Assignment 1

Due by Friday, May 18 before 11:59pm

Acknowledgments. Acknowledge all the sources you used to complete the assignment. DO NOT COPY! Please read `http://www.student.cs.uwaterloo.ca/~cs341` for general instructions and policies. In the assignments, all logarithms are base 2, unless a different base is explicitly specified.

1 Asymptotics [10 marks]

Prove or disprove each of the following statements.

(a) For any constant $b > 0$, the function $f : n \mapsto 1 + b + b^2 + b^3 + \cdots + b^n$ satisfies

$$f(n) = \begin{cases} \Theta(b^n) & \text{if } b > 1 \\ \Theta(1) & \text{if } b \leq 1. \end{cases}$$

(b) For every pair of functions $f, g : \mathbb{Z}^+ \to \mathbb{R}^{\geq 1}$ that satisfy $f = \Theta(g)$, the functions $F : n \mapsto 2^{f(n)}$ and $G : n \mapsto 2^{g(n)}$ also satisfy $F = \Theta(G)$.

(c) For every pair of functions $f, g : \mathbb{Z}^+ \to \mathbb{R}^{\geq 1}$ that satisfy $f = o(g)$, the functions $F : n \mapsto 2^{f(n)}$ and $G : n \mapsto 2^{g(n)}$ also satisfy $F = o(G)$.

2 Solving recurrences [10 marks]

Solve the following recurrence relations to obtain a closed-form big-Θ expression for $T(n)$. In each question, assume $T(c)$ is bounded by a constant for any small constant c.

(a) $T(n) \leq 9\,T(\frac{n}{3}) + n^2$

(b) $T(n) \leq 4\,T(\frac{n}{4}) + n \log n$

(c) $T(n) \leq T(\frac{n}{4}) + T(\frac{3n}{4}) + n$

(d) $T(n) \leq \sqrt{n} \cdot T(\sqrt{n}) + n$
Hint. The correct expression is somewhere between $\Omega(n)$ and $O(n \log n)$.

3 Testing primality [10 marks]

Analyze the time complexity of the following pseudocode in terms of n using big-Θ notation. For this analysis, each operation on integers (including multiplication and squaring) takes constant time.

Algorithm 1: IsPrime(n)

```
j ← 2;
while j^2 ≤ n do
    k ← 2;
    while j * k ≤ n do
        if j * k = n then
            return False;
        k ← k + 1;
    j ← j + 1;
return True;
```

4 Common sum [10 marks]

In the Common Sum problem, we are given two arrays A and B of length n containing non-negative (not necessarily distinct) integers, and we must determine whether there are indices $i_1, i_2, j_1, j_2 \in \{1, 2, \ldots, n\}$ for which

$$A[i_1] + A[i_2] = B[j_1] + B[j_2].$$

Design an algorithm that solves the Common Sum problem and has time complexity $O(n^2 \log n)$ in the setting where operations on individual integers take constant time.

Your solution must include a description of the algorithm in words, the pseudocode for the algorithm, a proof of its correctness, and an analysis of its time complexity in big-Θ notation.

5 Programming question [10 marks]

Implement the algorithm you obtained for the Common Sum problem in Question 4.

Input and output. The input consists of $n + 1$ lines. The first line contains an integer value $n \geq 1$. The following n lines have two integer values each: $A[i]$ and $B[i]$ for $i = 1, 2, \ldots, n$. We will not be testing whether your program detects input errors. However, our tests will check that your algorithm is fast enough.

The output must have 1 line that contains `TRUE` or `FALSE`.

Sample input 1

```
4
1 2
7 8
5 6
3 4
```

Sample output 1

```
TRUE
```

Sample input 2

```
7
5  16
15  6
25 21
20 31
20 16
10 11
0   1
```

Sample output 2

```
FALSE
```

The valid solution for the first instance is `TRUE` since $A[2] + A[4] = B[1] + B[2] = 10$.

The valid solution for the second instance is `FALSE` because there are no values of $i_1, i_2, j_1, j_2 \in \{1, 2, \ldots, n\}$ for which $A[i_1] + A[i_2] = B[j_1] + B[j_2]$.