Final: Monday, December 18, 8-11am, 2017

A	B	C	D	E	F	G	H	J	K
9am	10am	11am	noon	1pm	1pm	2pm	2pm	3pm	3pm
Rucha	Rucha	Srihita	Shant	Abhishek	Xilin	Shalan	Phillip	Vishal	Phillip
101	101	101	151	151	151	ECE	ECE	ECE	ECE
Armory	Armory	Armory	Loomis	Loomis	Loomis	1002	1002	1002	1002

Name:	
NetID:	
Name on Gradescope:	

- ***Don't panic!***
- If you brought anything except your writing implements, your double-sided **handwritten** (in the original) 8½" × 11" cheat sheet, and your university ID, please put it away for the duration of the exam. In particular, please turn off and put away *all* medically unnecessary electronic devices.
  - Submit your cheat sheet together with your exam. An exam without your cheat sheet attached to it will not be graded.
  - If you are NOT using a cheat sheet, please indicate so in large friendly letters on this page.
- **Best answer.** Choose best possible choice if multiple options seems correct to you – for algorithms, faster is always better.
- Please ask for clarification if any question is unclear.
- **This exam lasts 170 minutes.**
- Fill your answers in the Scantron form using a pencil. We also recommend you circle/mark your answer in the exam booklet.
- Please return ***all*** paper with your answer booklet: your cheat sheet, and all scratch paper. We will **not** return the cheat sheet.
- Do not fill more than one answer on the Scantron form - such answers would not be graded. Also, fill your answer once you are sure of your answer – erasing an answer might make the form unscannable.
- ***Good luck!***

Before doing the exam...

- Fill your name and netid in the back of the Scantron form, and also on the top of this page.
- **Fill in the pattern shown on the right in the Scantron form.**

  This encodes which version of the exam you are taking, so that we can grade it.

91 Ⓐ Ⓑ Ⓒ ● Ⓔ
92 Ⓐ ● Ⓒ Ⓓ Ⓔ
93 ● Ⓑ Ⓒ Ⓓ Ⓔ
94 ● Ⓑ Ⓒ Ⓓ Ⓔ
95 Ⓐ Ⓑ ● Ⓓ Ⓔ
96 Ⓐ Ⓑ ● Ⓓ Ⓔ

9

1. (3 points) Given k sorted arrays $A_1, A_2, \ldots, A_k$ with a total of n numbers stored in them (all numbers are distinct). Given a number x, one can compute the smallest number y, in these arrays, that is larger than x, in (faster is better)

(A) $O(nk)$ time.
(B) $O(n \log n)$ time.
(C) $O(n^2)$ time.
(D) $O(n)$ time.
(E) $O(k \log n)$ time.

---

2. (1 point) All problems in NP are solvable in exponential time. This statement is

(A) True.
(B) False.

---

3. (3 points) For a text file T, let $\langle T \rangle$ denote the string that is the content of T. Consider the language

$$L = \{\langle T \rangle \mid T \text{ is a valid java program that can be compiled}\}.$$

This language is

(A) Decidable.
(B) Context-free.
(C) None of the other answers.
(D) Regular.
(E) Undecidable.

---

4. (3 points) You are given an NFA N with n states (N might have ε-transitions), with the input alphabet being $\Sigma = \{0, 1\}$. Given a binary string $w \in \Sigma^*$ of length m, one can simulate N on a regular computer and decide if N accepts w. Which of the following is correct?

(A) None of the other answers is correct.
(B) This problem can not be done in polynomial time, because it is undecidable.
(C) This can done in $O(n^2 m)$ time.
(D) This can be done in $O(n^m)$ time, and no faster algorithm is possible.
(E) This can be done in $O(2^n m)$ time, and no faster algorithm is possible.

5. (3 points) Consider a CNF formula F with m clauses and n variables. The problem of computing an assignment that satisfies as many clauses as possible, is

(A) NP-Hard.
(B) Can be solved in linear time.
(C) None of the other answers are correct.
(D) Can not be solved in linear time, but can be done in polynomial time.

---

6. (3 points) You are given a directed graph G with n vertices and m edges. Consider the problem of deciding if this graph has a vertex s, from which you can reach all the vertices of G. Solving this problem is

(A) Doable in $O(n(n+m))$ time, and no faster algorithm exists.
(B) NP-Hard by a reduction from 3SAT.
(C) NP-Hard by a reduction to Hamiltonian path/cycle.
(D) NP-Hard by a reduction from Hamiltonian path/cycle.
(E) Doable in $O(n+m)$ time.

---

7. (3 points) Let $L_1, L_2 \subseteq \Sigma^*$ be context-free languages. Then the language $L_1 \cap L_2$ is always context-free.

