Assignment 2
Due: Monday, 21 September 2020 23:59 Canberra Time
Grace Period Ends: Tuesday, 22 September 2020 13:00 Canberra Time
Late Penalty: 100%

Notes:

- In this assignment, you need to submit: .
  - A written/typed answer to the questions as a single .pdf file, named A2-studentID.pdf.
  - Source codes as a single file, named A2-studentID.cpp file (or suitable extension for the programming language you use). In this assignment, you can use C/C++/Java. However, support will be provided only for C/C++. Moreover, please pay attention to the following:
    * To ensure we can run your program with no problem, you need to use the template provided during tutorials and test your program on the provided test cases (the template and test cases will be uploaded to the class website on 28 August 2020).
    * We will compile and run your program from command prompt in Linux using the command:
      ```
      > g++ -std=c++11 A2-studentID.cpp -o A2-studentID
      > ./A2-studentID input1.txt > output1.txt
      ```
  - You need to compress all files (the .pdf and .cpp) into a single .zip file named A2-studentID.zip and save this .zip file as draft in Wattle before the due date .
- We provide 13 hours grace period. This means, there will be no penalty if you submit OR save as draft before the grace period ends. However, we will NOT accept assignment after the grace period ends.
- Assignment marking:
  - The total mark you can get in this assignment is 100 points.
  - Whenever explanation/argument/reasoning/derivation is requested, the explanation/derivation is worth 80% of the points you can get for the question.
  - This assignment will contribute 20% to your overall mark.
- Discussion with your colleagues are allowed and encouraged. However, you still need to work on the assignment on your own AND write the names you discussed this assignment with.

[25 pts] Part A

1. [10 pt] Please find a tight asymptotic bound for the following recurrences and provide an explanation.
   (a) [5 pt] $T(n) = 2T(\frac{n}{4}) + \sqrt{n}$ where $T(n) = c$ for $n \leq 2$ and c is a constant positive integer.
   (b) [5 pt] $T(n) = 2T(\frac{n}{4} - 100) + \sqrt{n}$ where $T(n) = c$ for $n \leq 2$ and c is a constant positive integer.
2. [5 pt] Suppose n 6-sided fair dice are rolled independently. What is the expected sum of the outcome of these dice? Please provide the derivation.
3. [10 pt] For each statement below, please identify if the following sorting algorithm is stable, in-place, or both stable and in-place, and provide a short explanation to support your answer. Sometimes, these sorting algorithms can have various modifications. In this assignment, the following algorithms refer to the pseudo-codes discussed in lectures.
   (a) [2.5 pt] Insertion Sort
   (b) [2.5 pt] Merge Sort
   (c) [2.5 pt] Rand Quick Sort
   (d) [2.5 pt] Counting Sort

[40 pts] Part B

4. [10 pt] Your house-mate, Mr S, works as a file clerk in a law firm. As a file clerk, he needs to find files requested by the lawyers fast. Mr S knows that all files in the firm are marked with a unique ID, where each ID is a positive integer number in $[0, m]$. Furthermore, Mr S notices that all requests for files come in the form of a short range of IDs. However, the placement of the files with respect to their IDs are rather random, which makes it difficult for Mr S to find the right files fast. Tired with the increasing demand of requests for files, Mr S asked for your help. Given the above information, you believe you can design a method to re-arrange the files in $\Theta(n+m)$ time (n is the number of documents), so that Mr S can retrieve the files based on a range of IDs in $O(1)$ time. To be more specific, you can assume the range of IDs to be of the form $[a \cdots b]$, where $0 \leq a < b \leq m$ and $b - a \leq c$ for a positive integer constant c. To help Mr S, please provide the following:

   (a) [5 pt] The method for arranging the files along with an explanation that the run-time complexity of this method is indeed $\Theta(n + m)$ time. You can assume moving a single file takes constant time.

   (b) [5 pt] The method for retrieving the files given a range of IDs in $O(1)$ time. Please also provide an explanation of the time complexity.

5. [15 pt]

Algorithm 1 Search(A sorted array of integers A, An integer v)

```
1:  Let s = 1
2:  Let e = A.length
3:  while s ≤ e do
4:      m = ⌊(s+e)/2⌋
5:      if A[m] == v then
6:          Return m
7:      else if A[m] < v then
8:          Let s = m + 1
9:      else
10:         Let e = m − 1
11: Return unsuccessful.
```

Assuming that all elements in A are distinct, the value v is in A and can be any element of A with equal probability, and $n = 2^k$ for a positive integer k, please derive the following in regards to Algorithm 1.

   (a) [5 pt] An asymptotic worst-case upper bound. Please make your bound as tight as possible.

   (b) [10 pt] An asymptotic average-case upper bound. Please make your bound as tight as possible.
   Hint: It might help to think in terms of recurrence tree and consider the number of elements that can be identified at different levels of the tree.

