

Practice Final Exam VII

We strongly recommend that you work through this exam under realistic conditions rather than just flipping through the problems and seeing what they look like. Setting aside three hours in a quiet space with your notes and making a good honest effort to solve all the problems is one of the single best things you can do to prepare for this exam. It will give you practice working under time pressure and give you an honest sense of where you stand and what you need to get some more practice with.

This practice final exam is the final exam from Winter 2018. The sorts of questions here are representative of what you might expect to get on the upcoming final exam, though the point balance and distribution of problems might be a bit different.

The exam policies are the same for the midterms – closed-book, closed-computer, limited note (one double-sided sheet of 8.5" × 11" paper decorated however you'd like).

You have three hours to complete this exam. There are 78 total points.

Question	Points
(1) Graphs, Logic, and Sets	/ 10
(2) Equivalence Relation, Functions, and Sets	/ 19
(3) Strict Orders and Induction	/ 18
(4) Regular and Context-Free Languages	/ 17
(5) R and RE Languages	/ 10
(6) P and NP Languages	/ 3
	/ 78

Problem One: Graphs, Logic, and Sets**(10 Points)**

(We recommend spending about 20 minutes on this problem.)

This question explores a class of graphs called the *hypercube graphs*, which have a ton of applications throughout CS theory. If you study high-performance computing, parallel programming, binary encoding schemes, or combinatorics, you'll likely see them make an appearance.

We'll begin this exploration by asking you to translate a statement into first-order logic.

i. **(4 Points)** Given the predicates

$x \in y$, which states that x is an element of y , and

$x \notin y$, which states that x is not an element of y ,

along with the constant symbols S and T , which represent some two sets, translate the statement $|S \Delta T| = 1$ into first-order logic.

Let's introduce some new notation. For any natural number k , let's have

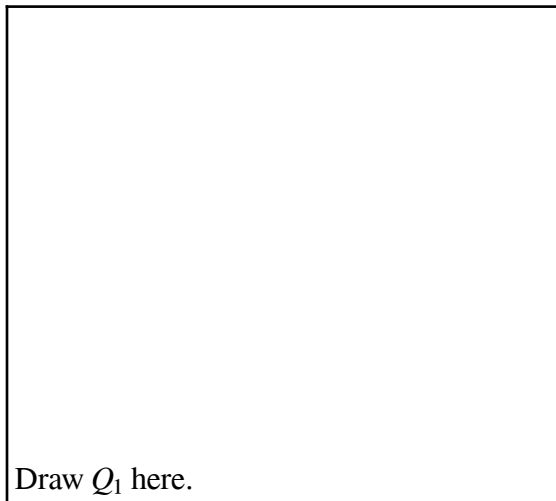
$$[[k]] = \{ n \in \mathbb{N} \mid n < k \}.$$

The *hypercube graph of order k* , denoted Q_k , is defined as follows:

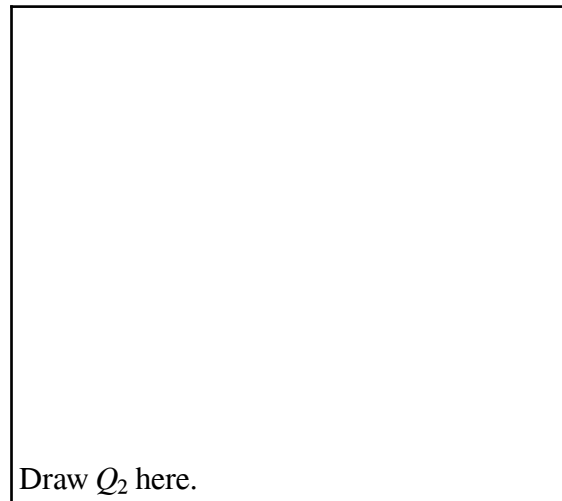
- The nodes of Q_k are the elements of $\mathcal{P}([k])$.
- There is an edge between a pair of nodes S and T if and only if $|S \Delta T| = 1$.

That definition might seem like a mouthful, but it makes a lot more sense once you have a visual for what's going on here.

- ii. **(3 Points)** In the space below, draw Q_1 , Q_2 , and Q_3 . (You should have eight nodes in Q_3 .) Please label each node with the set it corresponds to.



Draw Q_1 here.



Draw Q_2 here.



Draw Q_3 here.

