

Java Arrays and Generics

CSCI 334
Stephen Freund

Parametric Polymorphism & Type Erasure

```

class Stack<T> {
    public void push(T val)
    public T pop()
}

Stack<String> strs =
    new Stack<String>();

Stack<Point> pts =
    new Stack<Point>();

strs.push("moo");
pts.push(new Point(2,3));
String s = strs.pop();

class Stack {
    public void push(Object val)
    public Object pop()
}

Stack strs =
    new Stack();

Stack pts =
    new Stack();

strs.push("moo");
pts.push(new Point(2,3));
String s = (String)strs.pop();

Types guarantee strs will only
contain Strings, and pts will
only contain Points
    
```

Stack

```

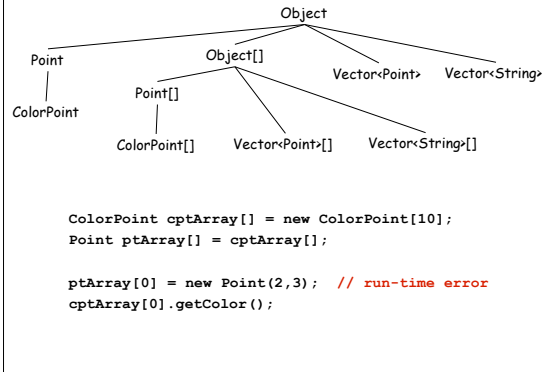
class Stack<T> {
    private T elems[] = new T[100];
    private int size = 0;

    public void push(T val) {
        elems[size++] = val;
    }

    public T pop() {
        T o = elems[size-1];
        size--;
        return o;
    }
}

moo:~] javac Stack.java
Stack.java:4: generic array creation
private T elems[] = new T[100];
                ^
1 error
    
```

Java Subtyping



Code

```

Vector<String>[] strVecArray =
    new Vector<String>[4];

Object[] objArray = strVec;

Vector<Point> ptVec =
    new Vector<Point>();

ptVec.add(new Point(2,3));

objArray[0] = ptVec;

String s =
    strVecArray[0].get(0);

s.length();
    
```

Code

```

Vector<String>[] strVecArray =
    new Vector<String>[4];

Object[] objArray = strVec;

Vector<Point> ptVec =
    new Vector<Point>();

ptVec.add(new Point(2,3));

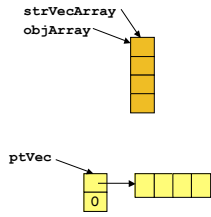
objArray[0] = ptVec;

String s =
    strVecArray[0].get(0);

s.length();
    
```

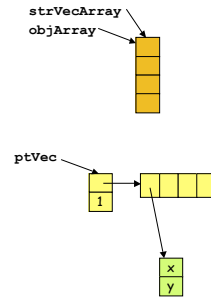
Code

```
Vector<String>[] strVecArray =  
    new Vector<String>[4];  
  
Object[] objArray = strVec;  
  
Vector<Point> ptVec =  
    new Vector<Point>();  
ptVec.add(new Point(2,3));  
  
objArray[0] = ptVec;  
  
String s =  
    strVecArray[0].get(0);  
  
s.length();
```



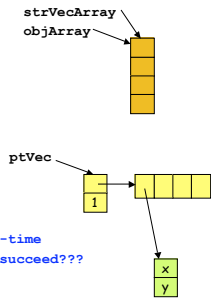
Code

```
Vector<String>[] strVecArray =  
    new Vector<String>[4];  
  
Object[] objArray = strVec;  
  
Vector<Point> ptVec =  
    new Vector<Point>();  
ptVec.add(new Point(2,3));  
  
objArray[0] = ptVec;  
  
String s =  
    strVecArray[0].get(0);  
  
s.length();
```



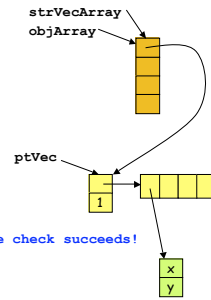
Code

```
Vector<String>[] strVecArray =  
    new Vector<String>[4];  
  
Object[] objArray = strVec;  
  
Vector<Point> ptVec =  
    new Vector<Point>();  
ptVec.add(new Point(2,3));  
  
objArray[0] = ptVec; // will run-time  
                    // check succeed???  
  
String s =  
    strVecArray[0].get(0);  
  
s.length();
```



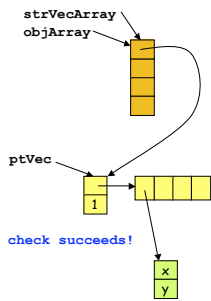
Code

```
Vector<String>[] strVecArray =  
    new Vector<String>[4];  
  
Object[] objArray = strVec;  
  
Vector<Point> ptVec =  
    new Vector<Point>();  
ptVec.add(new Point(2,3));  
  
objArray[0] = ptVec; // run-time check succeeds!  
  
String s =  
    strVecArray[0].get(0);  
  
s.length();
```



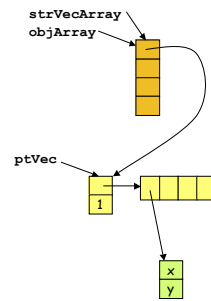
Code (Types Erased)

```
Vector[] strVecArray =  
    new Vector[4];  
  
Object[] objArray = strVec;  
  
Vector ptVec =  
    new Vector();  
ptVec.add(new Point(2,3));  
  
objArray[0] = ptVec; // run-time check succeeds!  
  
String s =  
    strVecArray[0].get(0);  
  
s.length();
```



Code

```
Vector<String>[] strVecArray =  
    new Vector<String>[4];  
  
Object[] objArray = strVec;  
  
Vector<Point> ptVec =  
    new Vector<Point>();  
ptVec.add(new Point(2,3));  
  
objArray[0] = ptVec;  
  
String s =  
    strVecArray[0].get(0);  
  
s.length(); // ERROR!!!
```



So How Do You Write Stack???

```
class Stack<T> {  
  
    private T elems[] = new T[100]; // won't compile  
  
    private int size = 0;  
  
    public void push(T val) {  
        elems[size++] = val;  
    }  
  
    public T pop() {  
        T o = elems[size-1];  
        size--;  
        return o;  
    }  
}
```

So How Do You Write Stack???

```
class Stack<T> {  
  
    private Object elems[] = new Object[100];  
  
    private int size = 0;  
  
    public void push(T val) {  
        elems[size++] = val;  
    }  
  
    public T pop() {  
        T o = (T)elems[size-1];  
        size--;  
        return o;  
    }  
}
```