(A) None of the other answers.
(B) True only if the languages L_1 and L_2 are decidable, and no other answer is correct.
(C) True.
(D) False if the languages L_1 and L_2 are decidable, , and no other answer is correct.
(E) False.

---

8. (3 points) Let G be a DAG with weights on its edges (which can be either positive or negative). The DAG G has n vertices and m edges. Computing the shortest path between two vertices in G can be done in:

(A) This can be solved in $O(n \log n + m)$ time using Dijkstra.
(B) This is NP-Hard.
(C) This can be solved in $O(nm)$ time using Bellman-Ford.
(D) This is not defined if there are negative cycles in the graph. As such, it can not be computed.
(E) This can be done in $O(n+m)$ time.

9. (5 points) Consider an array $A[1 \mathinner{..} n]$ of n numbers. For an interval $I = [i \mathinner{..} j]$ its *discrepancy* is the quantity $s(I) = \sum_{z=i}^{j} A[z]$. Such an interval I is k-*good*, if $s(I) \leq k$, where k is a prespecified parameter. Given A as above, and parameters k and u, a partition of $[1 \mathinner{..} n]$ into ℓ intervals $I_1, \ldots, I_\ell$ is ***(k,u)-excellent*** iff:

(I) For all i, I_i is k-good.
(II) $[1 \mathinner{..} n] = I_1 I_2 \cdots I_\ell$ (the concatenation of $I_1, I_2, \ldots, I_\ell$ is equal to $[1 \mathinner{..} n]$),
(III) $\ell \leq u$.

An algorithm can decide if there is a (k,u)-excellent partition of A in (faster is better):

(A) $O(n^4)$.
(B) $O(n^3 k)$ time.
(C) $O(n^3 u)$ time.
(D) $O(n^2 u)$ time.
(E) $O(n^2 k)$ time.

---

10. (2 points) Consider the following decision problem: Given a directed graph G, and two vertices u, v in G, is there a path from u to v in G?
This problem has a polynomial length certificate and polynomial time certifier. This claim is

(A) True.
(B) False.

---

11. (2 points) Consider the language

$$L = \left\{ 1^i 2^j 3^k \;\middle|\; i, j, k \geq 0, \text{ and } j \text{ is divisible by } p_1, p_2, \ldots, p_{100} \right\},$$

where p_j is the jth smallest prime number, for $j = 1, \ldots, 100$ (i.e., $p_1 = 2, p_3 = 3, \ldots, p_{100} = 541$). This language is

(A) Regular.
(B) Context-free.
(C) Undecidable.
(D) Finite.
(E) Decidable.

---

12. (3 points) You are given a graph G, and vertices u and v. Such a pair vertices is ***robustly connected***, if they remain connected, even if we remove any single vertex in G (except for u and v, naturally). Consider the problem of deciding if u and v are robustly connected.

(A) The problem is NP-Hard.
(B) This problem can be solved in polynomial time.

13. (3 points) If a problem is NP-Complete, then it can also be undecidable. This statement is

(A) False.
(B) None of the other answers.
(C) True.
(D) True if $P = NP$.
(E) False if $P = NP$.

---

14. (3 points) Let $\mathcal{PC}$ be the class of all decision problems, for which there is a polynomial time certifier that works in polynomial time, and furthermore, for an input of length n, if it is a YES instance, then there is a certificate that is a binary string of length $n^{O(1)}$. We have that:

(A) $\mathcal{PC}$ contains some NP-Complete problems, but not all of them.
(B) NP $\subsetneq \mathcal{PC}$.
(C) None of the other answers is correct.
(D) NP $= \mathcal{PC}$.
(E) All the problems in $\mathcal{PC}$ can be solved in polynomial time.

---

15. (3 points) Give a CNF formula F with n variables, and m clauses, where every clause has exactly two literals (reminder: a literal is either a variable or its negation). Then, one can compute a satisfying assignment to F in:

(A) $O(n + m)$ time.
(B) This is Satisfiability and it can not be solved in polynomial time unless $P = NP$.
(C) $O(n \log n + m)$ time.
(D) $O(n^2 + m^2)$ time.
(E) $O(2^n - 2^m)$ time.

---

16. (3 points) Given two DFAs N_1 and N_2 with n_1 and n_2 states, respectively. Then there is a DFA M that accepts the language $L(N_1) \oplus L(N_2)$, where $A \oplus B = (A \setminus B) \cup (B \setminus A)$.

(A) False.
(B) True, and the number of states of M is at most $n_1 n_2$.
(C) None of the other answers is correct.
(D) True, and the number of states of M is at most $2^{n_1} 2^{n_2}$.
(E) True, and the number of states of M is at most $O(n_1 + n_2)$.

