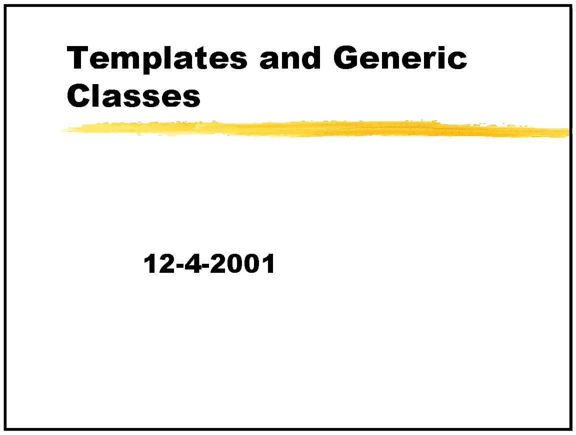
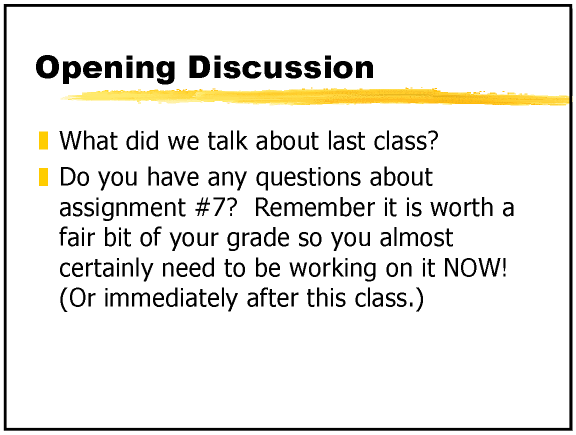


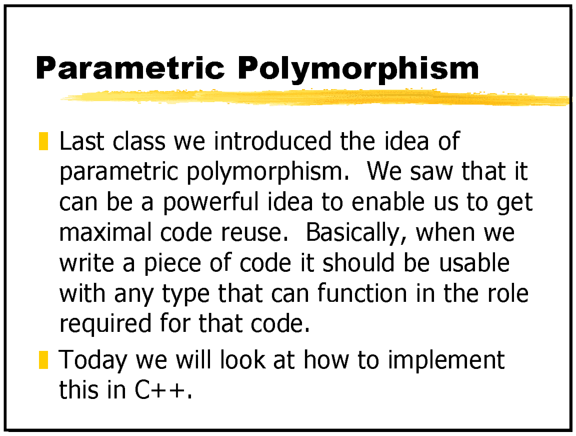
Templates and Generic Classes

12-4-2001



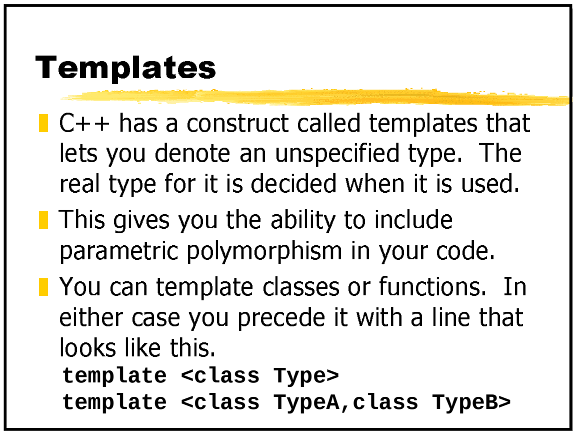
Opening Discussion

- What did we talk about last class?
- Do you have any questions about assignment #7? Remember it is worth a fair bit of your grade so you almost certainly need to be working on it NOW! (Or immediately after this class.)



Parametric Polymorphism

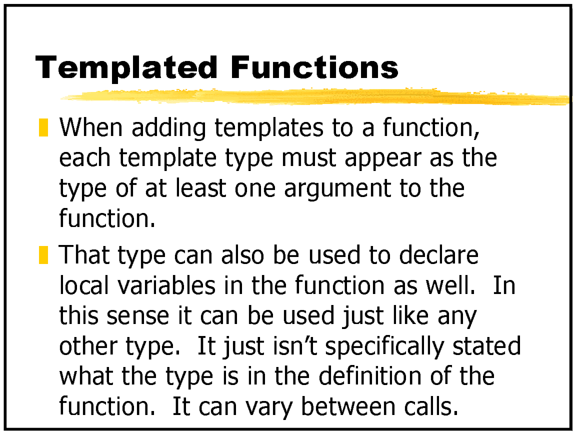
- Last class we introduced the idea of parametric polymorphism. We saw that it can be a powerful idea to enable us to get maximal code reuse. Basically, when we write a piece of code it should be usable with any type that can function in the role required for that code.
- Today we will look at how to implement this in C++.



Templates

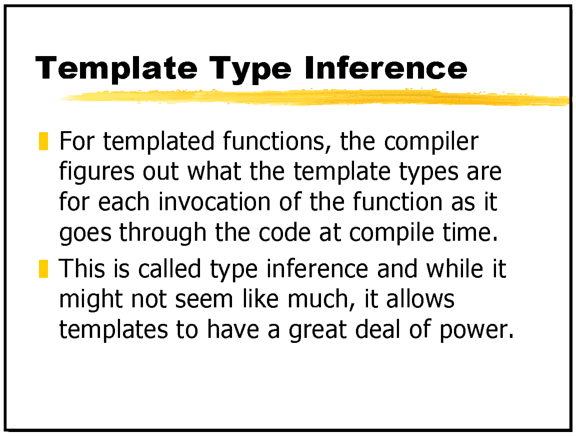
- C++ has a construct called templates that lets you denote an unspecified type. The real type for it is decided when it is used.
- This gives you the ability to include parametric polymorphism in your code.
- You can template classes or functions. In either case you precede it with a line that looks like this.

```
template <class Type>
template <class TypeA,class TypeB>
```



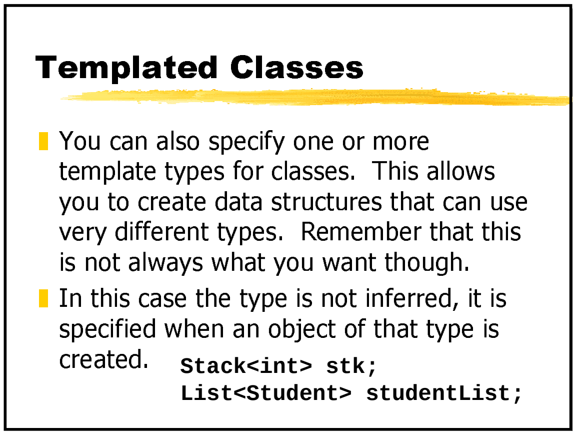
Templated Functions

- When adding templates to a function, each template type must appear as the type of at least one argument to the function.
- That type can also be used to declare local variables in the function as well. In this sense it can be used just like any other type. It just isn't specifically stated what the type is in the definition of the function. It can vary between calls.



Template Type Inference

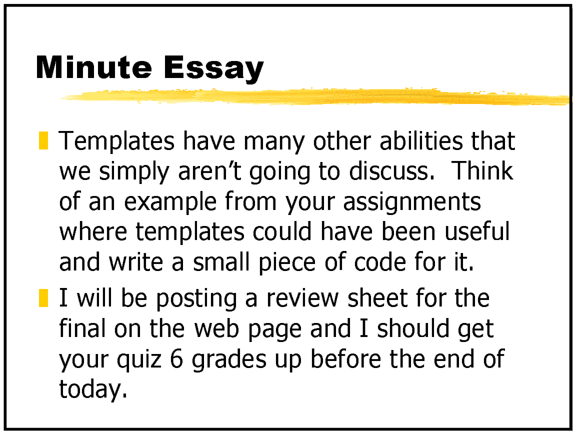
- For templated functions, the compiler figures out what the template types are for each invocation of the function as it goes through the code at compile time.
- This is called type inference and while it might not seem like much, it allows templates to have a great deal of power.



Templated Classes

- You can also specify one or more template types for classes. This allows you to create data structures that can use very different types. Remember that this is not always what you want though.
- In this case the type is not inferred, it is specified when an object of that type is created.

```
Stack<int> stk;
List<Student> studentList;
```



Minute Essay

- Templates have many other abilities that we simply aren't going to discuss. Think of an example from your assignments where templates could have been useful and write a small piece of code for it.
- I will be posting a review sheet for the final on the web page and I should get your quiz 6 grades up before the end of today.