ECE462
10/13/2010

Overloading → compiler matching type/class

↑ function with same name different argument types

```
f(int *) {...}          //1
f(string s) {...}       //2
f(Person *p) {...}      //3
f(student *s) {...}     //4
```

```
class Person {          virtual
public: f1() {...}      //a
};
class Student : public Person {
public: f1() {...}      //b
};

f2(Person *p) {
  p->f1();              //c
}
f2(Student *s) {
  s->f1();              //d
}
```

0.load 0.ride

// overriding at run-time

```
Person *p1;             f2(p1); //c->a
Person *p2;             f2(p2); //c->b
Student *s1;            f2(s1); //d->b
p1 = new Person...
p2 = new Student...
s1 = new Student...
```

if f1() not defined in Person,
f2(p1) & f2(p2) will fail
f2(s1) calls b

~~// f2(Person)~~

if f2(student) is removed
// c->A
// c->B
// c->B

```
f(5); //calls 1
f("ece"); //calls 2
Person *p = new Person
f(p); //calls 3

if (...) {
  p = new Person...
}
else {
  p = new Student...
}
f(p); //3
      // overloading determined
      // by compiler

Person *p = new Student()
  // still calls 3

Student *s = new Student
f(s); // calls 4

if f4 is erased
f(s); // calls 3

if f4 exists, f3 erased:
f(p); //error
f(s); //calls 4
```