EECS 376 Midterm Exam, Winter 2023

Standard Languages

You may rely on the following definitions without repeating them.

- $L_{\text{ACC}} = \{(\langle M \rangle, x) : M$ is a Turing machine that accepts $x\}$
- $L_{\text{HALT}} = \{(\langle M \rangle, x) : M$ is a Turing machine that halts on $x\}$
- ~~$L_{\varepsilon\text{-HALT}} = \{\langle M \rangle : M$ is a Turing machine that halts on $\varepsilon\}$~~
- ~~$L_{\emptyset} = \{\langle M \rangle : M$ is a Turing machine for which $L(M) = \emptyset\}$~~
- ~~$L_{\text{EQ}} = \{(\langle M_1 \rangle, \langle M_2 \rangle) : M_1, M_2$ are Turing machines for which $L(M_1) = L(M_2)\}$~~

Multiple Choice – 36 points

For each question in this section, **select exactly ONE answer by completely filling its circle** with a pencil or black ink. Each question in this section is worth 4 points.

1. Consider the following algorithm:

function $\textsc{Func}(A[1, \ldots, n])$
  if $n = 1$ **then**
    return $A[1]$
  $w \leftarrow \text{Func}(A[1, \ldots, \lceil 3n/8 \rceil])$
  $x \leftarrow \text{Func}(A[\lceil 3n/8 \rceil + 1, \ldots, \lceil 6n/8 \rceil])$
  $y \leftarrow \text{Func}(A[\lceil 5n/8 \rceil + 1, \ldots, n])$
  $z \leftarrow \text{Helper}(A[1, \ldots, n])$
  return $\max(w, x, y, z)$

Suppose that Helper takes $O(n)$ time on an array of n elements.

Choose the **tightest correct asymptotic bound** on the runtime of $\text{Func}(A[1, \ldots, n])$.

- ◯ $O(n)$
- ◯ $O(n^2)$
- ◯ $O(n^{\log_{8/3} 3})$
- ◯ $O(n \log n)$

2. On input n, algorithm M's worst-case runtime is $O(n^2)$, whereas algorithm N's is $\Theta(n^2 \log n)$.

   Choose the one statement that is **necessarily true**.

   ◯ On any input n, M's runtime is less than N's.

   ◯ For any input n larger than some fixed threshold, M's runtime is less than 1% of N's.

   ◯ For some (possibly small) input n, M's runtime is more than N's.

   ◯ For any input $n \geq 2^{10}$, M's runtime is at most 10% of N's.

3. Consider the following algorithm:

```
function PrintExamQuestion(n)
    while n > 0 do
        Print "Why is a raven like a writing desk?"
        n ← ⌊3n/4⌋
    Print "One is nevar backwards and one is for words"
```

   Choose the **tightest correct asymptotic bound** on the algorithm's runtime. (Assume that arithmetic operations take constant time.)

   ◯ $O(n^{4/3})$

   ◯ $O(n)$

   ◯ $O(\log n)$

   ◯ $O(1)$

4. Choose the correct option: (All / some / none) of the following statements are **true**:

   - Karatsuba's algorithm for multiplying n-bit integers is asymptotically faster than $\Theta(n^2)$-time naïve multiplication because it makes three recursive calls on integers of about $n/2$ bits, and its combine step takes $O(n)$ time.
   - The MergeSort algorithm for sorting an n-element array is asymptotically faster than $\Theta(n^2)$-time naïve sorting because it makes two recursive calls on arrays of about $n/2$ elements, and its combine step takes $O(n)$ time.
   - The divide-and-conquer algorithm for finding a closest pair of points in two dimensions, if modified to compute <u>all</u> pairwise distances between points in the "δ-strip," is not asymptotically faster than the $\Theta(n^2)$-time brute-force algorithm.

   ◯ All

   ◯ Some (but not all)

   ◯ None

5. True or False: any DFA, given any input string (made up of characters from the DFA's alphabet), either accepts or rejects.

   ◯ True

   ◯ False

   ◯ Determining the answer is undecidable

6. Choose the option that **makes the following statement true**: For (all / some / no) decidable languages L_1, L_2, L_3, the language

$$L = \{x \in \Sigma^* : x \text{ is in } \textbf{exactly 2 of } L_1, L_2, L_3\}$$

is decidable.

