COMP3506/7505 Project

Weighting: 25%

Due date: 23rd September 2022

Task

This project will require you to implement algorithms efficiently in order to solve desired tasks. In particular, you will be required to:

1. Complete programming tasks in reference to a given algorithm or desired efficiency.
2. Write a report explaining the efficiency of your implementation.

The specific details required for each of these two points are outlined in the questions below.

Submission Details

All assignment related documents and files will be submitted via Gradescope. The programming sections of your assignment will be marked by auto-grading software while the report section of your assignment will be marked by a tutor. Therefore, you will need to submit to two separate Gradescope submission portals. The name of these submission portals and the documents required to submit to them are listed below.

- **Autograder:** You must submit the programming part of your assignment to the autograder portal. You should only submit either the Java or Python code according to the language you have used. You *must not* submit your report PDF here. If you do so and fail to submit the project PDF to the Report Portal (see below), penalties will apply.

  You should submit only the following files:

  For *Python*: `hospital_1.py, hospital_2.py, hospital_3.py, patient.py, login_system.py tree_of_symptoms.py, alert_system.py`[1].

  For *Java*: `Hospital1.java, Hospital2.java, Hospital3.java, Patient.java, LoginSystem.java TreeOfSymptoms.java, AlertSystem.java`[1].

  Do NOT modify and do NOT submit the interface files (ending with `*Base.java`/ `*_base.py`)

  Your marks for this programming task will be calculated by running a series of tests against your solution using the Gradescope autograder. Each test will be assigned a certain number of marks according to the rigour of the test. The results of some tests will be available instantly after submission while others will be hidden until the assignment marks are released.

- **Report:** You must submit a PDF export of your report to the report portal. You must use the template provided for your report. If you do not use the template or you modify the template in anyway, with the exemption of adding your answers in the box provided, you will incur a penalty.

[1] The last file is for COMP7505 only.

Programming Details

Java Instructions

- You must use Java version 11 or higher. Additional language features introduced after version 11 may not be supported. It is recommended you use OpenJDK, for example, AdoptOpenJDK 11.
- Your solution will be automatically marked. No marks will be awarded for non-compiling submissions.
- You should NOT use any classes from the Java Collections Framework (JCF) (e.g. `ArrayList`, `LinkedList`, `HashMap`, `HashSet`, `Arrays`), unless an individual question specifies that JCF classes are allowed. If you wish to utilise similar functionality, you should implement the needed algorithms or data structures yourself. You may use other classes from `java.util` outside of the JCF, such as `Iterator`, `Stream`, `Random`, `Objects`. If you are unsure about whether a certain class is allowed to be used, please contact teaching staff immediately.
- You should NOT use any additional third-party libraries. No marks will be awarded to submissions which import non-supported libraries.
- Remove the `main()` method and any `System.out.print()` calls before submitting to the autograder.

Python Instructions

- You must use Python version 3.7 or higher.
- Your solution will be automatically marked. No marks will be awarded for non-compiling submissions.
- You should NOT use Python built-in sort methods, such as `list.sort()` or `sorted()`. You should NOT use Python built-in list methods, such as `append, clear, count, copy, extend, index, insert, pop, remove, reverse, sort`, or built-in methods `min` or `max`. If you wish to utilise similar functionality, you should implement the needed methods yourself.
- You should NOT use dictionary `dict` nor {}.
- The only permitted libraries you may use are `math` and `random`. You should NOT use any additional libraries, such as `numpy`, `scipy`, and `collections`. No marks will be awarded to submissions which import non-supported libraries.
- Remove `__main__` method and `print` functions before submitting to the autograder.

Late Submissions and Extensions

Please consult the courses ECP for the policies and procedures regarding late submission and extensions. Note that course staff **cannot** process requests for extension or otherwise. All such applications must be made through the school in accordance with ECP.

Academic Misconduct

This assignment is an individual assignment. Posting questions or copying answers from the internet is considered cheating, as is sharing your answers with classmates. All your work (including code) will be analysed by sophisticated plagiarism detection software.

Students are reminded of the University's policy on student misconduct, including plagiarism. See the course profile and the School web page: https://itee.uq.edu.au/current-students/guidelines-and-policies-students/student-conduct?p=1#1.

