Strings and Collections

Kate Gregory

www.gregcons.com/kateblog

pluralsight hardcore developer training

Objects and Classes

- **C++ is an object oriented language**
  - C++ apps are not just made of functions, but of classes and objects too
- **A class defines the idea of an object**
  - What data it can hold
  - What functions it can be asked to perform (usually on that data)
  - Example: Date
- **An object is an instance of a class**
  - Example: May 1st, 1990 or Dec 3rd, 2017
- **Functions inside a class are called *member functions***
- **The kind of functions shown earlier are called *free functions* or *nonmember functions***
- **C++ uses plenty of both**

Strings

- C++ has a very useful string class in the `std` namespace

```
#include <string>
```

- Can compare, combine and manipulate strings
- Also search for substrings, make replacements, …
- Makes string feel like a built in type
- For Unicode, use `wstring`

String manipulation

- **Operators:**
  - To combine two strings: + +=
  - To test two strings: == < > !=
  - The cout >> operator and cin << operator both work perfectly with strings
- **Member functions:**
  - length
  - substr
  - find
- **And more…**

Exercise

- **Write a program that asks the user for two words and tells them which is longer**
- **Hints:**
  - Use the code from Guess My Number as a reference
  - This app can run until the user says to stop, or just once: your choice
- **Once your app is working, try entering a phrase when you’re prompted and see what happens**

Collections

- **Many programs need to work with a number of similar items**
  - The people in a department
  - The items in an order
  - The transactions in an account
- **The Standard Library provides classes that are ready to use**
- **Simplest and best first choice: `vector`**
  - Holds a number of values, all of the same type
  - Size does not need to be known in advance
  - Easy to access a specific item or all of them

More on `vector`

- **To add an item to the vector:**
  - push_back()
  - insert() – moves items around, use only if you need it
  - Type of the item added must match type used when declaring the vector
- **To access all the elements of the vector:**
  - for loop and operator []
  - Range based for
  - Iterators begin() and end(), operator ++ for iterator, * for iterator
- **Free functions work on vector and other collections too**
  - count(), sort() and many more

- **Bonus tidbit: `string` is a collection (of characters) too**

Behind more curtains

- **In C++, operators are just functions**
  - strange names, no ()
- **You've seen many operators in this module**
  - +, += == for string
  - [] for vector
  - << for cout, >> for cin
- **Operator overloading gives an intuitive way to use objects**
  - They feel like built in types
- **Templates are a powerful way to write a library**
  - Work on any type, without giving up type safety
  - Work on both built in and user defined types
    - int, bool, double, string, Employee, OrderItem, …
    - Operator overloads are a big part of that

Summary

- **The string class is powerful and useful**
  - Intuitive operator overloads
  - Member functions
  - Works with some free functions in the standard library as well
- **The Standard Library includes classes to represent a collection of anything**
  - vector is the most generally useful collection
  - There are free functions to work with vector and other collections
- **The template mechanism in C++ allows us to generalize over types without losing type safety**
  - You write less code
  - Programs have less bugs
- **Operator overloading is extremely powerful**