**(Recommended Length of 20 pages or less)**

**COMP 1828 - Designing, developing and testing solutions for the London Underground system**

**NOTE:**

1. **The group leader must submit the group coursework. If another member submits it, valid justification and supporting evidence must be provided. Multiple submissions from different members, possibly arising from disagreements, may result in reduced marks and potential plagiarism issues.**
2. **The group should ensure that no member is excluded, provided they make an effort to participate, regardless of whether they start contributing at a later stage.**
3. **Each member of the coursework group is required to provide weekly evidence of their work.**

**TASK 1**

**Operational Station Status System**

**[20 marks]**

You may retain only those headings in bold to conserve space in the report.

**(1a)**

**Manual versus Code-Based Execution of a Data Organisation Strategy**

* **Data Structure Selection**: A Chained Hash Table (from Introduction to Algorithms, 4th Edition – Cormen et al., Chapter 11) was selected to manage the daily list of operational London Underground stations.

Justification:

The chained hash table supports:

• Insertion O(1) average time

• Deletion O(1) average time

• Membership check O(1) average time

Stations are assigned to “buckets” by a hash function; collisions are resolved through linked lists (chaining).

This provides excellent performance for networks with constant updates to operational status.

Other structures (e.g., arrays O(n) or balanced trees O(log n)) are slower for frequent lookups.

* **Create a Simple Dataset**:

A screenshot of a black screen

AI-generated content may be incorrect.

* **Manual Application**: Each station was inserted sequentially into an 8-bucket hash table.

The internal state after each insertion is shown below.

A screenshot of a computer program

AI-generated content may be incorrect.

A black background with white text

AI-generated content may be incorrect.

* **Code Implementation and Verification**:

**A computer screen shot of a program

AI-generated content may be incorrect.**

**(1b)**

**Empirical Measurement and Application**

**Empirical Performance Measurement:**

* **Data Generation**: Artificial datasets of n unique “station IDs” were generated directly in the program using a for loop with range(n).

Each integer (0 to n − 1) represents one unique operational station and is inserted into a new ChainedHashTable.

A black background with white text

AI-generated content may be incorrect.

* **Time Measurement**: For each dataset size n ∈ { 1 000, 5 000, 10 000, 25 000, 50 000},

the program performed 10 000 membership-check queries (a mix of hits and misses) using random numbers up to 1.2 × n.

Python’s time.perf\_counter() was used to measure total query time, from which the average time per lookup was computed.

Each test therefore measured the empirical cost of the .search() operation within the CLRS ChainedHashTable.

A screenshot of a black screen

AI-generated content may be incorrect.

* **Performance Analysis & Plotting**:

A graph with a line going up

AI-generated content may be incorrect.

Tool used – matplotlib.

The plot of average lookup time versus dataset size n shows only a slight upward drift at larger n, confirming the expected O(1) average-case performance.

Small variations arise from Python interpreter overhead and CPU caching effects rather than the algorithm itself.

* **Comparison**: The measured results closely match the theoretical analysis of hash-table operations.

Both insertion and membership checking remain effectively constant-time for practical dataset sizes.

Any deviations from ideal O(1) are attributable to system-level timing noise, not data-structure inefficiency.

**Application with London Underground Data:**

* **Data Acquisition and Implementation:** The file data.csv (derived from London Underground data.xlsx) was parsed using pandas.

The first string-type column was selected as station names, trimmed, lower-cased, and duplicates removed.

An instance of ChainedHashTable was populated with all station names.

**A computer screen shot of a program code

AI-generated content may be incorrect.**

* **Testing:**

**A screen shot of a computer

AI-generated content may be incorrect.**

**TASK 2**

**Journey Planner Based on Journey Time**

**[20 marks]**

You may retain only those headings in bold to conserve space in the report.

**(2a)**

**Manual versus Code-Based Execution of a Data Organisation Strategy**

* **Data Structure and Algorithm Selection**:
* **Create a Simple Dataset**: [Include a clear image (scanned if hand-drawn, or a screenshot if digitally created)]
* **Manual Application**: [Include a clear image (scanned if hand-drawn, or a screenshot if digitally created)]
* **Code Implementation and Verification**: [Provide key code fragments (not the full code), specifically:]
  + The part of your Python code that implements the dataset
  + The section where you call or use the required library code, as verification of compliance
  + Etc.