1 Hospital Appointment System

In this question, you need to implement an appointment system for three different hospitals, that would allow patients to book appointments according to their desired time. In particular, you need to complete the following functionality:

- `iterator`/`__iter__` - iterates through the patients in the correct order according to their time
- `addPatient`/`add_patient` - adds a new patient to the system. Returns True if the patient is successfully added, returns False otherwise.

It is known that all three hospitals work from 08:00 to 18:00 with a lunch break from 12:00 to 13:00. Additionally, each hospital has different time complexity requirements for each function. Based on these requirements, you have to chose the best data structure to represent the appointment system, as well as the best algorithm to order the patients by their times. The requirements are the following:

- **Hospital 1**: The timeslots are fixed. Each appointment lasts exactly 20 minutes. For example, a new patient may book an appointment for 08:00, 08:20, 08:40, etc.

  Any other timeslots, e.g. 08:01, 08:22, 08:41 are considered incorrect and should result in a patient not being added to the system. Each timeslot may be occupied by at most one patient, i.e. two patients can not both be booked for the 08:40 appointment. The patient is not added to the system if they request a timeslot that is already occupied.

  `addPatient`/`add_patient`; `iterator`/`__iter__` - as quickly as possible.
- **Hospital 2**: `addPatient`/`add_patient` in $O(n)$; `iterator`/`__iter__` - as quickly as possible. If two patients requested the same time, give a priority to the patient that is already in the system.
- **Hospital 3**: `addPatient`/`add_patient` in $O(1)$; `iterator`/`__iter__` - as quickly as possible. If two patients requested the same time, give a priority to the patient that is already in the system.

1.1 Programming

In the files `Hospital1.java`/`hospital_1.py`, `Hospital2.java`/`hospital_2,py` and `Hospital3.java`/`hospital_3.py`, you should implement the methods in the interface `HospitalBase`.

1.2 Report

Using the Gradescope document template provided, you must complete a report which analyses your code and the algorithms you have implemented. In the report, for each of the three hospitals you should:

1. State the data structure used to store patients and explain why this data structure is the best for the task in hand.
2. Describe the algorithm used to order the patients. Briefly explain how it works, from where it is called (from `iterator`/`__iter__` or from `addPatient`/`add_patient`) and why it is the best algorithm in comparison with the other known algorithms.
3. Assuming n to be the number of patients, state the best-case (Ω) and worst-case (O) time complexity of `iterator`/`__iter__` and `addPatient`/`add_patient` with respect to n.

PROJECT CONTINUES ON NEXT PAGE

2 Login System

Aside from keeping track of appointments throughout the day, hospitals are also required to store a secure collection of users who have signed up to the hospital appointment system. To accomplish this, you are required to implement a login system whereby users can sign up by providing their email address and password. Users must also be able to be removed from the system and change their password.

Plain text passwords are never stored, but the hash code is stored for checking against. Note that no two users can have the same email address, but many users may have the same password.

The login system should be implemented in the form of a hash table, with the following characteristics:

- **Size:** The hash table should be initialised to a static size $M = 101$. If the number of users exceeds the current size, resize the hash table using a tripling strategy: triple the size of the hash table.
- **Hash Function:** You should implement a hash function that consists of a custom hash code and a compression function.
  - **Hash Code:** For the hash codes use the following function. Assume that the key represents a string $s : s_1, s_2, s_3, ..., s_n$; and A_n represents an ASCII value for the letter s_n, and $c = 31$ is a constant. Then the hash codes should be calculated as follows:

$$h_1(s_1, s_2, .., s_n) = \Big[\Big(\big((A_1 * c + A_2) * c + A_3\big) * c + A_4\Big) * c + ...\Big] + A_n$$

  - **Compression Function:** For the compression function, use a division by a hash table size:

$$h_2(y) = y \ mod \ M$$

- **Collision Handling:** Use linear probing to handle collisions.

2.1 Programming

In the files `LoginSystem.java/login_system.py`, you should implement the methods in the interface `LoginSystemBase`.

2.2 Report

1. What is the advantage of storing a hash code of the password instead of the plain text password?
2. Do we need to store the email in the hash table? If your answer is yes, explain why it is necessary. If your answer is no, explain how you use the email (if you use it at all)?
3. Which of the following is an example of a collision? Explain your answer for both cases
   (a) Two users have the same email hash
   (b) Two users have the same password hash
4. What is the type of hash code function being used? Explain why it is suitable for use in this hash table.

