Q1 (True or False)
1 Point

To argue the correctness of an algorithm it is sufficient to test the algorithm on a large subset of possible inputs.

- ( ) True
- (•) False

Q2 (True or False)
1 Point

There is only one way to prove the correctness of any algorithm.

- ( ) True
- (•) False

Q3 (True or False)
1 Point

An algorithm cannot be correct if it does not terminate on valid input.

- (•) True
- ( ) False

Q4 (True or False)
1 Point

Time is the only resource that one may want to optimise when designing algorithms.

- ( ) True
- (•) False

Q5
2 Points

Consider the following recursive algorithm for checking whether a positive integer is even.

```
isEven(n)
  - if(n=1) return(0)
  - if(n=2) return(1)
  - return(isEven(n-2))
```

Q5.1
1 Point

Suppose we call the function `isEven(10)`. How many subsequent calls to `isEven` are made before our call returns?

- ( ) 8
- ( ) 5
- (•) 4
- ( ) 3

Q5.2
1 Point

Suppose we call the function `isEven(11)`. How many subsequent calls to `isEven` are made before our call returns?

- ( ) 8
- (•) 5
- ( ) 4
- ( ) 3

Q6 (True or False)
1 Point

In any algorithm, there is only one input of a particular size.

- ( ) True
- (•) False

Q7 (True or False)
1 Point

The running time of any algorithm is always the same for all inputs of the same size.

◯ True

◉ False

Q8
9 Points

In the class, we considered the worst-case running time for inputs of a particular size. Similarly, we can define the *best-case* running time as the **smallest** number of operations over inputs of a particular size.

Please review the following algorithms for answering the questions that follow:

- Bubble Sort
- Insertion Sort
- Selection Sort
- Binary Search

Q8.1
1 Point

What is the best-case running time of Insertion Sort?

◯ Logarithmic

◉ Linear

◯ Quadratic

◯ Exponential

Q8.2
1 Point

What is the worst-case running time of Insertion Sort?

◯ Logarithmic

◯ Linear

◉ Quadratic

◯ Exponential

Q8.3
1 Point

Which basic operations are involved in the statement `i = i + 1`?

- [x] Load
- [x] Store
- [ ] Comparison
- [x] Arithmetic

Q8.4
1 Point

What is the best-case running time of Selection Sort?

- ( ) Logarithmic
- ( ) Linear
- (•) Quadratic
- ( ) Exponential

Q8.5
1 Point

What is the worst-case running time of Selection Sort?

- ( ) Logarithmic
- ( ) Linear
- (•) Quadratic
- ( ) Exponential

Q8.6
1 Point

What is the best-case running time of Bubble Sort?

- ○ Logarithmic
- ● Linear
- ○ Quadratic
- ○ Exponential

Q8.7
1 Point

What is the worst-case running time of Bubble Sort?

- ○ Logarithmic
- ○ Linear
- ● Quadratic
- ○ Exponential

Q8.8
1 Point

What is the best-case running time of Binary Search?

- ○ Logarithmic
- ○ Linear
- ○ Quadratic
- ● Constant

Q8.9
1 Point

What is the worst-case running time of Binary Search?

- ● Logarithmic
- ○ Linear
- ○ Quadratic
- ○ Constant

Q9
1 Point

What is the least integer at which $(2n^2 + 3n + 1)$ starts exceeding $(50n + 1)$.

- ○ 22
- ○ 23
- ◉ 24
- ○ 25

Q10
1 Point

Algorithm A1 has a worst-case running time $(2n^2 + 3n + 1)$ and algorithm A2 has a worst-case running time $(50n + 1)$. As per asymptotic worst-case running time analysis, which one is the better algorithm?

- ○ A1
- ◉ A2
- ○ Both are equally good

Q11 (True or False)
1 Point

Algorithm A1 is better than algorithm A2 as per asymptotic worst-case running time analysis. Then this means that A1 has a smaller number of operations than A2 on all inputs.

- ○ True
- ◉ False

Q12 (True or False)
1 Point

Even though algorithm A1 is 10 times faster than A2 on all inputs, the algorithms can be regarded as equally good as per asymptotic worst-case running time analysis.

- ◉ True
- ○ False

Q13 (Big-O notation)
9 Points

Answer the following questions on big-$O/\Omega/\Theta$ notation.

Q13.1 (True or False)
1 Point

$\log n$ is $O(n)$.

- (●) True
- ( ) False

Q13.2 (True or False)
1 Point

$n \log n$ is $O(n)$.

- ( ) True
- (●) False

Q13.3 (True or False)
1 Point

$n \log n$ is $O(n^{1.5})$.

- (●) True
- ( ) False

Q13.4 (True or False)
1 Point

n^2 is $O(2^n)$.

- (●) True
- ( ) False

Q13.5 (True or False)
1 Point

$(2.5)^n$ is $O(2^n)$.

- ( ) True
- (•) False

Q13.6 (True or False)
1 Point

$2^{\log_2 n}$ is $O(n)$.

- (•) True
- ( ) False

Q13.7
1 Point

The running time of algorithm A1 is $\Omega(n)$ and A2 is $\Omega(n^2)$. Which algorithm is better as per asymptotic worst-case running time analysis?

- ( ) A1
- ( ) A2
- ( ) Both are equally good
- (•) Insufficient information

Q13.8 (True or False)
1 Point

An algorithm A has a running time $O(n)$. It is incorrect to say that the running time of A is $O(n^2)$.

- ( ) True
- (•) False

Q13.9 (True or False)
1 Point

An algorithm A has a running time 2^n. It is incorrect to say that the running time of A is $\Omega(n)$.

- ( ) True
- (•) False

Q14 (Bit-arithmetic)

5 Points

As we discussed in the class, whenever we want to do arithmetic on arbitrary long binary numbers, we cannot rely on the architecture-specific operations that are defined on inputs of fixed size. A reasonable input representation in such cases is a bit-array and we count the number of bit operations performed. Answer the questions that follow based on the assumption that the input binary number is given in a bit-array.

Q14.1

1 Point

What is the running time for adding two n-bit binary numbers using the method of long addition?

- (●) $O(n)$
- ( ) $O(\log n)$
- ( ) $O(n^2)$
- ( ) $\Theta(2^n)$

Q14.2

1 Point

What is the running time for multiplying two n-bit binary numbers using the method of long multiplication?

- ( ) $O(n)$
- ( ) $O(\log n)$
- (●) $O(n^2)$
- ( ) $\Theta(2^n)$

Q14.3

1 Point

What is the running time for dividing two n-bit binary numbers using the method of long division?

- ( ) $O(n)$
- ( ) $O(\log n)$
- (●) $O(n^2)$
- ( ) $\Theta(2^n)$

Q14.4 (True or False)
1 Point

There cannot exist an $O(\log n)$ algorithm for adding two n-bit numbers in a single processor system. (*Note that we always assume a single processor system in the course. This question is meant to emphasize this fact.*)

- (selected) True
- False

Q14.5
1 Point

What is the running time of the fastest algorithm that takes as input a binary number in bit-array format and returns whether the number it represents is an even or an odd number?

- $O(n)$
- $O(\log n)$
- $O(n^2)$
- (selected) $O(1)$