   - ◯ All
   - ◯ Some (but not all)
   - ◯ No
   - ◯ Determining the answer is undecidable

7. Choose the **only false statement** from the following.

   - ◯ There exists a language L that is decidable by a program written in Python, but is not decidable by any Turing machine.
   - ◯ Any language that is decidable by a DFA is also decidable by some Turing Machine.
   - ◯ There exists a C++ program that recognizes $\emptyset$ (the empty set).
   - ◯ If L_1 is undecidable and $L_1 \leq_T L_2$, then L_2 is undecidable.

8. Choose the **only decidable language** from the following.

   - ◯ $L_A = \{(\langle M \rangle, x) : M$ is a TM that accepts x after running for more than 376 steps$\}$
   - ◯ $L_B = \{(\langle D \rangle, x) : D$ is a DFA that accepts $x\}$
   - ◯ $L_C = \{(\langle M_1 \rangle, \langle M_2 \rangle) : M_1, M_2$ are TMs and $L(M_1) \subseteq L(M_2)\}$
   - ◯ $L_D = \{\langle M \rangle : M$ is a TM that accepts some input$\}$

9. Choose the **only true statement** from the following.

   - ◯ If language L is undecidable and Turing machine D is a decider, then there exists some $x \in L$ that D rejects.
   - ◯ If language L is undecidable, then even though no Turing machine decides L, there does exist a Turing machine M that accepts every $x \in L$ and does not accept every $x \notin L$.
   - ◯ If language L is undecidable and Turing machine D does not reject any $x \in L$ and does not accept any $x \notin L$, then D must loop on some input.
   - ◯ If language L is undecidable and D is a Turing machine, then at least one of these must hold: (1) D rejects or loops on every $x \in L$; (2) D accepts or loops on every $x \notin L$.

UNIQNAME (print): ______________________________

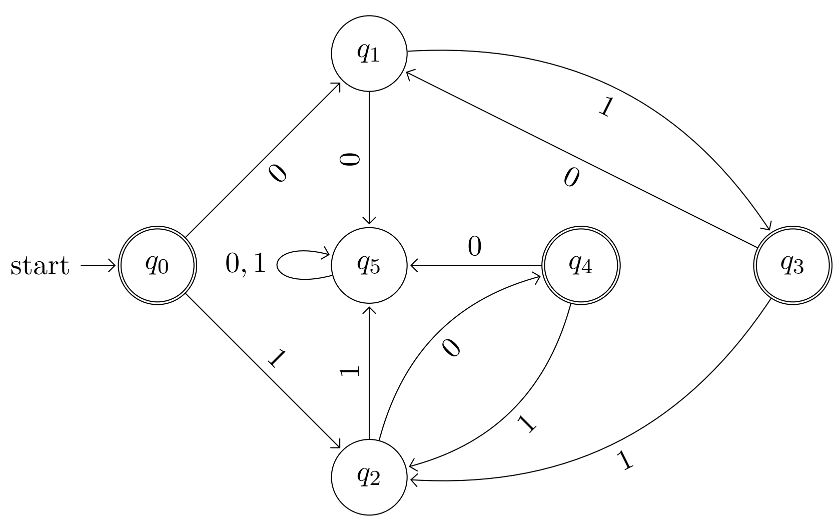
Shorter Answer – 24 points

Each of the three questions in this section is worth 8 points.

1. Give the **tightest correct asymptotic** (big-O) bound, as a function of n, on the **worst-case number of additions** done by the following algorithm, along with **a value of k that induces the worst case**. Also **state whether this is polynomial in the input size** or not. No explanation or proof is needed.

```
function FUNK(n, k)                    ▷ n is a positive integer, and k ∈ {1, 2, ..., n}
    x = 0
    for i = 1, 2, ..., k do
        for j = 1, 2, ..., n − k do
            x = x + 1
    return x
```

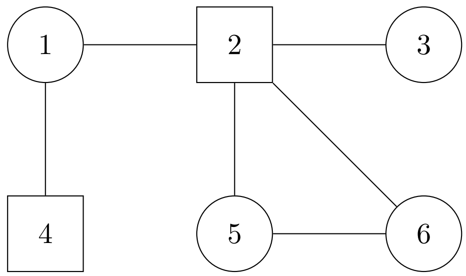
2. What language does the following DFA decide? Give your answer in "regex" form, or in precise English. No explanation or proof is needed.



