CS 251
Spring 2009
Voicu Popescu
Assignment 3
Due Friday February 6th, 11:59PM

Running Time Analysis and Stacks

1. Write a recursive function that given a positive integer *n* prints out the permutations of the numbers {1, 2, …, *n*}. For *n*=3, the permutations are {1, 2, 3}, {1, 3, 2}, {2, 1, 3}, {2, 3, 1}, {3, 1, 2}, {3, 2, 1}. The order in which the permutations are printed is not important. What is the Big-Oh running time of the function? *Warning: pay very close attention to the required file name, input and output format as an error here can be costly.*

2. Implement a C++ class that models the stack ADT. The elements are integers and are stored in an array. The array should increase in length by 50% when a push operation does not have room to store the new element (take the floor when the array length is odd, i.e. 9 becomes 13). Initial array length is 2. You should implement the operations in "stack.h" with the given signatures and explanations. *Warning: pay very close attention to the required file name, input and output format as an error here can be costly.*
   a. What is the Big-Oh amortized cost of a push operation?
   b. Use the stack class in a program that checks whether a string of characters is parenthetically correct, meaning that the parenthetic symbols (), {}, and [] are balanced and properly nested. The string ends with the character 'z' and does not contain any other 'z' characters. *Warning: pay very close attention to the required file name, input and output format as an error here can be costly.*

3. Extra-credit (3%) Write a program that solves the *Towers of Hanoi* puzzle (see problem C-4.6 at page 198 of textbook) for *n* = 5 using your stack class.

4. Turn in instructions
   a. Question 1: Name your program file as "permutation.cpp". "permutation.cpp" will have a main function that takes the input n as an argument to the executable. So, you should run with n=4 as follows:

      ```
      g++ -o permutation permutation.cpp
      ./permutation 4
      ```

      The main function calls the recursive function which prints out all permutations to the standard output. Print one permutation terminated by a newline at each line (The integers in a permutation are one after another, i.e. NOT separated by the space character).

      An example output for n=4 is provided as a file "permutation-sampleoutput". You should print all possible permutations. However, the order of the permutations may be different than this output (This file is given to show the output format, you will print out to standard output not to a file).

Of course once you get n=4 case to work you should devise additional cases with different n values to ensure your program truly works.

b. Question 2: Name your program file as “stack.cpp”. “stack.cpp” should implement the operations explained in “stack.h”. A sample test file “stackTest.cpp” is also provided. Examine and run that program to see what is expected. You can play with that file and add more test as you like. You will not turnin that file.

   Question 2 part b: Name your program file as “parenthetic.cpp”. “parenthetic.cpp” should read the input strings from file “parenthetic-input”. The input file has one string per line. The strings are read from the file. The program must print the answer as *yes* or *no* to the standard output per line corresponding to the each input line. A sample “parenthetic-input” file is provided. The corresponding output is provided as a file “parenthetic-output” (This file is given to show the output format, you will print out to standard output not to a file). (You do not have to evaluate the expressions)

c. Questions 1 and 2: Turn in a PDF document named as “a3.pdf” with your answers to the questions. **Read General turnin instructions parts E, F and G very carefully, on the website.**

d. Question 3 (Extra): Name your program file as “hanoi.cpp”. “hanoi.cpp” should solve *Towers of Hanoi* puzzle for *n* = 5 using your stack class. You need to solve the puzzle in the minimum number of moves. You are supposed to move all the disks from peg a to peg c. Your program should print out the elements in the three stacks from bottom to top at the beginning and after each move to the standard output. Assume that you have the elements 5,4,3,2,1 at peg a from bottom to top at the beginning. The format for output is given as files for n=2 (“hanoi2-output”) and n=3 (“hanoi3-output”). You can run your program with n=3, direct the output to a temporary file and diff your output with the given “hanoi3-output” file.

   Assuming that your executable is called “hanoi" which solves n=3:

   ```
   ./hanoi > myhanoi3-output
   diff myhanoi3-output hanoi3-output
   ```

   Then, make your program solve n=5, and check your output for correctness by yourself.

e. Your codes MUST compile on lore.cs.purdue.edu for it to be graded.

f. All of your documents (permutation.cpp, stack.cpp, parenthetic.cpp, a3.pdf, hanoi.cpp (OPTIONAL)) MUST be in the directory A3.

   In the directory above A3, run the command on lore.cs.purdue.edu

```
turnin -c cs251 -p A3 A3
```

optionally run the following command to verify what files were submitted

```
turnin -c cs251 -p A3 –v
```

Warning: turnin will be automatically disabled at 02/06/2009 at 11:59 PM EST and you will be unable to turnin your assignment after this time.

Warning: turning in multiple times is ok but only the last one will be graded as each turnin permanently overwrites the previous one.

Warning: failure to turnin exactly according to the instructions given above will result in your A3 receiving a grade of 0.