb) x/12xw d
0x600039510: 0x00403270      0x00000001      0x00000001      0x00000002
0x600039520: 0x00000003      0x00000006      0x00403260      0x00000001
0x600039530: 0x00000001      0x00000002      0x00000004      0x00000005
(gdb) x/16xg 0x100403260
0x100403260 <_ZTV1D>:        0x0000000000000000      0x0000000100403120
0x100403270 <_ZTV1D+16>:     0x0000000100401ea0      0x0000000100401e70
0x100403280 <_ZTV1D+32>:     0x0000000100401a90      0x0000000100401d70
0x100403290 <_ZTV1D+48>:     0x0000000100401940      0x0000000100401a20
0x1004032a0 <_ZTV1D+64>:     0xfffffffffffffff8      0x0000000100403120
0x1004032b0 <_ZTV1D+80>:     0x0000000100401f10      0x0000000100401f00
0x1004032c0 <_ZTV1D+96>:     0x0000000100401840      0x0000000100401ef0
0x1004032d0 <_ZTV1D+112>:    0x0000000100401940      0x0000000100401c80
(gdb) x/1i 0x100401ea0
   0x100401ea0 <D::~D()>:    push   %rbp
(gdb) x/1i 0x100401e70
   0x100401e70 <D::~D()>:    push   %rbp
(gdb) x/1i 0x100401d70
   0x100401d70 <D::quux()>:  push   %rbp
(gdb) x/3i 0x100401f10
   0x100401f10 <_ZThn24_N1DD1Ev>:    sub    $0x18,%rcx
   0x100401f14 <_ZThn24_N1DD1Ev+4>:  jmpq   0x100401ea0 <D::~D()>
   0x100401f19 <_ZThn24_N1DD1Ev+9>:  nop
(gdb) x/3i 0x100401f00
   0x100401f00 <_ZThn24_N1DD0Ev>:    sub    $0x18,%rcx
   0x100401f04 <_ZThn24_N1DD0Ev+4>:  jmpq   0x100401e70 <D::~D()>
   0x100401f09 <_ZThn24_N1DD0Ev+9>:  nop
(gdb) x/2i 0x100401ef0
   0x100401ef0 <_ZThn24_N1D4quuxEv>:    sub    $0x18,%rcx
   0x100401ef4 <_ZThn24_N1D4quuxEv+4>:  jmpq   0x100401d70 <D::quux()>

   0x1004010e0 <main(int, char**)>:     push   %rbp
   0x1004010e1 <main(int, char**)+1>:   push   %rbx
   0x1004010e2 <main(int, char**)+2>:   sub    $0x48,%rsp
   0x1004010e6 <main(int, char**)+6>:   lea    0x80(%rsp),%rbp
   0x1004010ee <main(int, char**)+14>:  mov    %ecx,-0x20(%rbp)
   0x1004010f1 <main(int, char**)+17>:  mov    %rdx,-0x18(%rbp)
   0x1004010f5 <main(int, char**)+21>:  callq  0x100401300 <__main>
   0x1004010fa <main(int, char**)+26>:  mov    $0x18,%ecx
   0x1004010ff <main(int, char**)+31>:  callq  0x100401310 <__wrap__Znwm>
   0x100401104 <main(int, char**)+36>:  mov    %rax,%rbx
   0x100401107 <main(int, char**)+39>:  mov    %rbx,%rcx
   0x10040110a <main(int, char**)+42>:  callq  0x100401bc0 <B::B()>
   0x10040110f <main(int, char**)+47>:  mov    %rbx,-0x48(%rbp)
   0x100401113 <main(int, char**)+51>:  mov    $0x30,%ecx
   0x100401118 <main(int, char**)+56>:  callq  0x100401310 <__wrap__Znwm>
   0x10040111d <main(int, char**)+61>:  mov    %rax,%rbx
   0x100401120 <main(int, char**)+64>:  mov    %rbx,%rcx
   0x100401123 <main(int, char**)+67>:  callq  0x100401e80 <D::D()>
   0x100401128 <main(int, char**)+72>:  mov    %rbx,-0x50(%rbp)
   0x10040112c <main(int, char**)+76>:  mov    -0x50(%rbp),%rax
   0x100401130 <main(int, char**)+80>:  mov    %rax,%rcx
   0x100401133 <main(int, char**)+83>:  callq  0x100401ae0 <B::foo()>
   0x100401138 <main(int, char**)+88>:  mov    -0x50(%rbp),%rax
   0x10040113c <main(int, char**)+92>:  add    $0x18,%rax
   0x100401140 <main(int, char**)+96>:  mov    %rax,%rcx
   0x100401143 <main(int, char**)+99>:  callq  0x100401920 <A::foo()>
   0x100401148 <main(int, char**)+104>: mov    -0x50(%rbp),%rax
   0x10040114c <main(int, char**)+108>: mov    (%rax),%rax
   0x10040114f <main(int, char**)+111>: add    $0x10,%rax
   0x100401153 <main(int, char**)+115>: mov    (%rax),%rax
   0x100401156 <main(int, char**)+118>: mov    -0x50(%rbp),%rdx
   0x10040115a <main(int, char**)+122>: mov    %rdx,%rcx
   0x10040115d <main(int, char**)+125>: callq  *%rax
   0x10040115f <main(int, char**)+127>: cmpq   $0x0,-0x50(%rbp)
   0x100401164 <main(int, char**)+132>: cmpq   $0x0,-0x50(%rbp)
   0x100401169 <main(int, char**)+137>: je     0x100401175 <main(int, char**)+149>
   0x10040116b <main(int, char**)+139>: mov    -0x50(%rbp),%rax
   0x10040116f <main(int, char**)+143>: add    $0x18,%rax
