## True/False Questions

$\mathcal{P}$ means $\mathcal{P}$–TIME
$\mathcal{NP}$ means $\mathcal{NP}$–TIME
$\mathcal{RE}$ means recursively enumerable
$\mathcal{NC}$ means Nick's class.
If $\mathcal{C}$ is any class of languages, co-$\mathcal{C}$ means the class of all languages which are complements of languages in $\mathcal{C}$.
A *binary language* is a language over the binary alphabet $\{0, 1\}$.
A *recursive* function is any function which can be computed by a machine.
A *recursive* real number is any real number whose $n^{\text{th}}$ decimal digit is a recursive function of $n$.
A *fraction* is a string consisting of a numeral, followed by a slash, followed by another numeral.

1. True or False. T = true, F = false, and O = open, meaning that the answer is not known science at this time. In the questions below, $\mathcal{P}$ and $\mathcal{NP}$ denote $\mathcal{P}$-TIME and $\mathcal{NP}$-TIME, respectively.

   (i) **T** The problem of whether a given string is generated by a given context-free grammar is decidable.

   (ii) **T** If $G$ is a context-free grammar, the question of whether $L(G) = \emptyset$ is decidable.

   (iii) **T** The Kleene closure of any $\mathcal{NP}$ langauge is $\mathcal{NP}$

   (iv) **T** The language $\{a^n b^n c^n d^n \mid n \geq 0\}$ is recursive.

   (v) **T** The language $\{a^n b^n c^n \mid n \geq 0\}$ is in the class $\mathcal{P}$-TIME.

   (vi) **O** There exists a polynomial time algorithm which finds the factors of any positive integer, where the input is given as a binary numeral.

   (vii) **F** Every undecidable problem is $\mathcal{NP}$-complete.

   (viii) **F** Every problem that can be mathematically defined has an algorithmic solution.

   (ix) **F** The intersection of two undecidable languages is always undecidable.

   (x) **T** Every $\mathcal{NP}$ language is decidable.

   (xi) **F** If $L_1$ and $L_2$ are $\mathcal{NP}$-complete languages and $L_1 \cap L_2$ is not empty, then $L_1 \cap L_2$ must be $\mathcal{NP}$-complete.

   (xii) **O** There exists a $\mathcal{P}$-TIME algorithm which finds a maximum independent set in any graph $G$.

   (xiii) **T** There exists a $\mathcal{P}$-TIME algorithm which finds a maximum independent set in any acyclic graph $G$.

   (xiv) **O** $\mathcal{NC} = \mathcal{P}$.

   (xv) **O** $\mathcal{P} = \mathcal{NP}$.

   (xvi) **O** $\mathcal{NP} = \mathcal{P}$-SPACE

(xvii) **O** $\mathcal{P}$-SPACE = EXP-TIME

(xviii) **O** EXP-TIME = EXP-SPACE

(xix) **T** The traveling salesman problem (TSP) is known to be $\mathcal{NP}$-complete.

(xx) **T** The language consisting of all satisfiable Boolean expressions is known to be $\mathcal{NP}$-complete.

(xxi) **T** The Boolean Circuit Problem is in $\mathcal{P}$.

(xxii) **O** The Boolean Circuit Problem is in $\mathcal{NC}$.

(xxiii) **T** 2-SAT is $\mathcal{P}$-TIME.

(xxiv) **O** 3-SAT is $\mathcal{P}$-TIME.

(xxv) **T** Primality is $\mathcal{P}$-TIME.

(xxvi) **F** There is a $\mathcal{P}$-TIME reduction of the halting problem to 3-SAT.

(xxvii) **T** Every context-free language is in $\mathcal{NC}$.

(xxviii) **T** Addition of binary numerals is in $\mathcal{NC}$.

(xxix) **F** Every language generated by a general grammar is recursive.

(xxx) **F** The problem of whether two given context-free grammars generate the same language is decidable.

(xxxi) **T** The language of all fractions (using base 10 numeration) whose values are less than $\pi$ is decidable.

