CS 32

Machine Problem 3

Deadline: 30 April 2017, 11:59PM

General Instructions

- For the machine problem, you are to write your implementation using the **C programming language**. You must use **gcc (version 4 or later)** as a reference compiler.
- The name assigned to the source code file that will contain the main method (assuming multiple source code files will be implemented) is *optbst.c*.
- Within the source code, place a comment at the very beginning stating your student number and your full name.
- In the event that multiple source code files were created, place them all in a compressed folder named *optbst.zip*.
- If you have consulted references (books, journal articles, online materials, other people), cite them as references by adding a comment detailing the references after your name. Use the MLA format for listing references: http://www.sciencebuddies.org/science-fair-projects/project_mla_format_examples.shtml. (For lines requiring italics, just place them inside quotation marks.)
- Add a comment before each function/method in your code, save for the main method, detailing the following: 1) input (if any) 2) output (if any) and 3) what the method does.
- You are only to submit the source code of your implementation. Submission of the machine problem should be done via e-mail. Attach the source code file/s, and write as the subject header of the e-mail: [CS 32] $-$ $<$ *Student Number* $>$ $-$ $<$ *Last Name, First Name* $>$ $-$ Machine Problem 3. For example, [CS 32] 202099969 - Kreuβ, Jonathan - Machine Problem 3. Send your answers to janmichaelyap@gmail.com.
- Together with your MP submission, **send a screenshot of your Command Prompt or Terminal after you have entered the following commands (and pressed Enter): gcc -v**. This will let me know the set-up of your GCC compiler in the event that I have technical problems with compiling and/or running your submission.
- **You should receive a confirmation e-mail from me stating receipt of your deliverable within 24 hours upon your submission of the MP**. If you have not received any, forward your previous submission using the same subject header once more.
- If your *program does not compile and/or does not run properly for at least one of the test inputs even after minor edits to the source code*, then your **submission is deemed invalid and will not be accepted**. You will be asked to submit a working deliverable. **This provision will also hold for subsequent (repeat) submissions**. **Date of submission is considered to be the date when an acceptable deliverable was submitted.**
- If you have any questions regarding an item (EXCEPT the answer and solution) in the machine problem, do not hesitate to e-mail me to ask them. However, **questions regarding this MP forwarded/received on or after 12:01am of 21 April 2017 will**
- **Follow the instructions as mentioned above, especially those pertaining to actual submission of the deliverable. Lack of or erroneous implementation of any part of the instructions may be considered a deficient submission, and thus may not be accepted. NOT be entertained**.

Optimal Binary Tree

For your final machine problem in CS 32, you are to implement the OPTIMUM_BST and DISPLAY_OPTIMUM_BST procedures as stated in Chapter 14.4 of our main text.

Implementation Specfications

For the implementation part, the input should be a text file where the content is composed of three lines, each containing white space separated values. The first line should contain the string-typed search keys for the binary tree, presumed to be in alphabetical order. The second line should contain the corresponding search probabilities (i.e. p_i's as described in Chapter 14.4) for each search key. The third line should contain the q-probabilities as described in Chapter 14.4. Assume that for the second and third lines, the data type is double-precision float. The output of your program should be a comma-separated listing of the search keys of the resulting optimum binary search tree, the order of the keys to be displayed as dictated by the DISPLAY_OPTIMUM_BST procedure in Chapter 14.4. A required feature in your implementation should be that the code is a straightforward implementation of the procedures mentioned, including how the data structure $\mathbb{B}$ is defined and used.

A .zip file containing text files is posted in the web page for you to test out your implementation.

Sample Run

In a file named *input1.txt*, let the following be the input:

Aliguyon Carpio Indarapatra Lam-ang Makiling Penduko Pumbakhayon Sulayman
0.10 0.06 0.08 0.04 0.02 0.09 0.12 0.08
0.04 0.03 0.05 0.09 0.03 0.06 0.05 0.02 0.04

```
Program started.

Enter file name> input1.txt

Optimal BST: Penduko, Indarapatra, Aliguyon, NULL, Carpio, NULL, NULL, Lam-ang, NULL,
Makiling, NULL, NULL, Pumbakhayon, NULL, Sulayman, NULL, NULL
```

Criteria for Grading

The machine problem submission will be graded via the following criteria:

- Program effectivity (50%)
  The program should be able to properly process erroneous inputs and be able to perform the algorithm according to specification on a number of test cases.
- Adherence to specification (30%)
  The implementation should, to the fullest extent possible, adhere to the specifications set for the machine problem.
- Program documentation and readability (20%)
  The program should have proper internal documentation (i.e. comments detailing functionalities), and that the documentation details the mechanisms of each function and some code blocks clearly. The code should also be readable in the sense that function and variable names are as intuitive as possible, and that proper indentations/blocking of statements is observed (e.g. statements on the same level/block should be properly aligned, statements inside a *for* loop should be indented, etc.).