Computer Science 4602
Fall 2021
Practice Quiz 3

You have 50 minutes. Answer all of the questions on this exam paper. Circle the letter of the best answer to each multiple-choice problem (marked [MC]), even if no answer is ideal. Make your written answers clear, readable, concise and focused on the problem.

You may use one prepared 8.5×11 sheet of paper during the quiz. **Check your work**.

1. Write a clearly legible

   - **F** to the **left** of each of the following that is **known to be false**,
   - **T** to the **left** of each of the following that is **known to be true**,
   - **U** to the **left** of each of the following that is not currently known to be true or to be false.

   (a) SAT is NP-complete.

   (b) SAT $\notin$ P.

   (c) SAT $\in$ NP.

   (d) P $\neq$ NP.

   (e) P $\subseteq$ NP

   (f) If $A \in$ P and B is nontrivial then $A \leq_p B$.

   (g) If $A \in$ NP and B is nontrivial then $A \leq_p B$.

   (h) $a^* \in$ NP, were a^* is a language consisting of strings of a's of any length.

   (i) $a^* \leq_p$ SAT.

   (j) SAT $\leq_p a^*$.

   (k) If A is NP-complete then $A \leq_p B$ for every language B in NP.

   (l) If A is NP-complete then $B \leq_p A$ for every language B in NP.

   (m) If A is NP-complete then A is computable.

   (n) If A is computable then $A \leq_p$ SAT.

   (o) If A and B are both NP-complete languages then $A \leq_p B$.

2. [MC] The definition of class P is:

   (a) P is the class of all languages that are not in NP.

   (b) P is the class of all languages that have polynomial-time evidence checkers.

   (c) P is the class of all languages for which there exists a deterministic polynomial-time algorithm.

   (d) P is the class of all languages for which a deterministic polynomial-time algorithm is known.

3. [MC] The definition of class NP is:

   (a) NP is the class of all languages for which there exists a polynomial-time evidence checker.

   (b) NP is the class of all languages for which a polynomial-time evidence checker is known.

   (c) NP is the class of all languages that are not in P.

   (d) NP is the class of all languages A where, for every language X, $X \leq_p A$.

4. [MC] Suppose that A and B are languages over alphabet Σ. A function f is a polynomial-time mapping reduction from A to B provided

   (a) $f : A \to B$, f is computable in polynomial time and, for all $x \in A$, $x \in B \iff f(x) \in A$.

   (b) $f : \Sigma^* \to \Sigma^*$, f is computable in polynomial time and, for all $x \in \Sigma^*$, $f(x) \in A \iff f(x) \in B$.

   (c) $f : A \to B$, f is computable in polynomial time and, for all $x \in \Sigma^*$, $x \in B \iff f(x) \in A$.

   (d) $f : \Sigma^* \to \Sigma^*$, f is computable in polynomial time and, for all $x \in \Sigma^*$, $x \in A \iff f(x) \in B$.

5. [MC] The definition of an NP-complete language is: A is NP-complete provided

   (a) A is a hardest problem in NP.

   (b) A not in NP and $X \leq_p A$ for every X that is in NP $-$ P.

   (c) A is not in P and $X \leq_p A$ for every X that is in NP.

   (d) A is in NP and $X \leq_p A$ for every X that is not in P.

   (e) A is in NP and $X \leq_p A$ for every X that is in NP.

6. Suppose that a polynomial-time algorithm is found for the Clique Problem. Which of the following can be concluded from that discovery? **Circle the letter of *every* one that can be concluded.**

   (a) P is not a subset of NP.

   (b) P $\neq$ NP.

   (c) P = NP.

   (d) The Clique Problem is not NP-complete.

   (e) The Clique Problem is in P.

   (f) The Clique Problem is not in NP.

   (g) There is a polynomial-time algorithm for the 3-SAT problem.

   (h) There does not exist a polynomial-time algorithm for the 3-SAT problem.

7. Give an example of a language that is known not to be in P.

8. Give an example of a language that is in NP$-$P, provided P $\neq$ NP.

9. A *proper factor* of positive integer n is a positive integer k that is a factor of n and where $1 \leq k < n$. A positive integer n is *perfect* if and only if the sum of all of the proper factors of n is equal to n. For example, 6 is perfect because the proper factors of 6 are 1, 2 and 3, and $1 + 2 + 3 = 6$.

   The following algorithm determines whether positive integer n is perfect. Is it a polynomial-time algorithm? Briefly explain why or why not.

```
isPerfect(n)
  sum = 0
  for i = 1, ..., n
    if i is a factor of n
       sum = sum + i
    end if
  end for
  if sum == n
    return true
  else
    return false
```

10. The **Hitting Set Problem (HS)** is as follows.

    Input. A positive integer N; a list of sets $x_1, \ldots, x_m$, where $x_i \subseteq \{1, \ldots, N\}$ for $i = 1, \ldots, m$; and a positive integer K.

    Question. Does there exist a set $S \subseteq \{1, \ldots, N\}$ where $|S| \leq K$ and $x_i \cap S \neq \{\}$ for $i = 1, \ldots, m$. That is, S must contain at least one member of each set x_i.

    (a) Give a polynomial-time evidence checker for HS. Be sure that it is correct for every input and that your description is clear and easy to understand.

(b) Give a polynomial-time reduction from the the Vertex Cover problem (VC) to HS. Be sure that the reduction is correct for all possible inputs. Describe the reduction in a clear, readable way. Be sure that I can find your definition of the reduction. Just words describing what the reduction might do are not adequate.

(c) Are the results of parts (a) and (b) of this problem sufficient for you to conclude that HS is NP-complete? Explain why or why not.

(d) Does there exist a polynomial-time reduction from HS to VC? Either argue that there probably is no such reduction or explain why there must exist such a reduction.