“Hey,” you might be saying, “didn’t we do this on Problem Set Five?” Yep! We incorporated this exam question into that problem set. Now – can you still solve this problem without checking the solutions set?

As a refresher from the previous page, if k is a natural number, we'll define

$$[[k]] = \{ n \in \mathbb{N} \mid n < k \}.$$

The *hypercube graph of order k* , denoted Q_k , is defined as follows:

- The nodes of Q_k are the elements of $\wp([[k]])$.
- There is an edge between a pair of nodes S and T if and only if $|S \Delta T| = 1$.

It turns out that for any natural number $k \geq 1$ that Q_k is 2-colorable.

- iii. **(3 Points)** Let $k \in \mathbb{N}$ be an arbitrary natural number where $k \geq 1$. In a 2-coloring of Q_k , each node will be assigned one of two different colors. Answer the questions given in the space below to tell us which nodes will be assigned which colors.

Which nodes will be given the first color?

Which nodes will be given the second color?

What two colors would you use? (Any two colors work. This is just for fun. 😊)

Problem Two: Equivalence Relations, Functions, and Sets (19 Points)

(We recommend spending about 50 minutes on this problem.)

Equivalence relations are a workhorse in discrete mathematics and can be used to rigorously pin down all sorts of structures. This problem explores an important operation on equivalence relations and its properties.

Let's begin with a refresher on a definition. If R is an equivalence relation over a set A and $x \in A$, then the *equivalence class of x* with respect to R , denoted $[x]_R$, is the set

$$[x]_R = \{ y \in A \mid xRy \}.$$

Intuitively, $[x]_R$ consists of all the elements of A that x is related to by R .

Now, a new definition. If R is an equivalence relation over a set A , then the set A / R is the set of all the equivalence classes of the elements of A . Formally speaking, we say that

$$A / R = \{ [x]_R \mid x \in A \}.$$

This set is sometimes called the *quotient set* of R .

- i. (3 Points) Consider the following binary relation E over the set \mathbb{N}^2 :

$$(m_1, n_1) E (m_2, n_2) \quad \text{if} \quad m_1 + m_2 \text{ is even and } n_1 + n_2 \text{ is even.}$$

(Remember that this “if” means “is defined as” and is not an implication.)

What is \mathbb{N}^2 / E ? Briefly justify your answer, but no formal proof is required.

Here's a terminology refresher from the previous page:

$$[x]_R = \{ y \in A \mid xRy \} \qquad A / R = \{ [x]_R \mid x \in A \}.$$

Although equivalence relations come in all sorts of shapes and flavors, there is a single equivalence relation that's, in some sense, the "most fundamental" equivalence relation: the equality relation. In the remainder of this problem, you'll show that every equivalence relation's behavior can be thought of as the behavior of the equals relation over some well-chosen collection of sets.

Let R be an equivalence relation over a set A . We can define a function $f: A \rightarrow A / R$ as follows:

$$f(x) = [x]_R.$$

That is, f maps each element of A to its equivalence class.

- ii. **(3 Points)** Below are a series of statements about the behavior of this function f . For each statement, decide whether that statement is always true regardless of what R is, always false regardless of what R is, or whether it depends on the choice of R . (Remember that R is assumed to be an equivalence relation.) No justification is necessary. There is no penalty for an incorrect guess.

The function f is injective.

- True, regardless of what R is. False, regardless of what R is. It depends on R .
-

The function f is surjective.

- True, regardless of what R is. False, regardless of what R is. It depends on R .
-

The function f is bijective.

- True, regardless of what R is. False, regardless of what R is. It depends on R .

As a refresher from the previous page, we've let R be an *equivalence relation* over some set A . We've used the following terminology:

$$[x]_R = \{ y \in A \mid xRy \} \qquad A / R = \{ [x]_R \mid x \in A \}.$$

We've also let $f : A \rightarrow A / R$ be a function defined as $f(x) = [x]_R$.

This function has a wonderful property:

For any $a \in A$ and any $b \in A$, we have aRb if and only if $f(a) = f(b)$.

In the next two parts of this problem, we'd like you to prove this.

- iii. **(5 Points)** Using a proof by contrapositive, prove that for any $a, b \in A$ that if $f(a) = f(b)$, then aRb .

Feel free to use this space for scratch work. There's room to write your answer on the next page of this exam.

(Extra space for your answer to Problem Two, Part (iii), if you need it.)

