Coursework 2

This section covers what you learnt and practised in weeks four, five and six.

# Code task

You need to code a Hash Table to store and retrieve a retail’s store name, full address (street, town, and postcode) and size ban. Remember that a hash table has three main components:

**key**: This should be the retailer's name

**value**: A structure **ST** holding the {Store name, full address (street, town, and postcode) and size band} values

**hash function**: Using the techniques reviewed in the lectures, construct a hash function based on the retailer's name.

Your program input is a retailer’s name and a postcode, and the output is the structure **ST.**

For example, if the input is Sainsburys, AB10 1HW, the output should be

Sainsburys Aberdeen St. Nicholas Centre Local, St. Nicholas Centre, Aberdeen, AB10 1HW, < 3,013 ft2 (280m2)

# Algorithm design

For this task, you will be assigned to one of the following groups:

* Group A: Bubble sort
* Group B: Merge sort
* Group C: Quick sort

Design an algorithm that sorts all postcodes alphabetically in ascending order. You need to use the sort algorithm from your assigned group. Your algorithm needs to include the sorting bit explicitly, i.e., you cannot just call a function quicksort(A) without providing the algorithm for that quicksort function.

For this part, you can use the program to order words from week’s three lab.

After sorting, your algorithm should perform a binary search for all postcodes starting with NW10

You need to report the number of steps your algorithm would take (in average), i.e., sorting + searching. For example, if sorting takes O (n log n) and searching all NW10 is O(n), then the total time would be O (n log n) + O (n).

# Marking Scheme

# Code task

|  |  |
| --- | --- |
| Mark | Assessment criteria |
| 100 | An assessment that could not be bettered within the time available |
| 82,85,92 | The hash function is programmed from scratch with a robust and welldesigned hash function. The hast table holds the ST structure. The code allows the user to perform more than one search by retailer’s name - postcode and provide the functionality to end the program. The output is presented in the form of a table, where each column is labelled. Coding style is remarkable (Optimal use of memory (in C), proper use of data frames (Python); sensible variable names; proper comments and indentation) |
| 72,75,78 | The hash function is programmed from scratch, and the hast table holds the ST structure. The code allows the user to perform more than one search by retailer’s name - postcode and provide the functionality to end the program. The output is presented in the form of a table, where each column is clearly labelled. Coding style is acceptable (sensible variable names, proper comments, and indentation |
| 62,65, 68 | The hash function is programmed from scratch, and the hast table holds the ST structure. The code does not allow the user to perform more than one search by retailer’s name - postcode and / or does not provide a functionality to end the program. The output is not presented in the form of a table. The coding style presents some omissions like not informative variable names or trivial comments. |
| 52,55,58 | The hash function is NOT programmed from scratch, i.e., use functions provided by the language program (like hash () in python or GetHashCode () in C#). The hast table holds the ST structure. The code does not allow the user to perform more than one search by retailer’s name - postcode and / or does not provide a functionality to end the program. The output is not presented in the form of a table. The coding style presents some omissions like not informative variable names or trivial comments. No comments are provided and/or irrelevant variable names |
| 42,45,48 | Same criteria than the 52,55,58 range plus the program does not return the expected output |
| 32,35,38 | Same criteria than the 42,45,48 range plus it throws one or more warnings / errors |
| 5,15,25 | The code is provided but it does not run. |
| 0 | No code provided or the code has nothing to do with the required task. |

# Algorithm design

|  |  |
| --- | --- |
| Mark | Assessment criteria |
| 100 | An assessment that could not be bettered within the time available |
| 82,85,92 | The algorithm presents an optimal strategy to sort postcodes based on the given algorithm (group A, B or C). The search strategy, based on a binary search, is optimal. In addition, the design includes an insight reflection for the number of steps reported. The algorithm follows the required style. |
| 72,75,78 | The algorithm presents an acceptable strategy to sort postcodes based on the given algorithm (group A, B or C). The search strategy, based on a binary search, is acceptable. The algorithm follows the required style. |
| 62,65, 68 | The algorithm presents an acceptable strategy to sort postcodes based on the given algorithm (group A, B or C). The search strategy, based on a binary search is acceptable. The algorithm does not follow the required style. |
| 52,55,58 | The algorithm is correct but presents some deficiencies in either the sorting/searching tasks or both. The algorithm follows the required style |
| 42,45,48 | The algorithm is correct but presents some deficiencies in the sorting/searching tasks, but it mostly follows the required style. |
| 32,35,38 | The algorithm is mostly correct and presents some deficiencies in the sorting/searching tasks, and does not follow follows the required style. |
| 5,15,25 | The algorithm is not correct |
| 0 | No algorithm provided |

The final mark is the average between coding task and algorithm design.

**Academic Misconduct**

All submissions will be processed through a code plagiarism tool. If signs of misconduct are found, all students involved will be contacted to discuss further steps. Please see here for information on academic integrity at the university [https://portal.roehampton.ac.uk/information/Pages/Academic-Integrity.aspx.](https://portal.roehampton.ac.uk/information/Pages/Academic-Integrity.aspx)

Our guiding principle is that academic integrity and honesty are fundamental to the academic work you produce at the University of Roehampton. You are expected to complete coursework which is your own and which is referenced appropriately. The university has in place measures to detect academic dishonesty in all its forms. If you are found to be cheating or attempting to gain an unfair advantage over other students in any way, this is considered academic misconduct and you will be penalised accordingly. Please don’t do it.