3. A dominating set S in a graph G is a set of vertices for which every vertex of G either is in S, or is adjacent to some vertex in S.

   We are interested in a smallest dominating set of a given graph, i.e., one that has the fewest possible vertices. (There may be more than one smallest dominating set.)

   For example, the following graph has a smallest dominating set $S^* = \{2, 4\}$: every vertex other than 2 and 4 is adjacent to 2 or 4 (or both), and there is no dominating set consisting of a single vertex.



   Consider the following greedy algorithm for finding a dominating set in a graph.

```
function GreedyDS(G)
    S ← ∅
    while G has at least one vertex do
        Select any vertex v in G that has largest degree (i.e., the most neighbors)
        Add v to S
        Remove v and all its neighbors, including all incident edges, from G
    return S
```

   Give a small graph G on which the algorithm **might not return a** smallest **dominating set**. Specifically, **give a sequence of vertices that the algorithm might choose** to make up its final output set, and **give an optimal dominating set of G that is smaller** than this output set.

   You may use the box on the next page to continue your answer. If you do, clearly write that your "answer continues to the next page."

UNIQNAME (print): ______________________________

Proofs and Longer Answers – 40 points

Each of the two questions in this section is worth 20 points.

1. Let $A[1, \ldots, n]$ be an array of $n \geq 1$ positive real numbers. A decreasing subsequence of A is a sequence of array elements $A[i_1] > A[i_2] > \cdots > A[i_m]$, where $i_1 < i_2 < \cdots < i_m$ are some (not necessarily contiguous) array indices.

   We are interested in the maximum sum obtainable by decreasing subsequences of A, i.e.,

   $$\max\{A[i_1] + \cdots + A[i_m] : A[i_1] > A[i_2] > \cdots > A[i_m] \text{ is a decreasing subsequence of } A\}.$$

   For example, for $A = [4, 2, 2, 3, 5, 1]$, both $S_1 = [4, 3, 1]$ and $S_2 = [5, 1]$ are decreasing subsequences. The sum over S_1 is $8 = 4 + 3 + 1$, whereas the sum over S_2 is $6 = 5 + 1$. It can be verified that S_1 has the maximum sum overall, i.e., no decreasing subsequence of A has a sum greater than 8. (Note that the subsequence $[4, 2, 2, 1]$ has a sum of 9, but it is not decreasing, because $2 \not> 2$.)

   (a) Let $H(i)$ denote the maximum sum obtainable by a decreasing subsequence of A that ends at index i, i.e.,

   $$H(i) = \max\{A[i_1] + \cdots + A[i_m] : A[i_1] > \cdots > A[i_m] \text{ is a decreasing subsequence of } A \text{ with } i_m = i\}.$$

   Give a correct recurrence relation for $H(i)$, including base case(s), that is suitable for an efficient dynamic-programming algorithm. Briefly justify your answer.

   Hint: an optimal decreasing subsequence ending at index i is either a single element, or has an immediate predecessor (a second-to-last element). What are the possible indices for this predecessor? What is true about the part of the subsequence that ends at this predecessor?

(b) Using your answer to part (a), **describe a dynamic-programming algorithm** that outputs the maximum sum over all decreasing subsequences of an input array A. Also **give the tightest correct asymptotic (big-O) running time** for your algorithm, as a function of n. Assume that addition, comparisons, and similar basic operations take constant time.

2. ~~Define the language~~

$$L = \{(\langle M \rangle, x) : M \text{ is a TM and } x \text{ is a } \underline{\text{shortest}} \text{ string that } M \text{ accepts}\}.$$

~~Prove that L is undecidable by **showing one of** $L_{\text{ACC}} \leq_T L$ or $L_{\text{HALT}} \leq_T L$.~~