As a refresher from the previous page, we've let R be an *equivalence relation* over some set A . We've used the following terminology:

$$[x]_R = \{ y \in A \mid xRy \} \qquad A / R = \{ [x]_R \mid x \in A \}.$$

We've also let $f : A \rightarrow A / R$ be a function defined as $f(x) = [x]_R$.

- iv. **(8 Points)** Prove that if $a, b \in A$ and aRb , then $f(a) = f(b)$. As a hint, how do you show that two sets are equal to one another?

Feel free to use this space for scratch work. There's room to write your answer on the next page of this exam.

(Extra space for your answer to Problem Two, Part (iv), if you need it.)

Problem Three: Strict Orders and Induction**(18 Points)***(We recommend spending about 35 minutes on this problem.)*

Let's begin with a new definition. If R and S are binary relations over the same set A , then the **composition of R and S** , denoted $R \circ S$, is a binary relation over A defined as follows:

$$x(R \circ S)y \quad \text{if} \quad \exists z \in A. (xRz \wedge zSy).$$

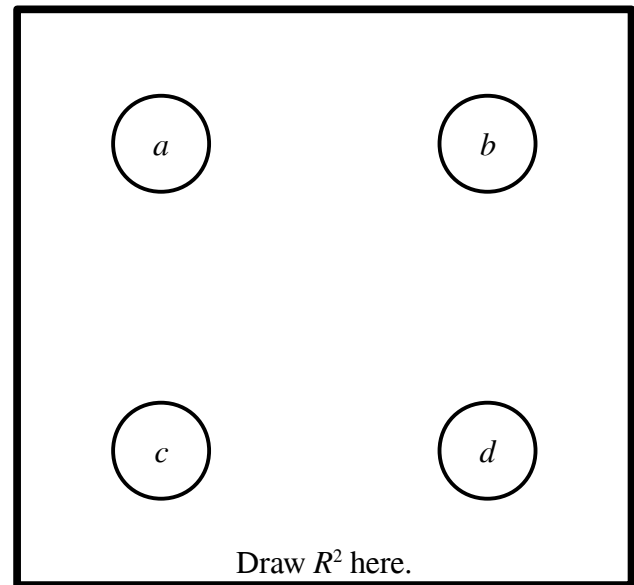
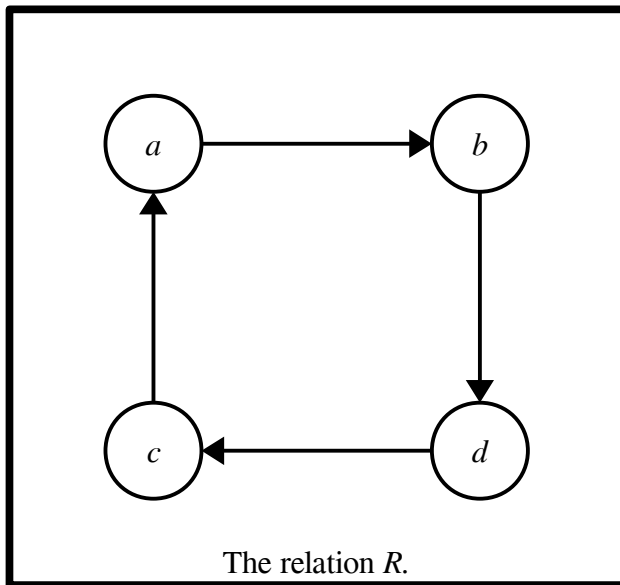
Having defined the composition of two relations, we can inductively define the **n th power** of a binary relation R over a set A as follows:

$$xR^1y \quad \text{if} \quad xRy$$

$$xR^{n+1}y \quad \text{if} \quad x(R \circ R^n)y$$

Remember that the word “if” in the above contexts means “is defined as” and is not an implication, and note that R^n is only defined when $n \geq 1$.

- i. **(3 Points)** Below is a drawing of a binary relation R over the set $\{a, b, c, d\}$. In the indicated space, draw a picture of R^2 . No justification is necessary.