   0x100401173 <main(int, char**)+147>: jmp    0x10040117a <main(int, char**)+154>
   0x100401175 <main(int, char**)+149>: mov    $0x0,%eax
   0x10040117a <main(int, char**)+154>: mov    (%rax),%rax
   0x10040117d <main(int, char**)+157>: add    $0x10,%rax
   0x100401181 <main(int, char**)+161>: mov    (%rax),%rax
   0x100401184 <main(int, char**)+164>: cmpq   $0x0,-0x50(%rbp)
   0x100401189 <main(int, char**)+169>: je     0x100401195 <main(int, char**)+181>
   0x10040118b <main(int, char**)+171>: mov    -0x50(%rbp),%rdx
   0x10040118f <main(int, char**)+175>: add    $0x18,%rdx
   0x100401193 <main(int, char**)+179>: jmp    0x10040119a <main(int, char**)+186>
   0x100401195 <main(int, char**)+181>: mov    $0x0,%edx
   0x10040119a <main(int, char**)+186>: mov    %rdx,%rcx
   0x10040119d <main(int, char**)+189>: callq  *%rax
```



```
   0x100040118f <main(int, char**)+175>: add    $0x18,%rdx ← ptr adjustment D->C
   0x100401193 <main(int, char**)+179>: jmp    0x10040119a <main(int, char**)+186>
   0x100401195 <main(int, char**)+181>: mov    $0x0,%edx
   0x10040119a <main(int, char**)+186>: mov    %rdx,%rcx
   0x10040119d <main(int, char**)+189>: callq  *%rax   ((C*)d)->foo()
   0x10040119f <main(int, char**)+191>: mov    -0x50(%rbp),%rax
   0x1004011a3 <main(int, char**)+195>: mov    (%rax),%rax
   0x1004011a6 <main(int, char**)+198>: add    $0x28,%rax
   0x1004011aa <main(int, char**)+202>: mov    (%rax),%rax
   0x1004011ad <main(int, char**)+205>: mov    -0x50(%rbp),%rdx
   0x1004011b1 <main(int, char**)+209>: mov    %rdx,%rcx
   0x1004011b4 <main(int, char**)+212>: callq  *%rax   d->bar()
   0x1004011b6 <main(int, char**)+214>: mov    -0x50(%rbp),%rax
   0x1004011ba <main(int, char**)+218>: mov    0x18(%rax),%rax
   0x1004011be <main(int, char**)+222>: add    $0x28,%rax
   0x1004011c2 <main(int, char**)+226>: mov    (%rax),%rax
   0x1004011c5 <main(int, char**)+229>: mov    -0x50(%rbp),%rdx
   0x1004011c9 <main(int, char**)+233>: add    $0x18,%rdx ← D->C ptr adjust
   0x1004011cd <main(int, char**)+237>: mov    %rdx,%rcx
   0x1004011d0 <main(int, char**)+240>: callq  *%rax   d->baz()
   0x1004011d2 <main(int, char**)+242>: mov    -0x50(%rbp),%rax
   0x1004011d6 <main(int, char**)+246>: mov    (%rax),%rax
   0x1004011d9 <main(int, char**)+249>: add    $0x18,%rax
   0x1004011dd <main(int, char**)+253>: mov    (%rax),%rax
   0x1004011e0 <main(int, char**)+256>: mov    -0x50(%rbp),%rdx
   0x1004011e4 <main(int, char**)+260>: mov    %rdx,%rcx
   0x1004011e7 <main(int, char**)+263>: callq  *%rax   d->quux()
   0x1004011e9 <main(int, char**)+265>: mov    -0x50(%rbp),%rax
   0x1004011ed <main(int, char**)+269>: mov    (%rax),%rax
   0x1004011f0 <main(int, char**)+272>: mov    %rax,-0x58(%rbp)   vtable = *d (with appropriate casts)
   0x1004011f4 <main(int, char**)+276>: mov    -0x58(%rbp),%rax
   0x1004011f8 <main(int, char**)+280>: add    $0x20,%rax
   0x1004011fc <main(int, char**)+284>: mov    (%rax),%rax
   0x1004011ff <main(int, char**)+287>: mov    -0x50(%rbp),%rcx
   0x100401203 <main(int, char**)+291>: mov    $0x63,%edx
   0x100401208 <main(int, char**)+296>: callq  *%rax   vtable[4](d, 99)
   0x10040120a <main(int, char**)+298>: cmpq   $0x0,-0x50(%rbp)
   0x10040120f <main(int, char**)+303>: je     0x100401228 <main(int, char**)+328>
   0x100401211 <main(int, char**)+305>: mov    -0x50(%rbp),%rax
   0x100401215 <main(int, char**)+309>: mov    (%rax),%rax
   0x100401218 <main(int, char**)+312>: add    $0x8,%rax
   0x10040121c <main(int, char**)+316>: mov    (%rax),%rax   get the deleting destructor from vtable
   0x10040121f <main(int, char**)+319>: mov    -0x50(%rbp),%rdx
   0x100401223 <main(int, char**)+323>: mov    %rdx,%rcx
   0x100401226 <main(int, char**)+326>: callq  *%rax   delete
   0x100401228 <main(int, char**)+328>: mov    $0x0,%eax
   0x10040122d <main(int, char**)+333>: add    $0x48,%rsp
   0x100401231 <main(int, char**)+337>: pop    %rbx
   0x100401232 <main(int, char**)+338>: pop    %rbp
*/
```