**Webster University**

**COSC 1560 Computer Programming II**

**Lab 6, October 30, 2014**

The object of this lab is to explore operator overloading.

* Download the program file [lab6.cpp](http://lab6_cpp.htm/) ([Here is a link to a plain-text version](http://lab6.cpp/))
* The program uses a constructor and eight functions, currently named

|  |  |  |  |
| --- | --- | --- | --- |
| anna | allen | fred | george |
| peggy | ralph | susan | operator<< (overloading the output operator) |

Manually (without running) examine the program listing and write the expected output:  
  
Original values: 7 4

Anna: 3

Allen: 7

Current values: 8 4

George: 12

Peggy: 9

Current values: 9 4

Susan: 4

Current values: 4 4

* Compile and run the program, circle anything above that differs from the actual output, and write the correct value to its right.
* Match the function to the operation listed in the table below:

|  |  |  |  |
| --- | --- | --- | --- |
| **function** | **returns** | | **other action** |
|  | **type** | **value** |  |
| george | MyDouble | sum of values |  |
| anna | MyDouble | difference of values |  |
| fred | double | value |  |
| allen | MyDouble | value before increment | value is incremented |
| peggy | MyDouble | value after increment | value is incremented |
| susan | MyDouble | value after copy | value is copied from argument |
| ralph | ostream& | current stream | value of object is sent to output stream |

* Six of the seven functions above do the expected job of overloaded operators, one is used by operator<<. Match operator functions to the named functions:

|  |  |
| --- | --- |
| anna | operator- |
| allen | operator++(int)           postfix increment |
| fred | operator double |
| george | operator+ |
| peggy | operator++(int)           prefix increment |
| ralph | caller of operator << |
| susan | Copy constructor |

* Replace the function names in lab6.cpp with the appropriate operator function names (the one called by operator<< is not replaced). In the definition of operator double(), leave off the type (that is it’s operator double(…) not double operator double(…). Compile and run it. If it doesn't produce the same output as the original, fix it and re-run it.
* Copy lab6.cpp into lab6b.cpp, replace the calls to the operator functions with the appropriate operators (in lab6b.cpp), compile, test and debug it.

|  |  |
| --- | --- |
| Use the following rules: | |
| Member functions, except operator double | |
| **In lab6.cpp** | **In lab6b.cpp** |
| a.operatorx(b) | a x b |
| a.operatorx() | x a |
| a.operatorx(0) | a x |
| operator double, and other type conversions | |
| a.operator double() | static\_cast<double>(a) *or*  (double)a *or* double(a) |
| Non-member functions | |
| operatorx(a, b) | a x b |

* Attach lab6.cpp and lab6b.cpp to lab 6 under Assignments in WorldClassRoom to submit them. Hand in you completed worksheet.