**(2b)**

**Empirical Measurement and Application**

**Empirical Performance Measurement:**

* **Data Generation**: [Present key code snippets (not full code) for generating an artificial tube network dataset. Include proper references if using external sources]
* **Time Measurement**:
* **Performance Analysis & Plotting**: [Plot a graph of average execution time vs network size *n* (where *n* is the total number of stations)]
  + State the tool used for plot generation
* **Comparison**: [Compare the plotted graph to the algorithm's theoretical time complexity, discussing any discrepancies or alignments]

**Application with London Underground Data:**

* **Data Acquisition and Implementation:**
* **Testing:**

**TASK 3**

**Journey Planner Based on Number of Stops**

**[20 marks]**

You may retain only those headings in bold to conserve space in the report.

**(3a)**

**Manual versus Code-Based Execution of a Data Organisation Strategy**

* **Data Structure and Algorithm Selection**:
* **Create a Simple Dataset**: [Include a clear image (scanned if hand-drawn, or a screenshot if digitally created)]
* **Manual Application**: [Include a clear image (scanned if hand-drawn, or a screenshot if digitally created)]
* **Code Implementation and Verification**: [Provide key code fragments (not the full code), specifically:]
  + The part of your Python code that implements the dataset
  + The section where you call or use the required library code, as verification of compliance
  + Etc.

**(3b)**

**Empirical Measurement and Application**

**Empirical Performance Measurement:**

* **Data Generation**: [Present key code snippets (not full code) for generating an artificial tube network dataset. Include proper references if using external sources]
* **Time Measurement**:
* **Performance Analysis & Plotting**: [Plot a graph of average execution time vs network size *n* (where *n* is the total number of stations)]
  + State the tool used for plot generation
* **Comparison**: [Compare the plotted graph to the algorithm's theoretical time complexity, discussing any discrepancies or alignments]

**Application with London Underground Data:**

* **Data Acquisition and Implementation:**
* **Testing:**

**TASK 4**

**Analysis of the Core Network Backbone**

**[20 marks]**

You may retain only those headings in bold to conserve space in the report.

**(4a) Manual versus Code-Based Execution of a Core Network Algorithm:**

* **Algorithm Selection**:
* **Create a Simple Dataset**: [Include a clear image (scanned if hand-drawn, or a screenshot if digitally created)]
* **Manual Application**: [Include a clear image (scanned if hand-drawn, or a screenshot if digitally created)]
* **Code Implementation and Verification**: [Provide key code fragments (not the full code), specifically:]
  + The part of your Python code that implements the dataset
  + The section where you call or use the required library code, as verification of compliance
  + Etc.

**(4b)**

**Empirical Measurement and Application**

**Empirical Performance Measurement:**

* **Data Generation**: [Present key code snippets (not full code) for generating an artificial tube network dataset. Include proper references if using external sources]
* **Time Measurement**:
* **Performance Analysis & Plotting**: [Plot a graph of average execution time vs network size *n* (where *n* is the total number of stations)]
  + State the tool used for plot generation
* **Comparison**: [Compare the plotted graph to the algorithm's theoretical time complexity, discussing any discrepancies or alignments]

**Application with London Underground Data:**

* **Core Backbone Calculation:**
* **Testing and Impact Analysis:**

**Progress Journal, Compliance with Instructions, Clarity of Language, etc.**

**[20 marks]**

1. **Answer the following questions by keeping only one of the options (YES or NO) and removing the other.** 
   1. **Are the group members identical to those listed in Moodle? YES / NO.**
   2. **Have the final cumulative credits, which will appear in the final report, been sent to all members in good time to allow for any potential discussion (i.e., not left until the last minute near submission time)?YES / NO.**
   3. **Has a copy of the final report been sent to all the members prior to submission? YES / NO.**
   4. **Does any group member have EC (Extenuating Circumstances) approval? YES / NO.**
      1. **If you answered YES to the EC question above, please answer the following: Has the member's name been specified, and has evidence of the EC approval been included in this report (e.g., at the end)? YES / NO.**
2. **A summary of final credits of member contribution**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Name | ID in 9 digits | Login ID or Email (e.g. jk7492y) | Cumulative credit (0% - 100% per person) |
| 1 | Surname, First Name (Group leader) | ????????? | ??????? | % |
| 2 | Surname, First Name | ????????? | ??????? | % |
| 3 | Surname, First Name | ????????? | ??????? | % |
| 4 | Surname, First Name | ????????? | ??????? | % |
| 5 | Surname, First Name | ????????? | ??????? | % |
| 6 | Surname, First Name | ????????? | ??????? | % |

(Member 1 should be the group leader.)