6. [15 pt] A new on-line game, called "Guess What the Monkey is Typing", has just been opened. In this game, a monkey types a random string of characters with length at most n. Let's denote this string as X. For simplicity, the monkey uses a simple keyboard that consists of only 6 characters: Q, W, E, R, T, Y. You can also assume that each character typed by the monkey is independent of the other characters and they are picked uniformly at random from the 6 characters.
   After the monkey finished typing, a player tries to guess the string typed by the monkey. The player can gain additional information about X by privately asking the host of the competition whether a string of his/her choosing, say S, is a prefix of X. A player can ask this question as many times as he/she wants, but must pay \$1 for each question. If the player guesses X correctly, he/she will get $\$4n$, and therefore collect a total of $\$P = \$(4n - k)$ where k is the number of questions asked before the player guesses X correctly.
   Given the above information, please answer the following questions:

   (a) [10 pt] If small win means the player receives a profit P, where $\$0.3n < \$P < \$0.6n$, is there a randomised algorithm such that in the expected worst-case scenario, the player receives a small win? If there is, please provide the algorithm and explanation on the expected worst-case profit. If not, please explain why such an algorithm does not exist.

(b) [5 pt] If large win means the player receives a profit $\$P \geq \$0.6n$, is there a randomised algorithm such that in the expected worst-case scenario, the player receives a large win? If there is, please provide the algorithm and explanation on the expected worst-case profit . If not, please explain why such an algorithm does not exist.

Note: Expected worst case means the average time the algorithm takes on the worst input of a given size. The expectation is computed over the probabilities due to randomisation in the algorithm.

[35 pts] Part C

7. To prepare for the upcoming bushfire season, the government of the Coast of the Southern (CS) Region is setting up an emergency helipad. This helipad will allow helicopters to land safely and quickly to evacuate residents from surrounding towns. Since the creation of a helipad also requires road constructions from the towns to the helipad, the government of CS would like to place the helipad, such that the the total distance between the towns and the helipad is minimised. An illustration is in Fig. 1.

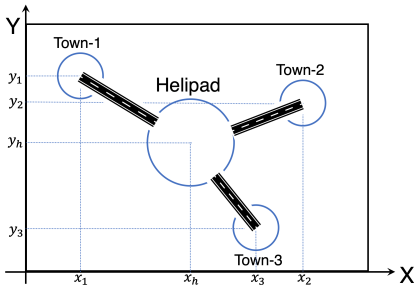

Figure 1: An illustration of a helipad and the towns served.

To be more precise, let's define the position of the helipad as the X-Y coordinate of its center, i.e., (x_h, y_h). If the helipad serves n towns and each town-i is specified as the X-Y coordinate of the town center, i.e., (x_i, y_i), then (x_h, y_h) must be set such that $D = \sum_{i=1}^{n} |x_i - x_h| + |y_i - y_h|$ is minimised. For simplicity, you can assume the X and Y coordinates are integers in [INT_MIN, INT_MAX] and $n <$ INT_MAX, where INT_MIN and INT_MAX are the minimum and maximum values for int data type in C/C++.

Since there is a desire to replicate this type of development in multiple regions, the federal government hires your company to develop a program that can compute the desired position of the helipad (as per the mentioned specification) given the locations of the towns it serves. To this end, please:

(a) [5 pt] Formulate the problem of finding the right position of the helipad as one of finding medians of the X coordinates and the Y coordinates. Please also prove that such a formulation will minimise D.

(b) [10 pt] Design a randomized algorithm with an expected running time of $\Theta(n)$ to find the position (x_h, y_h) that minimises D. Your algorithm can only use at most a constant number of extra storage outside the array that contains the positions of the towns the helipad serves. Please also provide an explanation that the expected time complexity of the algorithm you propose is indeed $\Theta(n)$.
You can assume that the positions of the towns are independent and uniformly distributed, positions of the helipad and towns can overlap, and all X coordinates are distinct and all Y coordinates are also distinct.
Hint: You might want to think of modifying RandQuickSort.

(c) [10 pt] Please implement your algorithm.

***Input to the Program*:** The program will accept two argument, which is the name of the input file. The input file contains $n + 1$ lines, where n is the number of towns served by the helipad. The first line consists of one integer number, n. Each line in the next n lines consists of two integer numbers, separated by a white space. They represent the x and y coordinate of each town. Example:
3
1 2
3 1
4 5
The above input means the helipad is serving three towns, centered at $(1, 2)$, $(3, 1)$, and $(4, 5)$.

***Output of the Program*:** A single line in the standard output, containing two integer numbers separated by a white space. The first number is x_h and the second number is y_h. The output of the example input is:
3 2

Program Marking: If your program compiles and runs, you will get 1 point. We will then run your program on 6 test cases: 2 cases would have up to 500 towns, 2 cases would have 501 – 500,000 towns, and 2 cases would have 500,001 – 250,000,000 towns. For each test case, your program will be given

$\left(\frac{15 \cdot \#towns}{100{,}000} + 0.15\right)$ms CPU time to find a solution. The time limit will be rounded up to 2 decimal digit. You can assume your program will have access to at most 8GB RAM. It will be run as a single thread process on a computer with Intel i7 3.4GHz processor. For each test case that your program solves correctly within the given time and memory limit, you will get 1.5 points.

Tips: Be careful not to use arrays allocated in the program stack. Also, although we will provide several test cases (available in the class website from 28 August 2020), obviously, your program will be tested on different test cases than the examples provided.

(d) [10 pt] Experimental analysis of the time complexity of your algorithm and comparison against the theoretical analysis (7b). You will need to have sufficient data to fit a function well and will need to generate your own test cases for this purpose.