17. (3 points) Consider the problem of checking if a graph has k vertices that are all adjacent to each other. This problem can be solved in

(A) Polynomial time.
(B) None of the other answers are correct.
(C) It is NP-Complete, so it can not be solved efficiently.
(D) Maybe polynomial time – we do not know. Currently fastest algorithm known takes exponential time.

18. (3 points) Given an undirected graph G, with n vertices and m edges, consider the decision problem of determining if the vertices of G can be colored (legally) by n colors (i.e., no adjacent pair of vertices have the same color). This problem is:

(A) Solvable in $O(1)$ time.
(B) Can be solved in polynomial time.
(C) Undecidable.
(D) Solvable in $O(n+m)$ time.
(E) NP-Complete.

19. (2 points) Consider a regular expression r that is of length m (i.e., writing r down requires m characters). Then, there is an equivalent NFA N with at most

(A) 10 states.
(B) $m^{O(m^2)}$ states.
(C) None of the other answers holds.
(D) $O(m)$ states.
(E) $2^{O(m)}$ states.

20. (3 points) Let $P_1, \ldots, P_{k+1}$ be $k+1$ decision problems. Consider a sequence of k polynomial reductions $R_1, \ldots, R_k$, where R_i works in quadratic time in its input size, and is a reduction from P_i to P_{i+1}. As such, there is a reduction from P_1 to P_{k+1} and its running time is

(A) $O(n^{2k})$
(B) $O(k^2 n^2)$
(C) $O(kn^2)$
(D) $O(2^k n^2)$
(E) $O\left(n^{2^k}\right)$

---

21. (2 points) Consider a Turing machine (i.e., program) M that accepts an input $w \in \Sigma^*$ if and only if there is a DFA that accepts w. Then the language of $L(M)$ is

(A) Σ^*.
(B) undecidable.
(C) not well defined.
(D) context-free.
(E) finite.

---

22. (3 points) Given an undirected graph G with n vertices and m edges, and a number k, deciding if G has a spanning tree with maximum degree k is

(A) Can be done in $O((n+m)\log n)$ time, and there is no faster algorithm.
(B) Can be done in $O(n \log n + m)$ time, and there is no faster algorithm.
(C) NP-Complete.
(D) Can be done in polynomial time.
(E) Can be done in $O(n+m)$ time.

---

23. (3 points) For the language $L = \{0^n 1^n \mid n \geq 0\}$, we have

(A) None of the sets suggested are fooling sets.
(B) $F = \left\{0^i \mid i \geq 0\right\}$ is a fooling set for L.
(C) All of the sets suggested are fooling sets.
(D) $F = \left\{0^i 1^i \mid i \geq 0\right\}$ is a fooling set for L.
(E) $F = \left\{0^i 1^j \mid i < j\right\}$ is a fooling set for L.

---

24. (3 points) For the following recurrence (evaluated from top to bottom in this order):

$$g(i,j) = \begin{cases} 1 & i < 0 \text{ or } j < 0 \\ g(\lfloor i/2 \rfloor, j) + 1 & i > j \\ g(i-1,j) + g(i-2,j-1) + g(i-3,j-2) & \text{otherwise.} \end{cases}$$

Assume that every arithmetic operation takes constant time (even if the numbers involved are large). Computing $g(n,n)$ can be done in (faster is better):

(A) $O(2^n)$.
(B) $O(n \log n)$ time.
(C) $O(n^3)$ time, using dynamic programming.
(D) $O(n^2)$ time, using dynamic programming.
(E) $O(n)$ time, by recursion.

25. (2 points) Let G be a directed graph with n vertices and m edges. Deciding if two vertices u, v are in the same connected component of G can be done in

(A) $O(nm)$ time using Bellman-Ford.
(B) None of the other answers is correct.
(C) $O(n+m)$ time.
(D) $O(n \log n + m)$ time.
(E) only exponential time since this problem is NP-COMPLETE.

26. (3 points) Consider the recurrence $f(n) = f\Bigl(\lfloor (2/3)n \rfloor\Bigr) + f\Bigl(\lfloor (1/3)n \rfloor\Bigr) + O(n)$, where $f(n) = O(1)$ if $n < 10$. The solution to this recurrence is

(A) $O(1)$.
(B) None of the above.
(C) $O(n)$.
(D) $O(n^2)$.
(E) $O(n \log n)$.

