Memory and C++

Eric Roberts
CS 106B
April 22, 2009

Data Types in C++

Before we can talk about objects in C++, it is useful to review the more traditional data types that C++ inherits from C:

- Atomic types:
  - **`short`**, **`int`**, **`long`**, and their **`unsigned`** variants
  - **`float`**, **`double`**, and **`long double`**
  - **`char`**
  - **`bool`**
- Enumerated types defined using the **`enum`** keyword
- Structure types defined using the **`struct`** keyword
- Arrays of some base type
- Pointers to a target type

Simple Arrays in C++

- We haven't actually used arrays in their low-level form this quarter, because the **`Vector`** class is so much better.
- From the client perspective, an array is like a brain-damaged form of **`Vector`** with the following differences:
  - The only operation is selection using `[]`
  - Array selection does not check that the index is in range
  - The length of an array is fixed at the time it is created
  - Arrays don't store their length, so programs that use them must pass an extra integer value that represents the ***effective size***
- Array variables are declared using the following syntax:

```
type name[n];
```

where *type* is the element type, *name* is the array name, and *n* is a constant integer expression indicating the length.