CS2040S Semester 1 2023/2024

Data Structures and Algorithms

Tutorial+Lab 01

Basic Java, Basic OOP, Analysis, Hands-on 1

For Week 03

Document is last modified on: August 21, 2023

1 Introduction and Objective

The purpose of this first tutorial+lab session is to recap the first four lecture sessions of CS2040S: Introduction, basic Java (especially Java ArrayList and basic OOP: Java class), basic analysis of algorithm, and to ensure that all students can code a simple Java program using their own computer at home (and/or laptop that he/she intends to use for Week 11 F2F Practical Exam). The first half of the session is generally the 'tutorial' part and the second half of the session is generally the 'hands-on/lab' part. The tutors will control the timings and they don't have to divide the sessions exactly by half. There will be a short break during the transition.

As this is the first session, we will do a quick ice breaking at the start of the session. Your TA is your main contact person of CS2040S related queries this semester. Only contact Prof Halim for questions that you are sure that your TA cannot (or has no privilege to) answer, e.g., questions about class policy, appeal for plagiarism verdict (if caught beyond reasonable doubt and you want to plead your case), etc.

To get the most out of the tutorial part of these sessions, please try out all the questions in the tutorial component and give some answer even if you encounter difficulties in answering some of them. Before, during, or after the tutorial session, don't hesitate to clear up all doubts and questions you might have, with the tutor.

Every week, you will try to solve one medium (or two easy) selected Kattis problem during the 'hands-on/lab' component. The tutors already know the selected Kattis problems for this semester. However, these selected problems will be revealed to you on the spot each week (if you happen to already solve it, then you are free to just leave the session or actually you can stay back to help your peers – you can learn more things by observing how others approached the problem that you have solved – possibly with just one way). Tutor will guide all students to get (near) Accepted solution for

each problem. These problems are not graded but attempting them during the hands-on time (and possibly to fully complete them afterwards) is beneficial to better understand CS2040S material.

The tutorial/lab participation marks returns to encourage class participation. These marks will be given by the tutor **at the end of the semester** using the following guideline:

- 0% if you only attend ≤ 5 out of 10 tutorial/lab sessions (we lose Monday of Week 13, due to Deepavali PH in-lieu),
- 1% for **at most the bottom three** most-passive students (assuming these students attend > 5 tutorial/lab sessions),
- 3% for **at least the top three** most-active students (answering questions when asked by TA – the correctness of your answers are secondary; or even just by asking your own questions to TA before/during/after class/during consultation); in each tutorial group, and
- 2% for the rest.

2 Tutorial 01 Questions

Q1). You are given a simple Java program below:

```
class ListArray<T> { // question 1a
  private int N;
  private Object[] A = new Object[100]; // question 1b
  public ListArray() {
    N = 0;
  }
  @SuppressWarnings("unchecked")
  public T get(int i) {
    return (T) A[i]; // question 1c
  }
  public int indexOf(T v) {
    for (int i = 0; i < N; ++i)
      if (A[i].equals(v))
        return i;
    return -1;
  }
  public void add(T v) { // question 1d
    if (N == 100)
      return;
    A[N++] = v;
  }
  public void add(int i, T v) {
    if (N == 100 || i < 0 || i > N) // question 1e
      return;
    for (int j = N-1; j >= i; --j)
    // for (int j = i; j <= N-1; ++j) // question 1f
      A[j+1] = A[j];
    A[i] = v;
    ++N;
  }
  public void remove(int i) {
    for (int j = i; j < N-1; ++j) // question 1g
      A[j] = A[j+1];
    --N;
  }
```

```
  public void printList() {
    for (int i = 0; i < N; ++i)
      System.out.print((i > 0 ? " " : "") + A[i]); // question 1h
    System.out.println();
  }
  public void sortList() {
    // sort array A, question 1i
  }
}

class ListArrayTest {
  public static void main(String[] args) {
    ListArray<Integer> LA = new ListArray<>();
    LA.add(5);
    LA.add(0, 1);
    LA.add(0, 4);
    LA.add(0, 7);
    LA.add(0, 2); // we should have A = {2, 7, 4, 1, 5} by now
    System.out.println(LA.get(3)); // 1, A[3] = 1
    System.out.println(LA.indexOf(4)); // 2, A[2] = 4
    System.out.println(LA.indexOf(6)); // not found, -1
    LA.remove(1); // we should have A = {2, 4, 1, 5} by now
    System.out.println(LA.indexOf(4)); // 1, A[1] = 4 now
    System.out.println(LA.indexOf(7)); // not found, -1
    LA.printList(); // unsorted
    LA.sortList(); // we should have A = {1, 2, 4, 5} by now
    LA.printList(); // sorted
  }
} // please copy paste the code above, test compile, and run it yourself
```

