CS2141 – Software Development using C/C++

References

What is a reference?

- A *reference* is an alias
- A reference is different from a pointer:
  - A reference cannot be null
  - Once established, a reference cannot be changed
  - It does not need to be dereferenced
  - All operators operate on the referenced value
- A reference could be thought of as a permanently dereferenced pointer

Using a Reference

- A reference is declared using the ampersand:

```
int i = 8;
int & j = i;  // j is now an alias for i
j++;          // i is now 9
i += 5;       // i is now 14, as is j
j = j * 3;    // i and j are now 42


TestClass A = new TestClass();
int & cur_val = A.value;
cur_val = 45; // Changes A's value
```

Passing by Reference

- The most common use of a reference is to pass by reference to a function, allowing the function to change the actual value:

```
void passExample( int & i )
{
   i++;
   i = i + 1;
}

int j = 5;
passExample( j );
cout << j << endl;
```

Passing by Constant Reference

- A constant reference is often used in place of passing by value when dealing with large objects:

```
void passExample( const BigObject & b )
{
   ...
}
```

- The result is the same as passing by value since the original object cannot be modified, but the code is more efficient since the large object does not get copied.

Returning a Reference

- A reference can be used as the target of an assignment, so sometimes functions return a reference instead of a value:

```
int values[100];

int & index( int i ) {
   return values[i + 2];
}

...

index(15) = 35;
```

The code will change the content of **`values[17]`** to 35.