# STL For Beginners

Dr. Ganesh R. Baliga

Computer Science Department

# Introduction

- STL: Standard Template Library
- Incorporated into the ANSI/ISO standard for the C++ language
- Collection of classes and functions

# Collections

- STL container classes: deque, vector, list, queue, set, stack, map, multiset, multimap, priority_queue.
- Examples
  - Customers lined at a store's checkout: queue (of customers)
  - Words present in a dictionary: set (of strings)

```
// Mutators; to change the private attributes

  void setNumerator ( int value );
  void setDenominator ( int value );

  private:
  int n_, d_;  // privately held attributes
};

// returns the sum of two Rationals
Rational operator + ( const Rational & f, const Rational & s );

// > operator for Rationals
bool operator > (cons 3-2. cont Ratio8(l & s ); )]TJ..0007 -2.352 TD04 Tc-0
```

- Overloading the operator > for the class Rational

```
// Comparing two Rational numbers
bool operator > (const Rational & f,     // first rational
                 const Rational & s )  // second rational
{
    bool result;
```

## Background: Templates

```
template <class T>
void swap ( T & first, T & second ) {
    T temp = first;
    first = second;
    second = temp;
}
```

- Now can use it to swap two ints, two Rational objects, etc.
- C++ class templates allow creation of generic classes.

# C++ Class Templates

```
template <class A, class B>
class pair {  // Useful standard STL class
public:
  A first;
  B second;
  pair ( A a, B b) : first ( a ), second ( b ) { // Constructor
  }
};
```

Now you can declare:

`pair<string,int>`

## STL Container: vector

Common Usage

Need to #include <queue>

# STL Container: stack

We are implementing the Back button on a web browser

```
#include <stack>
…
stack<URL> urlStack; // Declare stack of URL objects
…
// When the user goes to a new URL…
// One more URL on the stack
urlStack.push ( newURL );
…
// When the user hits the back button …
// Removes the top of the stack
urlStack.pop ( );
URL currentURL = urlStack.top ( ); // And now, go there!
```

## Aggregate Computations

- Need to process the items stored in a container.
- For a vector, this can be done as follows:

```
vector<int> vec;
…
for (int i = 0; i < vec.size( ) ; i++)
    ProcessIt ( vec[ i ] );
```

- Problem: Not all contai5.1s provide indexed access to their items.

# STL Iterators

- Finding the sum of student grades

```
vector<double> grades;
…
vector<double>::iterator it; // Declare an iterator object
for ( it = grades.begin ( );  // Initialize iterator to point to zeroth item
      it != grades.end ( );   // Loop as long as not at the end
      it++)                   // Advance the iterator to the next item
```

## STL Iterators

- Iterators can be used to modify a container at the position that they are pointing to.
  - insert ( iterator, item ) : inserts the item at the given iterator position
  - erase ( iterator ): removes the item at the given e[illegible]

## STL Container: map

- Simple application: Phone book

Need to #include <map>

# STL Container: map

- Another example: Creating a text book index

Two possibilities for storing the index.

```
map<string, set<int> > index;
multimap<string,int> index;
```

# STL Algorithms

- Example: To print a phone book
- Recall that a phone book is declared as:

```
map<string,string> phoneBook;
…
// Function printEntry outputs one entry in the phone book
void printEntry ( const
```

## STL Algorithms

# References