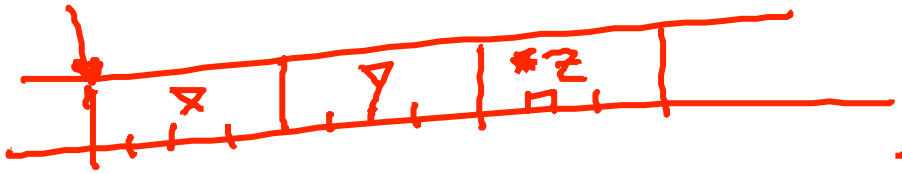
Classes (Object Orientation)

- **In Java/C++, you can define classes.**
  - For the most part, objects are just like structs
  - The main difference is that objects store their type

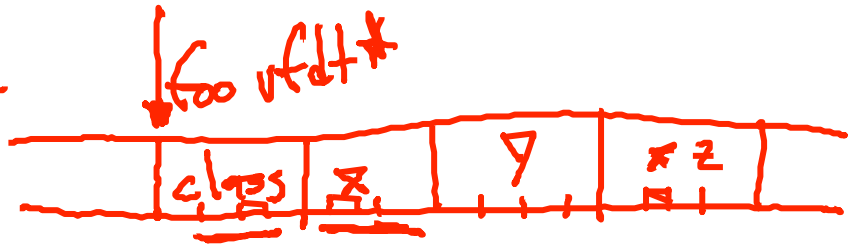
in C

```
struct foo {
    int x;
    int y;
    char *z;
};
```



in C++

```
class foo {
    int x;
    int y;
    char *z;
};
```



Inheritance

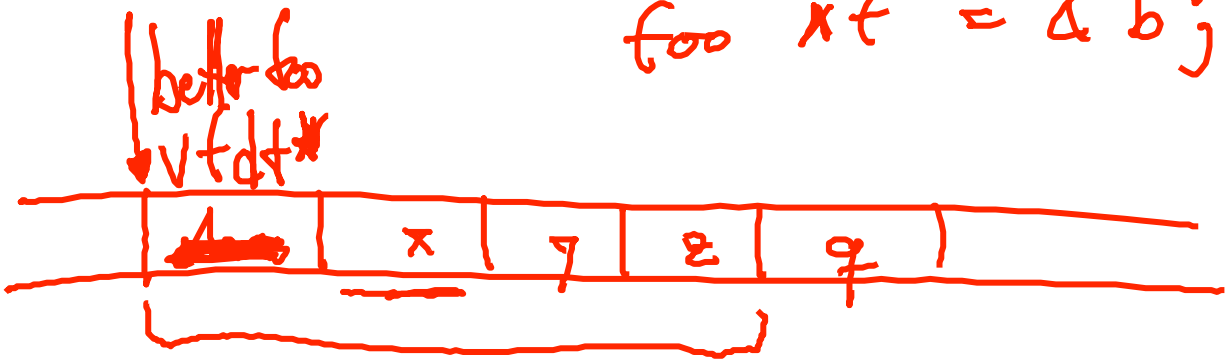
- In Java/C++, classes can "inherit" from other classes.

in C++

```
class foo {
    int x;
    int y;
    char *z;
};
```

in C++

```
class betterfoo : public foo {
    int q;
};
```



Virtual Functions (Polymorphism)

- In Java/C++, classes can redefine methods define by parents

```
class foo {
    ...
   void other_fun() {};
   int func() {...};
};
```

```
class betterfoo : ... {
    ...
   int func() {...};
};
```

betterfoo bj;
foo *f = &bj;
f->func(); // um

Virtual Functions (cont.)

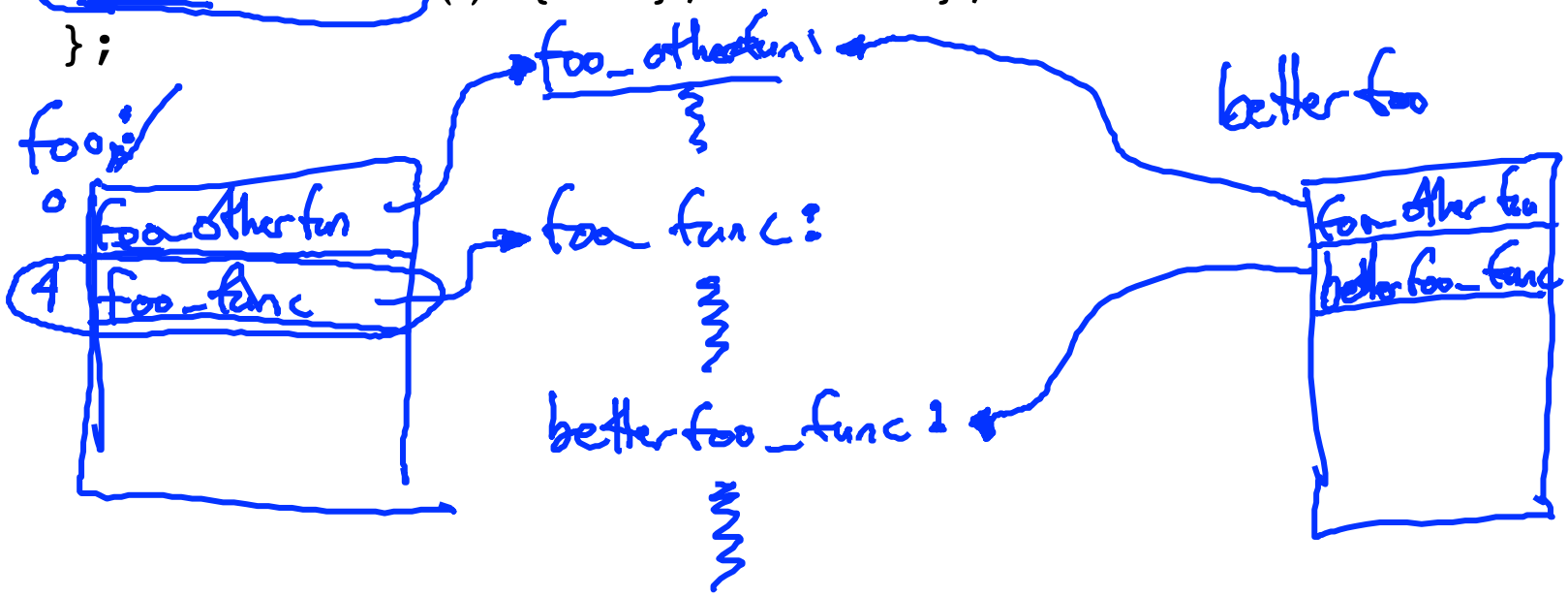
- **How do we know which should be executed?**

```
foo *f = (foo *)new betterfoo(...);
f->func();
```

Virtual Functions (cont.)

- **Virtual Function Dispatch Tables:**

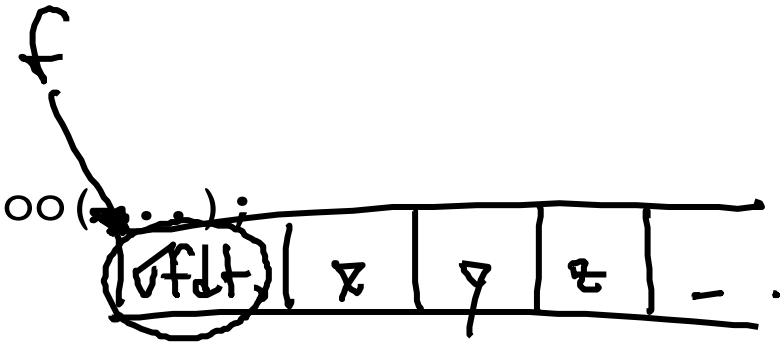
```
class foo {                          class betterfoo : ... {
    ...                                  ...
    void other_fun() {};                 int func() {...};
    int func() {...};                };
};
```



Virtual Functions (cont.)

- **Virtual function dispatch:**

```
foo *f = (foo *)new betterfoo(...);
f->func();
```




```
        $a0 = f
lw      $t0, 0($a0)     # t0 = vfdt*
lw      $t1, 4($t0)     # & func for that type
jalr    $t1             # implement call func();
```