As refreshers from the previous page, if R and S are binary relations over the same set A , then the relation $R \circ S$ is a binary relation over A defined as follows:

$$x(R \circ S)y \quad \text{if} \quad \exists z \in A. (xRz \wedge zSy).$$

The *n th power* of a binary relation R over a set A is defined as follows:

$$\begin{aligned} xR^1y & \quad \text{if} \quad xRy \\ xR^{n+1}y & \quad \text{if} \quad x(R \circ R^n)y \end{aligned}$$

Remember that the word “if” in the above contexts means “is defined as” and is not an implication, and note that R^n is only defined when $n \geq 1$.

- ii. **(15 Points)** Consider the $<$ relation over the set \mathbb{R} . Prove that the following is true for all nonzero natural numbers n :

$$\forall x \in \mathbb{R}. \forall y \in \mathbb{R}. (x < y \leftrightarrow x <^n y).$$

Feel free to use the inequality $a < \frac{a+b}{2} < b$, which is true for any real numbers a and b where $a < b$. You can assume that the $<$ relation over \mathbb{R} is a strict order.

(Extra space for your answer to Problem Three, Part (ii), if you need it.)

Problem Four: Regular and Context-Free Languages**(17 Points)***(We recommend spending about 50 minutes on this problem.)*

If you're living in a world where the only legal arithmetical operation is addition, you can write out a bunch of different expressions that evaluate to odd numbers:

- 3
- 6 + 3
- 3 + 3 + 3
- 6 + 3 + 6 + 3 + 6 + 3 + 6
- 6 + 6 + 6 + 3 + 6 + 6
- 3 + 3 + 3 + 3 + 3

Let $\Sigma = \{ 3, 6, + \}$ and consider the following language L_1 over Σ :

$L_1 = \{ w \in \Sigma^* \mid w \text{ doesn't use any numbers besides 3 and 6 and evaluates to an odd number. } \}$

All of the strings shown above are in L_1 . Here's a sampler of strings that *aren't* in L_1 :

- 6 *(this is an even number)*
- 6 + 6 *(this is an even number)*
- 63 *(no multidigit numbers)*
- + 6 *(syntactically invalid)*
- 6 ++ 6 *(syntactically invalid)*
- ϵ *(not a valid expression)*

This turns out to be a regular language.

- i. **(3 Points)** Design an NFA for L_1 . In the space at the bottom of the page, write a brief explanation (at most two sentences) for how your NFA works.

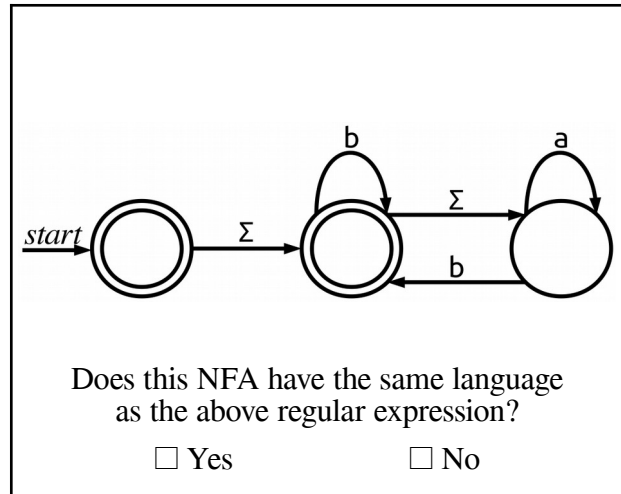
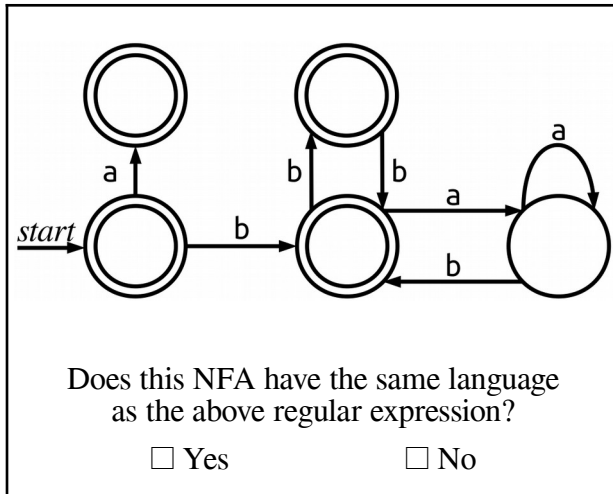
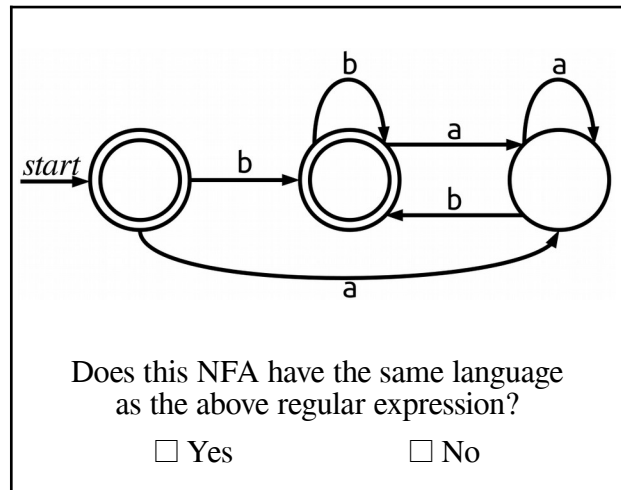
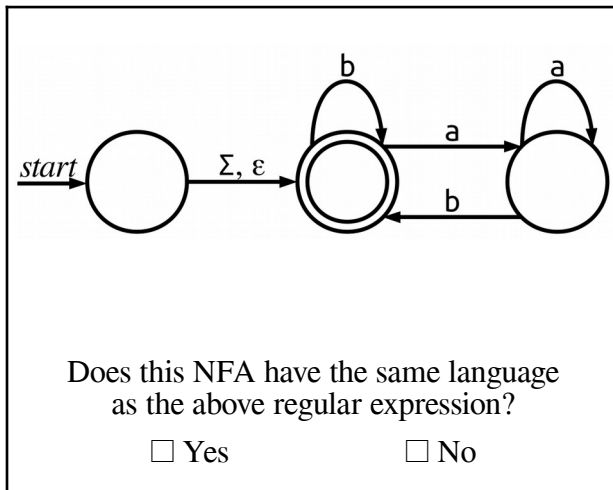
Explanation for this NFA (at most two sentences):