PROJECT CONTINUES ON NEXT PAGE

3 Tree of Symptoms

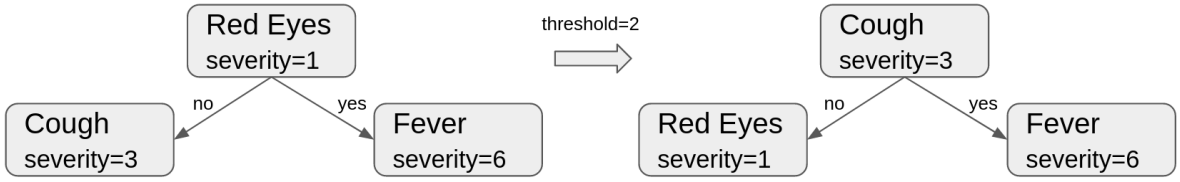

Figure 1: Tree of symptoms

Recently graduated doctors get nervous interviewing patients for the first time, so the hospital made a simple cheat-sheet of symptoms they need to enquire about. The cheat-sheet is in the form of a tree, where each node represents a symptom and its severity level, and a higher severity number indicates a more severe symptom. Following the tree of symptoms helps the new doctors to select the questions to ask and to eventually diagnose a patient with the correct disease.

An experienced doctor noticed that the current version of a cheat-sheet is not very efficient and proposed to improve it by arranging mild symptoms on one side of the tree, and severe symptoms on the other side. Additionally, he proposed to start an enquiry with the symptom of a particular severity, so that new doctors can quickly identify if the disease is serious and call for back-up if necessary.

You are asked to perform this upgrade of the cheat-sheet. In particular, you need to restructure a tree of symptoms in the following way:

1. The new root symptom must be the symptom with the smallest severity that satisfies `severity >= threshold`. If no such symptom exists, use the most severe symptom among existing as a new root node.
2. For all the nodes, the left sub-tree must contain more mild symptoms, and the right sub-tree must contain more severe symptoms.

For simplicity, we further introduce the following assumptions about the tree:

- There is at least one symptom in a tree.
- Each symptom has a unique severity level (no duplicates).

3.1 Programming

In the file `TreeOfSymptoms.java`/`tree_of_symptoms.py`, you should implement the methods in the base class `TreeOfSymptomsBase`.
Note that for this question, for Python you may use all the Python built-in methods, dictionaries and lists. For Java, you may use any of the classes from the Java Collections Framework. Any other additional third-party libraries are still NOT allowed.

3.2 Report

1. What is the type of the restructured tree?
2. For any given binary tree, is the reconstructed tree balanced? If so, explain why. If not, give a counter-example.
3. For any given binary tree, is the reconstructed tree unique? If so, explain why. If not, give a counter-example.

PROJECT CONTINUES ON NEXT PAGE

4 Alert System (COMP7505 Students Only)

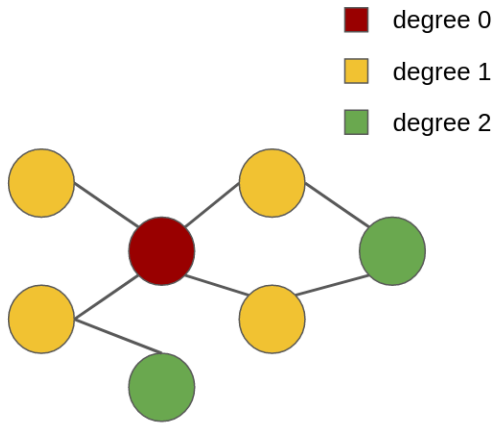

Figure 2: Graph of patients. Red node is the person with the infection, yellow and green nodes are their contacts of different degrees.

In this question, you need to implement an alert system that is capable of keeping track of people contracting some virus. The alert system should be organised in the form of a graph, where a node represents a person, and an edge between two nodes represents the fact that those two people were in contact with each other. If the virus is not contagious, only the person with the virus is marked as infected in the system (degree 0). However, if the virus is contagious, their contacts (degree 1) and the contacts of their contacts (degree 2) might be infected as well. Depending on how contagious the virus is, the neighbors of different degrees must be marked as infected.

Programming In the files `AlertSystem.java`/`alert_system.py`, you should implement the methods in the abstract class `AlertBase`.
Note that for this question, for Python you may use all the Python built-in methods, dictionaries and lists. For Java, you may use any of the classes from the Java Collections Framework. Any other additional third-party libraries are still NOT allowed.