Problem Set 2

Data Structures and Algorithms, Fall 2022

Due: September 22, in class.

Problem 1

(a) Recall the MERGESORT algorithm we discussed in class. Suppose you are given an input array A containing n integers, implement MERGESORT to sort the integers in A. You need to give the complete pseudocode of your implementation, including the MERGE procedure. You should also analyze the time complexity and the space complexity of your implementation. (For full credit, your implementation should have $O(n \log n)$ runtime, and use $O(n)$ space beside input.)

(b) Redo part (a), except that this time your input is a (singly) linked-list. In particular, you are given the head pointer and the size of the list. (For full credit, your implementation should have $O(n \log n)$ runtime, and use $O(\log n)$ space beside input.)

Problem 2

You are given an infix expression in which each operator is in $\{+, \times, (,)\}$ and each operand is a single digit positive integer. Write an algorithm to convert it to postfix, and remove parentheses along the way. (For example, given "$(1 + 2) \times 3$", whose value is 9, your algorithm should output "$12 + 3\times$".) You should give a brief overview of your algorithm, then provide pseudocode, and finally discuss its time complexity. (To get full credit, your algorithm should have $O(n)$ time complexity.)

Problem 3

Recall that in class we have introduced an algorithm that multiplies two n-bit binary integers x and y in time $O(n^{\lg 3})$. Call this procedure FASTMULTIPLY(x, y).

(a) We want to convert the decimal integer 10^n (a 1 followed by n zeros) into binary. Figure 1 shows an algorithm (assume n is a power of 2):

PWR2BIN(n)

```
1: if (n == 1) then
2:     return 1010_2
3: else
4:     z ← ???
5:     return FASTMULTIPLY(z, z)
```

Figure 1

Fill in the missing details. Then give a recurrence relation for the running time of the algorithm, and solve the recurrence.

(b) Next, we want to convert any decimal integer x with n digits (where n is a power of 2) into binary. Figure 2 shows an algorithm:

```
DEC2BIN(x)
1: if (n == 1) then
2:     return binary[x]
3: else
4:     Split x into two decimal numbers x_L, x_R with n/2 digits each
5:     return ???
```

Figure 2

Here, $binary[\cdot]$ is a vector that contains the binary representation of all one-digit integers. That is, $binary[0] = 0_2$, $binary[1] = 1_2$, up to $binary[9] = 1001_2$. Assume that a lookup in $binary$ takes $O(1)$ time. Fill in the missing details. Once again, give a recurrence for the running time of the algorithm, and solve it.

Problem 4

(a) Devise a variant of Karatsuba's algorithm that squares any n-digit number in $O(n^{\lg 3})$ time, by reducing to squaring three $\lceil n/2 \rceil$-digit numbers. *(Hint: notice that* $xy = (x^2 + y^2 - (x - y)^2)/2$*.)*

(b) Professor F. Lake claims that it is asymptotically faster to square an n-bit integer than to multiply two n-bit integers. Should you believe him? Prove your answer.

Problem 5

Professor Diogenes has n supposedly identical integrated-circuit chips that in principle are capable of testing each other. The professor's test jig accommodates two chips at a time. When the jig is loaded, each chip tests the other and reports whether it is good or bad. A good chip always reports accurately whether the other chip is good or bad, but the professor cannot trust the answer of a bad chip. Thus, the four possible outcomes of a test are as follows:

Chip A says	Chip B says	Conclusion
B is good	A is good	both are good, or both are bad
B is good	A is bad	at least one is bad
B is bad	A is good	at least one is bad
B is bad	A is bad	at least one is bad

(a) Consider the problem of finding a single good chip from among n chips, assuming that more than $n/2$ of the chips are good. Show that $\lfloor n/2 \rfloor$ pairwise tests are sufficient to reduce the problem to one of nearly half the size.

(b) Show that the good chips can be identified with $O(n)$ pairwise tests, assuming that more than $n/2$ of the chips are good. Give and solve the recurrence that describes the number of tests.

(c) [Bonus Question] Prove that if more than $n/2$ chips are bad, the professor cannot necessarily determine which chips are good using any strategy based on this kind of pairwise test. Assume that the bad chips can conspire to fool the professor.[1]

[1] You are *not* required to solve bonus questions or problems, but correctly solving them will grant you additional credit.