(xxxii) **T** For any two languages $L_1$ and $L_2$, if $L_1$ is undecidable and there is a recursive reduction of $L_1$ to $L_2$, then $L_2$ must be undecidable.

(xxxiii) **F** For any two languages $L_1$ and $L_2$, if $L_2$ is undecidable and there is a recursive reduction of $L_1$ to $L_2$, then $L_1$ must be undecidable.

(xxxiv) **F** If $P$ is a mathematical proposition that can be written using a string of length $n$, and $P$ has a proof, then $P$ must have a proof whose length is $O(2^{2^n})$.

(xxxv) **T** If $L$ is any $\mathcal{NP}$ language, there must be a $\mathcal{P}$–TIME reduction of $L$ to the partition problem.

(xxxvi) **O** If $L$ is $\mathcal{NP}$ and also co-$\mathcal{NP}$, then $L$ must be $\mathcal{P}$.

(xxxvii) **T** A language is $\mathcal{RE}$ if and only if it is generated by a grammar.

(xxxviii) **T** If $L$ is $\mathcal{RE}$ and also co-$\mathcal{RE}$, then $L$ must be decidable.

(xxxix) **T** Every language is enumerable.

(xl) **F** If a language $L$ is undecidable, then there can be no machine that enumerates $L$.

(xli) **F** There exists a mathematical proposition which is true, but can be neither proved nor disproved.

(xlii) **T** There is a non-recursive function which grows faster than any recursive function.

(xliii) **T** There exists a machine that runs forever and outputs the string of decimal digits of $\pi$ (the well-known ratio of the circumference of a circle to its diameter).

(xliv) **F** For every real number $x$, there exists a machine that runs forever and outputs the string of decimal digits of $x$.

(xlv) **T** Every subset of any enumerable set is enumerable.

(xlvi) **T** There is a polynomial time reduction of the subset sum problem to the binary numeral factorization problem.

(xlvii) **F** For any real number $x$, the set of fractions whose values are less than $x$ is $\mathcal{RE}$.

(xlviii) **T** For any recursive real number $x$, the set of fractions whose values are less than $x$ is recursive (i.e., decidable).

(xlix) **T** The membership problem for any CFL is in the class $\mathcal{P}$-TIME.

(l) **F** 2-SAT is known to be $\mathcal{NP}$-complete.

(li) **T** The complement of any $\mathcal{P}$-TIME language is $\mathcal{P}$-TIME.

(lii) **T** The complement of any $\mathcal{P}$–SPACE language is $\mathcal{P}$–SPACE.

(liii) **T** The complement of any decidable language is decidable.

(liv) **F** The complement of any undecidable language is undecidable.

(lv) **F** The complement of any $\mathcal{RE}$ language is $\mathcal{RE}$.

The *jigsaw puzzle problem* is, given a set of various polygons, and given a rectangular table, is it possibe to assemble those polygons to exactly cover the table?

The *furniture mover's problem* is, given a room with a door, and given a set of objects outside the room, it is possible to move all the objects into the room through the door?

(lvi) **T** The jigsaw puzzle problem is known to be $\mathcal{NP}$ complete.

(lvii) **F** The jigsaw puzzle problem is known to be $\mathcal{P}$–SPACE complete.

(lviii) **F** The furniture mover's problem is known to be $\mathcal{NP}$ complete.

(lix) **T** The furniture mover's problem is known to be $\mathcal{P}$–SPACE complete.

(lx) **T** The complement of any recursive language is recursive.

(lxi) **T** For any infinite countable sets $A$ and $B$, there is a 1-1 correspondence between $A$ and $B$.

(lxii) **F** The set of all binary languages is countable.

(lxiii) **T** A language $L$ is recursively enumerable if and only if there is a machine which accepts $L$.

(lxiv) **T** Every $\mathcal{NP}$ language is reducible to the independent set problem in polynomial time.

(lxv) **T** If a Boolean expression is satisfiable, there is a polynomial time proof that it is satisfiable.

(lxvi) **T** The general sliding block problem is $\mathcal{P}$–SPACE complete.

(lxvii) **F** The halting problem is decidable.