**class** Vec {

**private** Object[] elems = **new** Object[16];

**private** int end = 0;

**public** void add**(**Object e**) {**

**if(**end == elems.length**) {** ... **}**

elems[end] = e;

end=end+1;

**}**

**public Object get(int index) {**

**if(index >= end) { throw ... }**

**else {**

**return elems[index];**

**}**

**}**

**}**

Vec v = **new** Vec**()**;

v.add**(new** Cat**())**;

Cat c = (Cat)v.get(0);

How can we say v is a Vec of Cats?

**Java Generics**

* History

– Introduced in Java1.5

– Similar to C++ templates, but actually quite different as well!

• Before Java generics:

– **Can only say things like: ‘v’ is a Vector of Objects**

– Then,canputanyObjectinto‘v’withoutrestriction

– WithaVectorofjustCats,havetocastObjectstoCats

• WithJavaGenerics:

– **Can say things like: ‘v’ is a Vector of Cats**

– Then,canonlyputCatsinto‘v’

– And,canonlygetCatsoutof‘v’–nocastingrequired!