This code will be revisited soon during discussion of List ADT (read `https://visualgo.net/en/list?slide=2-1` until 2-8). For now, please answer the following sub-questions (see the comments):

(a) What does this line means?

(b) Anything wrong with this line?

(c) Any potential issue with this line?

(d) What is the difference of this 'add' versus the other 'add'?

(e) What does this line means?

(f) What if we use this commented line instead of the line before it? Any potential issue?

(g) Any potential issue with this line?

(h) What does this line means?

(i) Implement this routine using any sorting algorithm that you know!

Analysis/Order of Growth

Q2). What is the *tightest* bound of the following function? $\mathbf{F}(n) = \log(\mathbf{2}^n) + \sqrt{n} + \mathbf{100\,000\,000}$

n log 2

1. $O(n)$
2. $O(n \log n)$
3. $O(n^2)$
4. $O(1)$
5. $O(2^n)$

Q3. What is the *tightest* bound of the following functions below: $\mathbf{F}(n)$, $\mathbf{G}(n)$, and $\mathbf{H}(n)$?
The options:

1. $O(2^n)$
2. $O(n^2)$
3. $O(n \log n)$
4. $O(n)$
5. $O(\log^2 n)$
6. $O(\log n)$
7. $O(1)$
8. none of the above

Q3.a). $\mathbf{F}(n) = n + \frac{1}{2}n + \frac{1}{3}n + \frac{1}{4}n + \frac{1}{5}n + \frac{1}{6}n + \frac{1}{7}n + \frac{1}{8}n + \mathbf{...} + \mathbf{1}$

Q3.b). $\mathbf{G}(n) = n + \frac{1}{2}n + \frac{1}{4}n + \frac{1}{8}n + \frac{1}{16}n + \frac{1}{32}n + \frac{1}{64}n + \frac{1}{128}n + \mathbf{...} + \mathbf{1}$
$\mathbf{G}(n)$ is basically the sum of the reciprocals of powers of two up to n.
To simplify the analysis, assume that n is a powers of two.

Q3.c). $\mathbf{H}(n) = n + \frac{1}{2}n + \frac{1}{3}n + \frac{1}{5}n + \frac{1}{7}n + \frac{1}{11}n + \frac{1}{13}n + \frac{1}{17}n + \mathbf{...} + \mathbf{1}$
$\mathbf{H}(n)$ is basically the sum of the reciprocals of prime numbers up to n.
To simplify the analysis, assume that n is a prime number.
The analysis is very mathematical...
For CS2040S level, just estimate the time complexity of $\mathbf{H}(n)$ w.r.t $\mathbf{F}(n)$ and $\mathbf{G}(n)$.

Hands-on 1

TA will run the second half of this session with one chosen Kattis problem involving List ADT.

Problem Set 1+2

We will end the tutorial with a short PS1 debrief and an overview of PS2.

3 Note

Remember that outside the official tutorial+lab hours, there will be tutor(s) who will stand by at his/their designated (e-)consultation time slot each week on Fri 8-10pm or Sat 10am-12nn (unless they mention some exceptions) to answer CS2040S related queries. This information can be found at `https://www.comp.nus.edu.sg/~stevenha/cs2040s.html`, scroll to 'registration' section. Remember that you do NOT have to be in that tutor's class to join a tutor's consultation slot. It is a free-and-easy unstructured session. Anyone who wants to study together with the tutor or have any CS2040S related question(s) can join.