QUIZ 3: 60 Minutes

Last Name: Solutions

First Name: ______

RIN: ______

Section: ______

Answer **ALL** questions.

NO COLLABORATION or electronic devices. Any violations result in an F.

NO questions allowed during the test. Interpret and do the best you can.

GOOD LUCK!

Circle at most one answer per question.

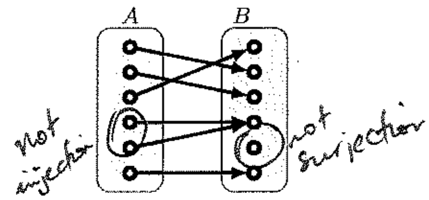
10 points for each correct answer

You **MUST** show **CORRECT** work to get full credit.

When in doubt, TINKER.

Total
200

1. Which describes the function on the right that maps A to B.



A

[A] f is not an injection (1-to-1) and f is not a surjection (onto).

[B] f is an injection (1-to-1) and f is not a surjection (onto).

[C] f is not an injection (1-to-1) and f is a surjection (onto).

[D] f is an injection (1-to-1) and f is a surjection (onto).

[E] None of the above.

2. A set S contains all the distinct functions which map $\{0, 1\}$ to $\mathbb{N}$. What is the cardinality of S?

[A] 0.

[B] 1.

[C] Bigger than 1 but finite.

D

[D] The same as $|\mathbb{N}|$.

[E] Strictly larger than $|\mathbb{N}|$.

$f: \begin{matrix} 0 \to i \\ 1 \to j \end{matrix}$ all possible pairs (i, j)

$\mathbb{N} \times \mathbb{N}$ is countable.

$|\mathbb{N} \times \mathbb{N}| = |\mathbb{N}|$

3. A set S contains all the distinct functions which map $\mathbb{N}$ to $\{0, 1\}$. What is the cardinality of S?

[A] 0.

[B] 1.

[C] Bigger than 1 but finite.

[D] The same as $|\mathbb{N}|$.

E

[E] Strictly larger than $|\mathbb{N}|$.

$f: \mathbb{N} \to \{0, 1\}$ specify f by an infinite binary string

$\to$ |infinite binary strings| = uncountable.

4. What is a computing problem?

[A] A person who knows how to write a program in python.

[B] A machine that transitions between states.

[C] A rule for deciding if a string belongs to a set.

D

[D] Any set of finite binary strings.

[E] A Turing Machine.

Computing Problem $\to$ Decision Problem $\to$ Language.

set of finite binary strings.

5. Which set is *not* countable, i.e., has a cardinality strictly larger than $|\mathbb{N}|$?

[A] $\mathbb{Q}$, the rational numbers. ✓

[B] All distinct finite binary strings. ✓

[C] All possible Turing Machines. ✓

D

[D] All possible computing problems. ✗

[E] They are all countable.

B

6. The language $\mathcal{L} = \{11, 111\}^*$ (Kleene star). Which string is not in $\mathcal{L}$?

[A] ε. ✓

[B] 1. ✗

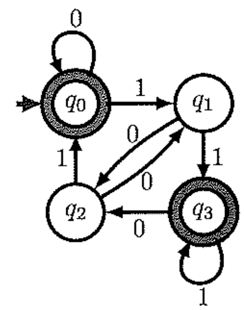
[C] 1111. 11 · 11

[D] 11111. 11 · 111

[E] They are all in $\mathcal{L}$.

C

7. What is the final resting state for the DFA on input 110110.



$q_0 \to q_1 \to q_3 \to q_2 \to q_0 \to q_1 \to q_2$.

[A] q_0.

[B] q_1.

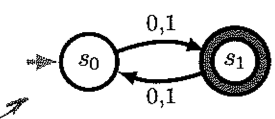
[C] q_2.

[D] q_3.

[E] None of the above

D

8. Let $\Sigma = \{0, 1\}$. Which regular expression is the problem solved by the DFA.



[A] Σ^*. ✗

[B] $\Sigma \bullet \Sigma \bullet \Sigma$. ← only 3-bit strings.

[C] $(\Sigma \bullet \Sigma) \bullet \Sigma^*$. ← contains 11

[D] $\Sigma \bullet (\Sigma \bullet \Sigma)^*$. ✓

[E] None of the above.

