Pair Programming 8 Activities

* **Always use the pair programming tests to ensure your program works properly. Evaluation is based primarily upon correct execution. Activities without test screen shots and/or code will be earned. Both must be included.**
* **Take a screen shot with a white background of each execution in the tests.**
* **Download the source code file for inclusion in the turn in document.**
* **Turn in pair programming activities using the pair programming turn in document.**
* **It is each individual’s responsibility to turn in the assignment and pair programming is graded individually so make sure you share the work you and your partner did together as you go.**
* **Do not share work with your partner that you did not do together.**
* **Pair Programming is group work, but you can only work with your assigned partner. If you do not work with your partner, you can only earn 50% of pair programming points.**
* **Make sure you have your partner’s name, username and contact information such as Pellissippi WebMail.**

**PAIR PROGRAMMING 8A HAS CHANGED – INSTEAD OF WRITING THE LINKED LIST CODE, THE ASSIGNMENT PROVIDES THE LINKED LIST CODE WITH SOME ERRORS. PAIR PROGRAMMING 8A THEN REQUIRES STUDENTS TO FIND AND CORRECT THESE ERRORS. USING THE DEBUGGER WILL BE IMPERATIVE.**

8a. (7 points) This program requires the use of the IDE’s debugger to find seven errors in C++ code that uses a linked list. The code does not contain syntax errors – just one linker and seven logic/runtime errors. Download the LinkedListWeek8.zip file that contains a the program files for a program that uses a linked list of books. Unzip it and use the code to create files in an IDE’s workspace/project called pp8a. Execute the program. It should have the output in the tests provided with this assignment, but it has errors in it. Use the IDE’s debugger to find and correct these errors. Debugger tools you may find helpful are described below. See the IDE debugger’s Getting Started video in this week’s content for more information.

* Breakpoints – setting a breakpoint on a line of code means the program will pause execution before executing this line of code. It is useful to pause program execution so variable values can be verified. This is quicker that putting a cout statement in the code.
* Examine local variables – the value of variables can be seen while the program’s execution is paused. Used in conjunction with breakpoints, this allows the programmer to see all variable values that are in scope and is much quicker than a bunch of cout statements.
* Call stack – the call stack is the stack of function calls made to get to the currently executing statement or the last statement to execute before the program crashed. Therefore, the call stack can indicate what statement in what function caused the program to crash and what path of execution was taken to get to the statement.
* Continue/Stop functionality continues program execution until the next breakpoint or the program ends and stops program execution, respectively.
* Stepping into a function will “step into” a function as opposed to executing the function without pause. This is often used after a breakpoint to continue program execution inside the called function.
* Executing the next line of code in the debugger will execute the next instruction after pausing due to a breakpoint.
* Stepping out of a function will continue program execution after the return of the currently executing function and pause at the instruction after the function call.

Once the program executes correctly, capture a screen shot of the correct execution.

8b. (3 points) This program creates a binary file called books.bin, reads from it, and prints the books read from the binary file to standard output to verify the binary writing and reading work. In the IDE, create another workspace/project called pp8b. Copy the code in the Pair Programming 8 Assignment’s main8b.cpp into this project’s main.cpp. Add two new files to the project (DON’T ADD EXISTING FILES) book.h and book.cpp and copy the code from these pp8a files to the pp8b files. Add code to the Book class to complete pp8b which includes:

* ostream& write( ostream& ) const;
* istream& read( istream& );

Put b1.txt in the Pair Programming 8 Assignment in the program’s folder. Depending upon the IDE, cleaning or rebuilding (which cleans and builds) the project may remove this file, so if you clean and/or rebuild, you will have to recopy b1.txt.

See the Lecture Notes for further instructions and help with the pair programming.

Use the pair programming test cases to test your program. Make corrections if the program output does not match the tests. Capture screen shots of each correct execution.