Question 1

Let \( L = \{a_1, a_2, \ldots, a_n\} \) be a list of \( n \) integers.

a) Write an EREW algorithm for computing \( a_1 \cdot a_2 \cdot \cdots \cdot a_n \) using \( \frac{n}{2} \) processors. Analyze the complexity of your algorithm. Is your algorithm cost optimal?

Solution: The idea is to find the product of the numbers using binary approach.

```
Function: Multiplication EREW PRAM
    Model: EREW PRAM with \( \frac{n}{2} \) processors
    Input: \( L[1 : n] \)
    Output: \( \prod_{i}^{n} L[i] \)
    for \( i = 1 \) to \( \log_2 n \) do
        for \( j = 1 \) to \( \frac{n}{2^i} \) in parallel do
            \( L[j] = L[j] \cdot L[\frac{n}{2^i} + j] \)
        end for
    end for
    return \( L[1] \)
end
```

The first for loop runs for \( \log_2 n \) times. The for loop inside is run in parallel and has complexity \( O(1) \).

\[ W(n) = \log_2 n \quad C(n) = \frac{3}{2} \cdot \log_2 n \]

The algorithm is not cost optimal since \( W^*(n) = \Theta(n) \)

b) Write an EREW algorithm for computing \( a_1 \cdot a_2 \cdot \cdots \cdot a_n \) using \( \frac{n}{\log n} \) processors (You may assume that \( \frac{n}{\log n} \) is an integer). Analyze the complexity of your algorithm. Is your algorithm cost optimal?

Solution:
Now each processor will compute the product of $\log_2 n$ numbers sequentially and $\frac{n}{\log_2 n}$ numbers will be obtained. Then the algorithm in part a) will be used to compute the product of $\frac{n}{\log_2 n}$ numbers.

\[ W(n) = \log_2 n + \log\left(\frac{n}{\log_2 n}\right) = \log_2 n \quad C(n) = \frac{n}{\log_2 n} \cdot \log_2 n = n \]

The algorithm is cost optimal since $W^*(n) = C(n)$.

c) Write a CRCW algorithm for testing whether $a_1 \cdot a_2 \cdot \ldots \cdot a_n = 0$. Analyze the complexity of your algorithm. Is your algorithm cost optimal? You will get more or less points depending on the time complexity of your algorithm.

**Solution:**

The product is equal to 0 if one of the integers in the list is equal to 0. Now the model is CRCW and we can use concurrent reads and writes. We will use $n$ processors. Processor $i$ will check if $a_i$ is equal to 0 and write 0 to location say $a_1$ if a 0 is detected. More than one processor may write 0 at the same time but since the model allows concurrent writes, this is not a problem. The problem can be also solved using $p$ processors without specifying $p$.

\[ W(n) = 1 \quad C(n) = n \]

The algorithm is cost optimal since $W^*(n) = C(n)$.

**Question 2**

Suppose that there are $n$ students in a class where $n$ is even. I want to find a student who scored better than half of the students in the class.

a) Determine the lower bound for the worst-case complexity of the problem.

**Solution:**

The problem is equivalent to finding an integer greater than the median given a list of $n$ elements.

If an element is greater than the median, than it is greater than at least $\frac{n}{2}$ of the remaining elements. Assume that all elements in the list are distinct for simplicity. When a comparison based algorithm compares two distinct elements $x$ and $y$, we say that $x$ wins the comparison if $x > y$, otherwise $x$ loses. An element that is greater than the median, say $x$ must win at least $\frac{n}{2}$ comparisons. Let us prove that this is the lower bound for the number of comparisons needed. Suppose for a contradiction that $x$ is the output of the algorithm and $x$ wins $\frac{n}{2} - 1$ comparisons. Equivalently, there are $\frac{n}{2} - 1$ elements which are smaller than $x$. There might be an element $y$ which is greater than $x$ and which is not involved in any comparison. Then, the $\frac{n}{2} - 1$ elements and $x$ are smaller than $y$, which means that $x$ is not greater than the median. We obtain a contradiction.

Therefore, we conclude that the lower bound is equal to $\frac{n}{2}$. In fact this lower bound is optimal. Choosing two numbers and keeping the larger in hand, one can find a number greater than the median after $\frac{n}{2}$ comparisons.
b) Describe a Monte Carlo algorithm for solving the problem which gives a correct answer \( \frac{1}{2} \) of the time. What is the runtime of your algorithm?

**Solution:**

Select an element from the list. With probability \( \frac{1}{2} \), the selected number is greater than the median since

\[
\frac{|\text{numbers greater than the median}|}{|\text{number is the list}|} = \frac{1}{2}.
\]

The runtime of the algorithm is \( O(1) \).

c) Modify your algorithm in part b) to obtain an algorithm which gives a correct answer \( \frac{3}{4} \) of the time. What is the runtime of your algorithm?

**Solution:**

Select two numbers sequentially and choose the larger one. The answer is not correct if both of the selected numbers are smaller than the median, which has probability \( \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4} \). The correct answer is given with probability \( \frac{3}{4} \).

Only 1 comparison is needed and the runtime of the algorithm is \( O(1) \).

d) Now generalize the idea in part c) to describe an algorithm which gives a correct answer \( 1 - \frac{1}{2^k} \) of the time for some constant \( k \). What is the runtime of your algorithm?

**Solution:**

Select \( k \) numbers sequentially and output the largest one. The answer is not correct if all of the selected \( k \) numbers are smaller than the median, which has probability \( \frac{1}{2^k} \). The correct answer is given with probability \( 1 - \frac{1}{2^k} \).

The runtime of the algorithm is still \( O(1) \) since the total number of comparisons needed is \( k - 1 \), which is a constant independent of \( n \).