$\Sigma \cdot \underbrace{(\Sigma \cdot \Sigma)^*}_{}$ even.

strings with an odd # bits.

D

9. How many 5 bit strings are in the (YES)-set of the DFA in problem 8.

[A] 4.

[B] 8.

[C] 16.

[D] 32.

[E] None of the above.

All 5-bit strings accepted (as odd #bits)

$\therefore 2^5 = 32$.

D

10. Which computing problem *cannot* be solved by a DFA (deterministic finite automata)?

[A] $\mathcal{L} = \{$strings with no 1s$\}$. ✓

[B] $\mathcal{L} = \{$strings with an odd number of 1s$\}$. ✓

[C] $\mathcal{L} = \{$strings that are *not* 1111$\}$ ✓ Can solve 1111 (finite) → complement

[D] $\mathcal{L} = \{$strings with more 1s than 0s$\}$. ← needs memory to count ✗.

[E] Each problem above can be solved by a DFA.

11. The main limitation of the DFA which prevents it from solving $\mathcal{L} = \{0^{\bullet n}1^{\bullet n+3} | n \geq 0\}$ is:

No external memory.

D

- $\boxed{\text{A}}$ The DFA can't have more than one yes-state. ✗
- $\boxed{\text{B}}$ The input string can be arbitrarily long. ✗
- $\boxed{\text{C}}$ The DFA can go into an infinite loop. ✗
- $\boxed{\text{D}}$ The DFA cannot remember how many 0s have gone by because it has only finitely many states.
- $\boxed{\text{E}}$ None of the above, because a DFA can solve $\mathcal{L}$.

12. Which string cannot be generated by the CGF shown?

$1: S \to \varepsilon \mid A \mid B$
$2: A \to 0 \mid 0B$
$3: B \to 1 \mid 1A$

D

- $\boxed{\text{A}}$ ε ✓
- $\boxed{\text{B}}$ 010 ✓
- $\boxed{\text{C}}$ 101 ✓
- $\boxed{\text{D}}$ 011 ✗
- $\boxed{\text{E}}$ They can all be generated.

1 10 101

$B \to 1A \to 10B \to 101A$

$S \to A \to 0B \to 01A \to 010B$

ε 0 01 010

Strings with alternating 0,1.

13. Which CFG generates all strings with an even number of bits, including ε.

D

- $\boxed{\text{A}}$ $S \to \varepsilon \mid SS$ ✗ can only generate ε
- $\boxed{\text{B}}$ $S \to \varepsilon \mid 0 \mid 1 \mid SS$ ✗ $S \to 0$ odd # bits
- $\boxed{\text{C}}$ $S \to \varepsilon \mid 01S$ ✗ cannot generate 10
- $\boxed{\text{D}}$ $S \to \varepsilon \mid 00S \mid 01S \mid 10S \mid 11S$ ✓ $S \to \varepsilon$ or any 2 bits + even ✓
- $\boxed{\text{E}}$ None of the above.

14. Which comparison between DFAs and CFGs is correct? CFG is more powerful than DFA

B

- $\boxed{\text{A}}$ A DFA can solve language $\mathcal{L}$ if and only if a CFG can generate language $\mathcal{L}$. ✗
- $\boxed{\text{B}}$ If a DFA can solve language $\mathcal{L}$, then a CFG can generate language $\mathcal{L}$. ✓
- $\boxed{\text{C}}$ If a CFG can generate language $\mathcal{L}$, then a DFA can solve language $\mathcal{L}$. ✗ 0^n1^n
- $\boxed{\text{D}}$ There is some language $\mathcal{L}$ which a DFA can solve, but no CFG can generate that language $\mathcal{L}$. ✗ CFG is more powerful than DFA.
- $\boxed{\text{E}}$ None of the above.

15. In the theory of computing, we define computing problems and algorithms as:

Computing Problem: Language

Algorithm: TM-decider.