27. (2 points) You are given a set $\mathcal{I} = \{I_1, I_2, \ldots, I_n\}$ of n weighted intervals on the real line. Consider the problem of computing a value $x \in \mathbb{R}$, that maximizes the total weight of the intervals of $\mathcal{I}$ containing x. This problem:

(A) Can be done in linear time.
(B) NP-HARD.
(C) Undecidable.
(D) Can be done in polynomial time.
(E) NP-COMPLETE.

28. (2 points) You are given an unsorted set Z of n numbers. Deciding if there are two numbers x and y in Z such that $x = 1 - y$ can be solved in (faster is better):

(A) $O(n^2)$ time.
(B) $O(n^2 \log n)$ time.
(C) $O(n^{3/2})$ time.
(D) $O(n)$ time.
(E) $O(n \log n)$ time.

29. (3 points) You are given a graph G with n vertices and $m = O(n)$ edges, and with weights on the edges. In addition, you are given the MST tree T of G.
Next, you are informed that the price of some edge e in the MST T had changed from its current cost (either increased or decreased), to a new cost α. Deciding if T is still the MST of the graph with the updated weights can be done in (faster is better):

(A) $O(\log n)$ time, after preprocessing the graph in $O(n)$ time.
(B) $O(nm)$ time algorithm, and no faster algorithm is possible.
(C) $O(1)$ time.
(D) $O(n)$ time.
(E) $O(n \log n + m)$ time.

---

30. (2 points) For a word $w = w_1 w_2 \ldots w_m$, with $w_i \in \{0, 1\}^*$, let $w^Z = \overline{w_1} \ldots \overline{w_m}$, where $\overline{0} = 1$ and $\overline{1} = 0$. If a language L is a regular language, then the language $L^Z = \left\{ w^Z \;\middle|\; w \in L \right\}$ is regular.

(A) True
(B) False

---

31. (3 points) Given an array $C[1 \,..\, n]$ with n real numbers (C is not sorted), consider the problem computing and printing out the largest $\lfloor \log^3 n \rfloor$ numbers in C – the numbers should be output in sorted order. This can be done in

(A) $O(n)$ time, and no faster algorithm is possible.
(B) $O(n \log n)$ time, and no faster algorithm is possible.
(C) $O(\log^4 n)$ time, and no faster algorithm is possible.
(D) $O(\log^3 n)$ time, and no faster algorithm is possible.

---

32. (3 points) Let B be the problem of deciding if the shortest path in a graph between two given vertices is smaller than some parameter k (where the weights on the edges of the graph are positive).
Let C be the problem of deciding if a given instance of 2SAT formula is satisfiable. Pick the correct answer out of the following:

(A) There is no relation between the two problems, and no reduction is possible.
(B) There is a polynomial time reduction from C to B, but only if $P \neq NP$..
(C) There is a polynomial time reduction from B to C.
(D) None of the other answers is correct.
(E) There is a polynomial time reduction from B to C, but only if $P = NP$.

33. (3 points) Given a directed graph G with n vertices and m edges, consider the problem of deciding if there is a simple path that visits at least half of the vertices of G.

(A) At least two of the other answers are correct.
(B) Can be solved in $O(nm)$ time, and no faster algorithm is possible.
(C) NP-Hard.
(D) NP-Complete.
(E) Can be solved in $O(n+m)$ time.

---

34. (3 points) You are given a directed graph G with n vertices, m edges, and positive weights on the vertices (but not on the edges). In addition, you are given two vertices u and v, and a number t. The *weight* of a path π is the total weight of the vertices of π. Consider the problem of computing a (simple) path σ connecting a vertex u to a vertex v, such that the weight of σ is at least t. This problem is

(A) Solvable in $O(n+m)$ time.
(B) Undecidable.
(C) Solvable in $O(n \log n + m)$ time.
(D) Polynomially equivalent to Eulerian cycle.
(E) NP-Hard.

---

35. (3 points) The number of undecidable languages is

(A) uncountable.
(B) countable.
(C) $2^{\mathbb{R}} = \aleph_2$.
(D) None of the other answers are correct.
(E) undecidable.

---

36. (3 points) You are given two algorithms B_Y and B_N. Both algorithms read an undirected graph G and a number k. If G has a vertex cover of size $\leq k$, then B_Y would stop (in polynomial time!) and output YES (if there is no such vertex cover then B_Y might run forever). Similarly, if G does not have a vertex cover of size $\leq k$, then the algorithm B_N would stop in polynomial time, and output NO (if there is such a vertex cover then B_N might run forever).
In such a scenario:

(A) One can in polynomial time output if G has a an vertex cover of size $\leq k$ or not.
(B) This would imply that $P \neq NP$.
(C) At least two of the other answers are correct.
(D) Impossible since $P \neq NP$.
(E) This would imply that $P = NP$.