Templates

Templates

- One of the most powerful features of C++ is templates

- There are two types of templates:
  - Class templates
  - Function templates

Function Templates

- How would you write a program that contains functions to return the largest of three arguments?

```
int num1, num2, num3;
cout << "Enter three integers: ";
cin >> num1 >> num2 >> num3;
cout << "The largest number is: " << largest(num1, num2, num3);

double num4, num5, num6;
cout << "Enter three doubles: ";
cin >> num4 >> num5 >> num6;
cout << "The largest number is: " << largest(num4, num5, num6);

char c1, c2, c3;
cout << "Enter three characters: ";
cin >> c1 >> c2 >> c3;
cout << "The largest character is: " << largest(c1, c2, c3);
```

Function Templates

- All three functions that we have written do exactly the same thing

- The only differing thing is the data type of the arguments being used

- We can use templates that will allow the function to be called with different arguments

Function Templates

```
template< class T >
T largest(T var1, T var2, T var3)
{
  T temp = var1;
  if(var2 > temp)
  {
    temp = var2;
  }
  if(var3 > temp)
  {
    temp = var3;
  }
  return temp;
}
```

Your Turn

- Write a generic function that searches an array for a specified value. If the value is found, return the index of where it was found; otherwise, return -1

- Write a main function that tests the generic function search for both integers and reals in the same program

Problem

- Write a template function **swap** that will swap the values of two variables

- How would you test this template function?

- Discuss in detail whether the following code is legal or not. If not, why not. If so, what is actually going on for this to work.

```
string s1 = "ab", s2 = "cd";

swap (s1, s2);

cout << s1 << " " << s2 << endl;
```

3