virtual1.cpp

```
class ClassA
{
public:
    void sun()         { printf("\tClassA sun\n"); }
    virtual void mon() { printf("\tClassA mon\n"); }
    virtual void tue() { printf("\tClassA tue\n"); }
    virtual void wed() { printf("\tClassA wed\n"); }
};

class ClassB : public ClassA
{
public:
    virtual void mon() { printf("\tClassB mon (%#lx)\n", this); }
};

int main(int argc, void **argv)
{
    void(**vTable)(ssize_t);

    printf("A:\n");
    ClassA * a = new ClassA;
    vTable = *(void(***)(ssize_t))a;
    printf("  vTable[0]: %#lx", vTable[0]); vTable[0](0x1000);
    printf("  vTable[1]: %#lx", vTable[1]); vTable[1](0x1000);
    printf("  vTable[2]: %#lx", vTable[2]); vTable[2](0x1000);

    printf("B:\n");
    ClassB * b = new ClassB;
    vTable = *(void(***)(ssize_t))b;
    printf("  vTable[0]: %#lx", vTable[0]); vTable[0](0x1000);
    printf("  vTable[1]: %#lx", vTable[1]); vTable[1](0x1000);
    printf("  vTable[2]: %#lx", vTable[2]); vTable[2](0x1000);

    return 0;
};
```

```
A:
  vTable[0]: 0x4008a8       ClassA mon
  vTable[1]: 0x400890       ClassA tue
  vTable[2]: 0x400878       ClassA wed
B:
  vTable[0]: 0x4008c0       ClassB mon (0x1000)
  vTable[1]: 0x400890       ClassA tue
  vTable[2]: 0x400878       ClassA wed
```

virtual2.cpp

```
class ClassA
{
public:
    void sun()         { printf("\tClassA sun\n"); }
    virtual void mon() { printf("\tClassA mon\n"); }
    virtual void tue() { printf("\tClassA tue\n"); }
    virtual void wed() { printf("\tClassA wed\n"); }
};

class ClassB : public ClassA
{
public:
    void sun()          { printf("\tClassB sun\n"); }
    virtual void mon()  { printf("\tClassB mon\n"); }
    virtual void thu()  { printf("\tClassB thu\n"); }
};

int main(int argc, void **argv)
{
    void(**vTable)(ssize_t);

    printf("A:\n");
    ClassA * a = new ClassA;
    vTable = *(void(***)(ssize_t))a;
    printf("  vTable[0]: %#lx", vTable[0]); vTable[0](0x1000);
    printf("  vTable[1]: %#lx", vTable[1]); vTable[1](0x1000);
    printf("  vTable[2]: %#lx", vTable[2]); vTable[2](0x1000);
    printf("  vTable[3]: %#lx\n", vTable[3]); // vTable[3](0x1000);

    printf("B:\n");
    ClassB * b = new ClassB;
    vTable = *(void(***)(ssize_t))b;
    printf("  vTable[0]: %#lx", vTable[0]); vTable[0](0x1000);
    printf("  vTable[1]: %#lx", vTable[1]); vTable[1](0x1000);
    printf("  vTable[2]: %#lx", vTable[2]); vTable[2](0x1000);
    printf("  vTable[3]: %#lx", vTable[3]); vTable[3](0x1000);
    printf("\n");

    return 0;
};
```

```
A:
  vTable[0]: 0x400948       ClassA mon
  vTable[1]: 0x400930       ClassA tue
  vTable[2]: 0x400918       ClassA wed
  vTable[3]: 0
B:
  vTable[0]: 0x400978       ClassB mon
  vTable[1]: 0x400930       ClassA tue
  vTable[2]: 0x400918       ClassA wed
  vTable[3]: 0x400960       ClassB thu
```

virtual3.cpp (page 1/3)

```
class ClassA
{
public:
    void sun()         { printf("ClassA sun\n"); }
    virtual void mon() { printf("ClassA mon\n"); }
    virtual void tue() { printf("ClassA tue\n"); }
    virtual void wed() { printf("ClassA wed\n"); }

private:
    char aData[256];
};

class ClassB
{
public:
    void sun()          { printf("ClassB sun\n"); }
    virtual void mon()  { printf("ClassB mon\n"); }
    virtual void tue()  { printf("ClassB tue\n"); }
private:
    int bData;
};

class ClassC : public ClassA, public ClassB
{
public:
    virtual void mon()  { printf("ClassC mon (%#lx)\n", this); }
};

void MyFunctionA(ClassA *a)
{
    printf("a: %#lx\n", a);
}

void MyFunctionB(ClassB *b)
{
    printf("b: %#lx\n", b);
}

void MyFunctionC(ClassC *c)
{
    printf("c: %#lx\n", c);
}


int main(int argc, void **argv)
{
    ClassC * c = new ClassC;
    MyFunctionA(c);
    MyFunctionB(c);
    MyFunctionC(c);

    printf("\nUsing C-style cast (c->a and c->b)\n");
    ClassA * a = (ClassA *)c;
    ClassB * b = (ClassB *)c;
    MyFunctionA(a);
```

