**Final Part 1**

Directions:

Published text will be allowed (open book policy) along with handouts posted on Titanium course webpage and copies of your personal program listings from earlier assignments. No electronic media (hard drives, flash drives, CD’s, etc. ) including cell phones will be allowed except for authorized software (compiler, word processor). Furthermore, no access to the Web will be allowed during the exams except for accessing the Titanium site.

You might find yourself under some time pressure in this examination. Please check the point value for each problem so that you do not spend more time on one problem than it is worth. Please make sure to read each problem carefully before working on it. You will also get partial credit.

**Academic Dishonesty Policy**

Academic dishonesty includes such things as cheating, plagiarism, and helping someone else commit an act of academic dishonesty. Cheating is defined as the act of obtaining or attempting to obtain credit for work by the use of any dishonest, deceptive, fraudulent, or unauthorized means. Any test, paper or report submitted by you and that bears your name is presumed to be your own original work. The consequences of cheating and academic dishonesty – including a formal discipline file, possible loss of future employment opportunities – are simply not worth it. Additional information on this policy is available from University Policy Statement 300.021.

Please PRINT your name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Please SIGN your name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Please make sure to read each problem carefully before working on it. **Every file you turn in must have your name at the beginning as documentation. NO OTHER DOCUMENTATION IS REQUIRED unless specifically stated.**

|  |  |  |
| --- | --- | --- |
| Problem | Possible | Earned |
| 1 | 20 |  |
| 2 | 20 |  |
| 3 | 20 |  |
| Total | 60 |  |

**1.** Write a complete C++ program to calculate the **two** largest numbers in a set of quiz scores. Specifically, the program should do the following:

The program asks for and reads in N, the total number of students taking the quiz. The program will then prompt the user and reads in the N quiz scores into an array called scores.

The main program calls a function called **largest** that takes the array scores and the number N, and returns the largest of the N scores. The main program uses the result to print the largest score.

The main program then calls a function **secondlargest** that also takes the array scores and N and *any other parameter you wish to pass in*. The function **secondlargest** will return the *second* largest of the scores. The main program uses this result to print the second largest score.

For full credit, your program should handle the case where there are two or more students with the highest score. In this case, both the largest and second largest score will be the same.

Please write and use the functions described above! Note that both functions **largest** and **secondlargest** return a value.

Assumptions you may make to simplify coding:

* The number of quiz scores (N) will not be larger than 100.
* The quiz scores will be integers.
* All quiz scores are integers in the range from 0 to 100, inclusive.

Your program should be a single .cpp file. Name your program **prob1*YourLastNameFirstInitial*.cpp** where ***YourLastNameFirstInitial*** is replaced by your last name followed by your first name initial. Turn it in to the **Final Problem 1** on the CPSC 301 Titanium site. Put in your name at the start of the file as a comment.

Sample session (**Bold** numbers are printed by the program, normal numbers are typed in by the user.)

**Please enter the number of scores**

5

**Please enter the quiz scores**

73 82 91 76 70

**The largest: 91**

**The second largest: 82**

**2.** Write a complete C++ program that does the following:

The program then reads integers from the keyboard, and as long as the integers are positive or zero, the program inserts a new node holding that integer **at the rear of a linked list**.

When a negative integer is entered, the program should print out all the integers in the linked list **from front to back**. Note that this should write the numbers in the order in which the numbers were entered by the user.

Sample session (**Bold** text is printed by the program, normal numbers are typed in by the user):

**Please enter integers to be put into a linked list.**

**Type a negative number to stop.**

3 8 0 2 –2

**Numbers in the linked list:**

**3**

**8**

**0**

**2**

<program ends>

Hints:

* The program should maintain two pointers: **front** and **rear**.
* Remember to initialize the pointers!
* The **front** pointer should always point to the first node, and **rear** should always point to the last node.
* **Do not make a List class.** You may have a Node class or define a node as a struct ; your choice
* In the example above, the linked list at the end should look like the following:

8

2

0

3

rear

front

For full credit, **test** that your program works even for an empty linked list: if the user enters a negative integer as the first number, no numbers should be printed out. Example:

**Please enter integers to be put into a linked list.**

**Type a negative number to stop.**

–1

**Numbers in the linked list:**

<program ends>

Name your program **prob2*YourLastNameFirstInitial*.cpp** where ***YourLastNameFirstInitial*** is replaced by your last name followed by your first name initial. Submit it through the **Final Problem 2** assignment on the CPSC 301 Titanium site. If you wrote more than one file, make sure to submit them as well. You can name them what you wish as long as you include your last name in the file names. Remember to put in your name at the start of each file as a comment.

**3.** Write a **recursive** function called **recurseprint** that takes the front pointer to a linked list of nodes and *any other parameter you wish to pass in* to print ***every other value*** (i.e., the 1st value, 3rd value, 5th value, …) in the linked list beginning from the front. If this **recurseprint** function was called at the end of your program in problem 2, it should print out:

**3 0**

(i.e., only the first and third number from the four nodes are printed out).

**A non-recursive version of recurseprint** **will get no credit.**

Hints:

* The **recurseprint** function will need another parameter in addition to the front pointer. This additional parameter should be used within the recursive function to decide if printing should be skipped for that node’s value. What could this additional parameter be? What should it’s initial value be?
* To test if an integer variable i is odd or even, use the following code (% is the remainder operator in C++):

if (i%2 == 1)

Have the revised program with the **recurseprint** function in a file called **prob3*YourLastNameFirstInitial*.cpp** where ***YourLastNameFirstInitial*** is replaced by your last name followed by your first name initial. Submit it through the **Final Problem 3** assignment on the CPSC 301 Titanium site. Remember to put in your name at the start of the file.