Note From Class

Thursday, January 28, 2016 4:00 PM

Student Questions:

1. What Should Happen When You Try:
   // test positively
2. `assert( isWellFormedCommandString( "4AE" ) );`
3. `/// test negatively`
   `assert( !isWellFormedCommandString( "4AE" ) );`
   `Aborted…..`
4. What Should Happen When You Try:
   `assert( isWellFormedCommandString( "2n4s" ) );`
   `assert( !crossedOriginOnItsPath( "2n4s" ) );`
5. Textbook Example 5.1
   Code for this is in Lec5 slides

Class Scheduling

- Today is the last day of new content
- We will use Tuesday to review for the Midterm
- Midterm : No Scantron, No Devices, Closed Book But You Can Bring 1 Sheet Of Notes
  Please Bring A Photo-ID With You
  Some Students Have Documented Disabilities And Will Be Given Extra Time
- Release Some Sample Problems On To CCLE Sunday

Parameter Passing Schemes

Pass By Value

Declared as:

```
void foo( int i );
void foo( int i, int j );
void foo( string s, double d );
```

Invoked by:

```
foo( j );
foo( 12 );
foo( 'A' );
foo( true );
foo( 12.3456 );  /// possible loss of data warning
foo( d );        /// possible loss of data warning
```

What it means: 1. caller must send an rvalue to the type required
2. what arrives in the callee is a copy of the value sent

3. "very safe" computing because nothing the callee does changes the caller in any way
4. the only communication between the caller and the callee is the return value

Pass By Reference

Declared as:
```
void foo( int& i );
```
Invoked by:
```
int j = 0;
foo( j );
```
What it means: 1. caller must send an lvalue to the type required
2. what arrives truly what the caller sent
'an alias' to what the caller sent
3. "very strict" if you ask for an int, send int
4. output values from our functions……

Pass By Constant Reference

Declared as:
```
void foo( const int & i );
```
Invoked by:
```
int j;
foo( j );
```
What it means: 1. caller must send an lvalue to the type required
2. what arrives is truly what the caller sent
'an alias' for the exact thing that was sent
3. "very safe" computing because argument is "locked down"
the argument cannot be found on the left-hand-side of an assignment statement read-only value that cannot be change

Array Parameter

Declared as:
```
void foo( int array[ ], int size );
```
Invoked by:
```
int data[ 12 ];
```

```
foo( data, 12 );
```

What it means: 1. caller can send a 1-dimensional array of any size
2. a companion parameter (which is the size of the array) needs to be sent along with the array itself
3. any changes made to the array elements will be seen by the caller

Const Array Parameter

Declared as:
```
void foo( const int array[ ], int size );
```

Invoked by:
```
int data[ 12 ];
foo( data, 12 );
```

What it means: 1. caller can send a 1-dimensional array of any size
2. a companion parameter (which is the size of the array) needs to be sent along with the array itself
3. the function is not allowed to change any array elements

```
Int sampleArray[ 10 ];
fillArray( sampleArray, 10, numberUsed );
// entire array and all the data that was entered….
Sort( sampleArray, numberUsed );
Sort( sampleArray[ 1 ], numberUsed-1 );
Sort( sampleArray[ 5 ], 5 );     //// sort just the back half, assuming 10 values were entered
```