1. **Contribution table**

This table serves as the final record of each member's actual work contribution. Complete the table below, adapting the columns for your group's size and members. For each member, place a tick (e.g., ✓) in the row corresponding to each subtask to which they made a significant contribution. The final pattern of ticks needs to meet the requirements described in the section that follows.

1. ***Initial* allocation before the commencement of work:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Name** | **Subtask 1(a)** | **1(b)** | **2(a)** | **2(b)** | **3(a)** | **3(b)** | **4(a)** | **4(b)** |
| 1 | Surname, First Name (Group leader) |  |  |  |  |  |  |  |  |
| 2 | Surname, First Name |  |  |  |  |  |  |  |  |
| 3 | Surname, First Name |  |  |  |  |  |  |  |  |
| 4 | Surname, First Name |  |  |  |  |  |  |  |  |
| 5 | Surname, First Name |  |  |  |  |  |  |  |  |
| 6 | Surname, First Name |  |  |  |  |  |  |  |  |

1. ***Final* allocation/contribution upon completion of the work:**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **#** | **Name** | **Subtask 1(a)** | **1(b)** | **2(a)** | **2(b)** | **3(a)** | **3(b)** | **4(a)** | **4(b)** |
| 1 | Surname, First Name (Group leader) |  |  |  |  |  |  |  |  |
| 2 | Surname, First Name |  |  |  |  |  |  |  |  |
| 3 | Surname, First Name |  |  |  |  |  |  |  |  |
| 4 | Surname, First Name |  |  |  |  |  |  |  |  |
| 5 | Surname, First Name |  |  |  |  |  |  |  |  |
| 6 | Surname, First Name |  |  |  |  |  |  |  |  |

Briefly describe the process your group followed to manage the work allocation, covering your initial plan and any subsequent redistributions made as the project progressed. For example, a fair method for your initial plan may be to first collectively agree on bundling the eight subtasks into four pairs of equivalent effort, and then assign these bundles randomly among the members.

The **final work contribution**, as recorded in the table above, must then meet the following requirements:

* **Core Requirement:** To ensure a fair and consistent workload for every active member regardless of group size, each person is responsible for delivering a total amount of work equivalent to **completing at least two of the eight subtasks entirely on their own, as if it were an individual piece of coursework.** The total volume of your individual contributions across all assigned subtasks needs to meet this standard.
  + **Example 1 (Shared Effort):** If two members are assigned to four different subtasks and split the work for each one equally (50/50), then each member has met the requirement (4 subtasks × 50% contribution = 2 full subtask equivalents).
  + **Example 2 (Parallel Development):** Alternatively, if two members are assigned to the same set of two subtasks and work on them independently in parallel, each producing a complete solution, then each has also met the requirement (2 subtasks × 100% contribution = 2 full subtask equivalents). The group can then review both solutions and choose the better one, or synthesise a superior hybrid version for the final submission.
* **Recommendation:** A common but risky approach is to assign each subtask to only one person. This creates a single point of failure if that person is delayed or their work is not up to standard, and it prevents the collaboration essential for improving quality. Therefore, it is **highly recommended** that **every subtask has at least two members** assigned to it. This creates redundancy, mitigates risk, and provides a crucial opportunity for discussion and quality control, which leads to a higher-quality outcome. This may not always be feasible, especially in smaller groups, but should be implemented where possible.

1. **Weekly progress log**

|  |  |
| --- | --- |
| **Week** | **Brief description of each member's contributions; Confirmation of weekly email sent by each member; Attendance at weekly Teams meeting; Cumulative credit earned up to that week** |
| 20/Oct - | Simon (Contributed to Task 1a, email sent, Teams meeting not attended, 80%), Kai (…..), …. |
| 27/Oct - |  |
| 03/Nov - |  |
| 10/Nov - |  |
| 17/Nov - |  |

1. **Evidence of a weekly email on cumulative credits and a Teams online meeting**

For each week, you need to provide the two pieces of evidence:

* Screenshot of the group leader's email to all members, showing: Sender, Recipients, Date, and Content (names and cumulative credits). For instance:

A screenshot of a phone

Description automatically generated

* Screenshot of Teams meeting (on a non-Monday). For instance,



* You may consolidate evidence to save space, e.g., all email screenshots followed by all Teams meeting screenshots.

- Week 20-26/Oct :

- Week 27/Oct – 2/Nov:

- Week 3-9/Nov:

- Week 10-16/Nov:

- Week 17-23/Nov: