

CS11600: Introduction to Computer Programming (C++)

Lecture 13

Svetlozar Nestorov
University of Chicago

Outline

- Template classes
- Example: generic Stack class
- Static members
- Constant parameters
- Example: a generic List of arrays
- Standard Template Library (STL)

2/7/2003

Svetlozar Nestorov, CS 116: Intro to Programming II

2

Template Classes

- Generic, type-independent classes.
- Common use as container classes:
 - list, set, bag, queue
- Basic syntax:

```
template <class T>
class MyClass { ... };
```
- Template may involve multiple classes.

2/7/2003

Svetlozar Nestorov, CS 116: Intro to Programming II

3

Instantiation

- Fill in types and use as a regular class:

```
MyClass<int> obj1;
MyClass<char*> obj2;
MyClass<double> scores[10];
MyClass<int> *n = new MyClass<int>;
```
- Example: generic Stack class.

2/7/2003

Svetlozar Nestorov, CS 116: Intro to Programming II

4

Static Data Members

- One static data member per *class instantiation*, not per class.
- Initialization syntax:

```
template <class T>
Type MyClass<T>::name = value;
```
- A different initialization for specific type:

```
Type MyClass<int>::name = value;
```

2/7/2003

Svetlozar Nestorov, CS 116: Intro to Programming II

5

Constant Parameters

- In addition to classes, template classes may depend on a constant parameter.

```
template <class T, Type val>
class MyClass { ... };
```
- Example: Stack with max depth:

```
template <class T, int maxDepth>
class Stack { ... };
```

2/7/2003

Svetlozar Nestorov, CS 116: Intro to Programming II

6

Standard Template Library

- Generic container classes:
`vector`, `list`, `deque`, `set`, `multiset`, `map`, `multimap`,
`stack`, `queue`, `priority_queue` (last 3 are adaptors).
- Iterators to navigate the container classes:
`iterator`, `reverse_iterator`, and `const_iterator`,
`const_reverse_iterator`
- No iterators for adaptors.
- Generic algorithms: