

Templates

Wagner Truppel
Lecturer, Dept. of Computer
Science & Engineering
UC Riverside

wagner@cs.ucr.edu
<http://www.cs.ucr.edu/~wagner>

<http://www.cs.ucr.edu/cs12>

©2003 WL Truppel CS 12: Intro. Computer Science II • Lecture 12 1

C++ Templates

- Suppose you want to find the maximum of two **integer** values
- You could write a function for that:

```
int max(int a, int b)  
{ if (a > b) return a;  
  else return b; }
```
- Suppose now that you want to find the maximum of two **float** values
- You could write another function for that:

```
float max(float a, float b)  
{ if (a > b) return a;  
  else return b; }
```


©2003 WL Truppel CS 12: Intro. Computer Science II • Lecture 12 2

C++ Templates

- Suppose next that you want to find the maximum of two **double** values
- You could write a function for that too:

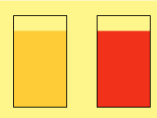
```
double max(double a, double b)  
{ if (a > b) return a;  
  else return b; }
```
- What if you want to find the maximum of two **Speed** values?
- You could also write a function for that:

```
Speed max(Speed & a, Speed & b)  
{ if (a > b) return a;  
  else return b; }
```




©2003 WL Truppel CS 12: Intro. Computer Science II • Lecture 12 3

A simple puzzle...

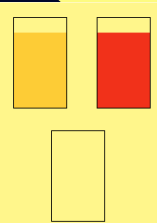


- Say you have a glass of beer and another glass, of red wine.
- Now suppose that for some reason, you want to swap the contents of the two glasses.
- How do you do it?



©2003 WL Truppel CS 12: Intro. Computer Science II • Lecture 12 4

A simple puzzle...



- Say you have a glass of beer and another glass, of red wine.
- Now suppose that for some reason, you want to swap the contents of the two glasses.
- How do you do it?
- You need **another** glass...
- Does it matter that the glasses had beer and wine?

©2003 WL Truppel CS 12: Intro. Computer Science II • Lecture 12 5

Another puzzle...

- How do you swap two *integers* ?
- The integers are stored in two variables
- But you need a **third** one...

```
void swap(int & var1, int & var2)
{
    int temp = var1;
    var1 = var2;
    var2 = temp;
}
```

- Does it matter that you have two integers as opposed to two values of another type?

©2003 WL Truppel CS 12: Intro. Computer Science II • Lecture 12 6

A general swap function

- We'd like to be able to say:
 - ◆ Give me two values of some type **T** and I'll swap them with this function


```
void swap(T & var1, T & var2)
{
    T temp = var1;
    var1 = var2;
    var2 = temp;
}
```

©2003 WL Truppel CS 12: Intro. Computer Science II • Lecture 12 7

No problem, right ?

- We could simply write this:


```
void swap(T & var1, T & var2)
{
    T temp = var1;
    var1 = var2;
    var2 = temp;
}
```
- It doesn't work !
 - ◆ because **T** would be interpreted as the name of some class we defined, rather than as a **parameter**



©2003 WL Truppel CS 12: Intro. Computer Science II • Lecture 12 8

No problem, right ?

Function templates

- We need to tell the compiler that **T** is a **parameter** and not an actual type. How?


```
template<class T>
void swap(T & var1, T & var2)
{
    T temp = var1;
    var1 = var2;
    var2 = temp;
}
```
- Despite the keyword **class**, you can now use this function for **any type that supports the assignment operator**.
- `swap(int1, int2); // swap int variables`
- `swap(char1, char2); // swap char variables`
- `swap(foo1, foo2); // swap Foo variables`

©2003 WL Truppel CS 12: Intro. Computer Science II • Lecture 12 9

Class templates

- We've seen **function templates** (also referred to as template functions)
- We can also have **class templates** (never referred to as template classes)
- Same basic idea, same basic syntax

©2003 WL Truppel CS 12: Intro. Computer Science II • Lecture 12 10

Class templates

```
template<class T>
class Pair
{
    private:
        T first;
        T second;
    public:
        Pair(T first, T sec);
        T getFirst() const;
}

template<class T>
Pair<T>::Pair(T first, T sec)
{ first = first; second = sec; }
```

- Inside the class, just use **T** as if it was some specific type (class or not)
- Member functions are then template functions

©2003 WL Truppel CS 12: Intro. Computer Science II • Lecture 12 11

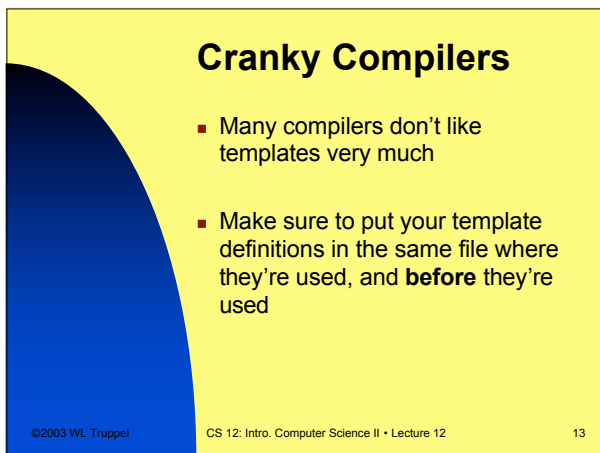
Class templates

```
template<class T>
class Pair
{
    private:
        T first;
        T second;
    public:
        Pair(T first, T sec);
        T getFirst() const;
}

Pair<int> point(2, 7); // a two dimensional point
Pair<Lens> glasses(Lens(), Lens()); // a pair of lenses
```

- How to use it? Just as before, but now you need to specify what type to use:
- `Pair<int> point(2, 7);` // a two dimensional point
- `Pair<Lens> glasses(Lens(), Lens());` // a pair of lenses

©2003 WL Truppel CS 12: Intro. Computer Science II • Lecture 12 12



Cranky Compilers

- Many compilers don't like templates very much
- Make sure to put your template definitions in the same file where they're used, and **before** they're used

©2003 WL Truppel CS 12: Intro. Computer Science II • Lecture 12 13