virtual3.cpp (page 2/3)

```
    MyFunctionB(b);
    MyFunctionC(c);

    printf("\nUsing C-style cast (c->a and a->b)\n");
    a = (ClassA *)c;
    b = (ClassB *)a;
    MyFunctionA(a);
    MyFunctionB(b);
    MyFunctionC(c);

    printf("\nUsing reinterpret-style cast (c->a and c->b)\n");
    a = reinterpret_cast<ClassA *>(c);
    b = reinterpret_cast<ClassB *>(c);
    MyFunctionA(a);
    MyFunctionB(b);
    MyFunctionC(c);

    printf("\nUsing reinterpret-style cast (c->a and a->b)\n");
    a = reinterpret_cast<ClassA *>(c);
    b = reinterpret_cast<ClassB *>(a);
    MyFunctionA(a);
    MyFunctionB(b);
    MyFunctionC(c);


    printf("\nUsing static-style cast\n");
    a = (ClassA *)c;
    //b = static_cast<ClassB *>(a);
    MyFunctionA(a);
    printf("b: invalid static_cast from type 'ClassA*' to type 'ClassB*'\n");
    //MyFunctionB(b);
    MyFunctionC(c);

    printf("\nUsing dynamic-style cast\n");
    b = dynamic_cast<ClassB *>(c);
    a = dynamic_cast<ClassA *>(b);
    MyFunctionA(a);
    MyFunctionB(b);
    MyFunctionC(c);


    return 0;
};
```

virtual3.out (page 3/3)

```
a: 0x1e3d010
b: 0x1e3d118
c: 0x1e3d010

Using C-style cast (c->a and c->b)
a: 0x1e3d010
b: 0x1e3d118
c: 0x1e3d010

Using C-style cast (c->a and a->b)
a: 0x1e3d010
b: 0x1e3d010
c: 0x1e3d010

Using reinterpret-style cast (c->a and c->b)
a: 0x1e3d010
b: 0x1e3d010
c: 0x1e3d010

Using reinterpret-style cast (c->a and a->b)
a: 0x1e3d010
b: 0x1e3d010
c: 0x1e3d010

Using static-style cast
a: 0x1e3d010
b: invalid static_cast from type 'ClassA*' to type 'ClassB*'
c: 0x1e3d010

Using dynamic-style cast
a: 0x1e3d010
b: 0x1e3d118
c: 0x1e3d010
```

virtual4.cpp (page 1/2)

```
class ClassA
{
public:
    void sun()         { printf("\tClassA sun\n"); }
    virtual void mon() { printf("\tClassA mon\n"); }
    virtual void tue() { printf("\tClassA tue\n"); }
    virtual void wed() { printf("\tClassA wed\n"); }
};

class ClassB
{
public:
    void sun()          { printf("\tClassB sun\n"); }
    virtual void mon()  { printf("\tClassB mon\n"); }
    virtual void tue()  { printf("\tClassB tue\n"); }
};

class ClassC : public ClassA, public ClassB
{
public:
    virtual void mon()  { printf("\tClassC mon (%#lx)\n", this); }
};

int main(int argc, void **argv)
{
    void(**vTable)(ssize_t);
    void(**vTable2)(ssize_t);

    printf("A:\n");
    ClassA * a = new ClassA;
    vTable = *(void(***)(ssize_t))a;
    printf("  vTable[0]: %#lx", vTable[0]); vTable[0](0x1000);
    printf("  vTable[1]: %#lx", vTable[1]); vTable[1](0x1000);
    printf("  vTable[2]: %#lx", vTable[2]); vTable[2](0x1000);

    printf("C:\n");
    ClassC * c = new ClassC;
    vTable = *(void(***)(ssize_t))c;
    vTable2 = ((void(***)(ssize_t))c)[1];
    printf("  vTable[0]: %#lx", vTable[0]); vTable[0](0x1000);
    printf("  vTable[1]: %#lx", vTable[1]); vTable[1](0x1000);
    printf("  vTable[2]: %#lx", vTable[2]); vTable[2](0x1000);

    printf("\n");
    printf("  vTable2[0]: %#lx", vTable2[0]); vTable2[0](0x1000);
    printf("  vTable2[1]: %#lx", vTable2[1]); vTable2[1](0x1000);
    printf("  vTable2[2]: %#lx\n", vTable2[2]); // vTable2[2](0x1000);

    return 0;
};
```

virtual4.out (page 2/2)

```
A:
  vTable[0]: 0x4009ac       ClassA mon
  vTable[1]: 0x400994       ClassA tue
  vTable[2]: 0x40097c       ClassA wed
C:
  vTable[0]: 0x4009fa       ClassC mon (0x1000)
  vTable[1]: 0x400994       ClassA tue
  vTable[2]: 0x40097c       ClassA wed

  vTable2[0]: 0x4009f4      ClassC mon (0xff8)
  vTable2[1]: 0x4009c4      ClassB tue
  vTable2[2]: 0



0x00000000004009f4 in non-virtual thunk to ClassC::mon() ()
(gdb) disass
Dump of assembler code for function _ZThn8_N6ClassC3monEv:
0x00000000004009f4 <_ZThn8_N6ClassC3monEv+0>:   add    $0xfffffffffffffff8,%rdi
0x00000000004009f8 <_ZThn8_N6ClassC3monEv+4>:   jmp    0x4009fa <_ZN6ClassC3monEv>
End of assembler dump.
```