E

- $\boxed{\text{A}}$ A computing problem is a string. An algorithm is a recognizer. ✗
- $\boxed{\text{B}}$ A computing problem is a set of finite binary strings. ✓ An algorithm is a recognizer. ✗
- $\boxed{\text{C}}$ A computing problem is a Turing Machine. ✗ An algorithm is a decider.
- $\boxed{\text{D}}$ A computing problem is a set of finite binary strings. ✓ An algorithm is a person. ✗
- $\boxed{\text{E}}$ A computing problem is a set of finite binary strings. ✓ An algorithm is a decider. ✓

16. Why do we prefer a Turing machine decider over a Turing machine recognizer? Decider always halts.

D

- $\boxed{A}$ Because there are some yes sets that are accepted by a decider but not a recognizer. ✗
- $\boxed{B}$ Because a decider can write to the tape, but a recognizer cannot. ✗
- $\boxed{C}$ Because a decider has a finite number of states, but a recognizer has infinitely many states. ✗
- $\boxed{D}$ Because any useful algorithm should always halt giving an answer. ✓
- $\boxed{E}$ We don't prefer one over the other because both are the same thing. ✗

17. Consider the computing problem $\mathcal{L} = \{0^{\bullet m}1^{\bullet n}0^{\bullet k} \mid m, n, \geq 0 \text{ and } n = m + k\}$. Which claim is not true?

E

- $\boxed{A}$ A DFA cannot solve $\mathcal{L}$. ✓ Set $k=0$ gives $0^m 1^m$ and DFA cannot solve.
- $\boxed{B}$ A DFA with an external top-access stack memory can solve $\mathcal{L}$. ✓ Yes: push 0's; pop with 1's; push additional 1's; Pop with 0's.
- $\boxed{C}$ A CFG can generate $\mathcal{L}$ ✓ $0^m 1^m \cdot 1^k 0^k$ → CFG generates $0^m 1^m$ and $1^k 0^k$ ∴ can generate concatenation.
- $\boxed{D}$ A Turing machine decider can solve $\mathcal{L}$. ✓ → TM is stronger than CFG
- $\boxed{E}$ None of the above. All true.

18. Which problem is not solvable by an algorithm?

D

- $\boxed{A}$ $\mathcal{L} = \{\langle M \rangle \mid M \text{ is a valid Turing Machine.}\}$ ← compiler ✓
- $\boxed{B}$ $\mathcal{L} = \{0^{\bullet n} \mid n \geq 0\}$. ✓
- $\boxed{C}$ $\mathcal{L} = \{0^{\bullet 2^n} \mid n \geq 0\}$. ✓
- $\boxed{D}$ Determining if any given python program correctly says if an input n is prime or not. ✗ Program Verification
- $\boxed{E}$ None of the above.

19. Problem $\mathcal{L}_A$ is reducible to $\mathcal{L}_B$, that is $\mathcal{L}_A \leq_{\text{R}} \mathcal{L}_B$. We know that $\mathcal{L}_B$ is decidable. Which is true?

C

- $\boxed{A}$ $\mathcal{L}_A$ must be undecidable. ✗
- $\boxed{B}$ $\mathcal{L}_A$ can be undecidable. ✗
- $\boxed{C}$ $\mathcal{L}_A$ must be decidable. ✓
- $\boxed{D}$ $\mathcal{L}_A$ must be finite. Not Necessarily.
- $\boxed{E}$ None of the above.

$\mathcal{L}_A < \mathcal{L}_B$ $\quad$ $\mathcal{L}_B$ decidable.

∴ $\mathcal{L}_A$ is easier → decidable.

20. Let $\mathcal{M}$ be the set of all possible Turing Machines. Which statement is not true?

D

- $\boxed{A}$ Every Turing Machine in $\mathcal{M}$ can be uniquely encoded into a finite binary string. → its description
- $\boxed{B}$ All Turing Machines in $\mathcal{M}$ can be listed: $\{\langle M_1 \rangle, \langle M_2 \rangle, \langle M_3 \rangle, \langle M_4 \rangle, \ldots, \}$. → Yes, countable
- $\boxed{C}$ $\mathcal{M}$ is countable. → Yes → See
- $\boxed{D}$ Given any computing problem $\mathcal{L}$, there is a Turing Machine in $\mathcal{M}$ which solves $\mathcal{L}$. → No: Computing problems are uncountable.
- $\boxed{E}$ All of the above are true.