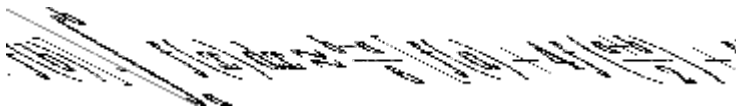


CSE 122– HW 5 Rubric

Student

Name: Noah Cherry

Grader
Name: Nils Carlson

Questions	Score	Total
<p>1) simpson.c</p> <p>Used Simpson's rule to calculate the definite integrals of:</p> <ul style="list-style-type: none"> - $\ln(x)$ from 0.5 to 2 (5 points) (should be 0.232868) - e^x from 0 to 1 (5 points) (should be 1.7183) - $\cos(x)$ from 0 to π (5 points) (should be 0) - $\sin(x)$ from 0 to π (5 points) (should be 2) - \sqrt{x} from 0 to 10 (5 points) (should be 21.082) <p>Code requirements:</p> <ul style="list-style-type: none"> - File is named simpson.c (1 point) - Used Simpson's rule (3 points)  <ul style="list-style-type: none"> - Used function pointers for $\ln(x)$, e^x, etc. (3 points) - No memory leaks (check using valgrind) (3 points) 	35	35
<p>2) scrabble.c</p> <ul style="list-style-type: none"> - File is named scrabble.c (1 point) <p>2a) <u>Binary search</u></p> <p>Code Requirements:</p> <ul style="list-style-type: none"> - No linear search - Wrote and used their own binary search (11 points) - No memory leaks (check with valgrind) (3 points) 	10	15
<p>2b) <u>Use set of tiles to determine the following</u></p> <p>Code Requirements:</p> <ul style="list-style-type: none"> - 1) Find a word (8 points) - 2) Determine the best play (8 points) - 3) quit (5 points) - No memory leaks (check with valgrind) (4 points) 	10	25

CSE 122– HW 5 Rubric

<p>3) heap.c (et. All)</p> <p><u>Implement a heapsort for a max heap</u></p> <p>Code Requirements:</p> <ul style="list-style-type: none"> - File names are grand.c and heap.c (2 points) - Performs heapify (insert nodes / make max heap) (7 points) - Deletes nodes (performing the actual heapsort) (7 points) - struct heap_t was not modified (6 points) - sort the heap in place -- another array is not allocated and used to build/sort the heap (7 points) - No memory leaks (check with valgrind) and the integers are read in from a file (6 points) 	35	35
<p>Subjective Criteria (10 points)</p> <ul style="list-style-type: none"> • Code is well commented • Code follows Linux Kernel Coding Style • No spaghetti code / bad variable names, etc. 	7	10
Total	97	120
<p>Comments:</p> <p>You do not input the dictionary properly and it segfaults. The problematic code is <code>words[i] = sscanf(*tok, "%s");</code> If you check the man page for sscanf, you will see that it returns an integer. What you need to do is malloc memory for every word and copy the word into the new memory. You also do not need to tokenize since fgets reads in one line into buf at a time. The body of the loop after the mallocing should be something like</p> <pre>words[i]=malloc(sizeof(char)*(strlen(buf) + 1)); strncpy(words[i], buf, strlen(buf) - 1); // exclude the newline character i++; data->nval++;</pre>		

CSE 122– HW 5 Rubric

With the fix to reading in the dictionary, your binary search does work.

Missing Doxygen comments and makefile