Consider the following regular expression over the alphabet $\{a, b\}$:

$$\Sigma?(a^+b \cup b^+)^*$$

This question explores some properties of this regular expression.

- ii. **(4 Points)** Below are four NFAs over the alphabet $\{a, b\}$. For each NFA, decide whether the language of that NFA is the same as the language described by the above regular expression. No justification is necessary. There is no penalty for an incorrect guess.



Let $\Sigma = \{a, b\}$ and consider the finite language $L_2 = \{\epsilon, a\}$.

- iii. (7 Points) Design a DFA for L_2 that is as small as possible. Then, prove that your DFA is as small as possible by using the following theorem that you proved on Problem Set Seven:

Theorem: Let L be a language over Σ . Suppose there's a *finite* set S such that any two distinct strings $x, y \in S$ are distinguishable relative to L (that is, $x \not\equiv_L y$ for any two strings $x, y \in S$ where $x \neq y$.) Then any DFA for L must have at least $|S|$ states.

Feel free to use the space below for scratch work. There's room to write your answer on the next page of this exam.

(Extra space for your answer to Problem Four, Part (iii), if you need it.)

On Problem Sets Six, Seven, and Eight, you explored languages involving taking a walk with your dog. The next part of this problem concerns more of the challenges of pet ownership.

Imagine that you and your dog are taking a walk without a leash. Let $\Sigma = \{y, d\}$, where y represents you taking a step and d represents your dog taking a step. Then, consider this language L_3 :

$$L_3 = \{ w \in \Sigma^* \mid w \text{ represents a walk where your dog ends up one step ahead of you} \}.$$

For example, the following strings are in L_3 :

- d
- dyd
- $yyyydddd$
- $dydydydyd$
- $ddydydd$
- $yyyyddddyydddd$

Notice that there are no restrictions on how far apart you and your dog can get. The only restriction is that your dog ends up a single step ahead of you.

The following strings are not in L_3 :

- yd
- y
- $yyyddd$
- $dddyyy$
- $dddyyyyy$
- ϵ

This language happens to be context-free.

- iv. **(3 Points)** Write a CFG for L_3 . In the space at the bottom of the page, write a brief explanation (at most two sentences) for how your CFG works.

As a hint, split the walk into three pieces – the part before the extra step your dog takes, the extra step your dog takes, and the part after that extra step.

Explanation for this CFG (at most two sentences):

Problem Five: R and RE Languages**(10 Points)**

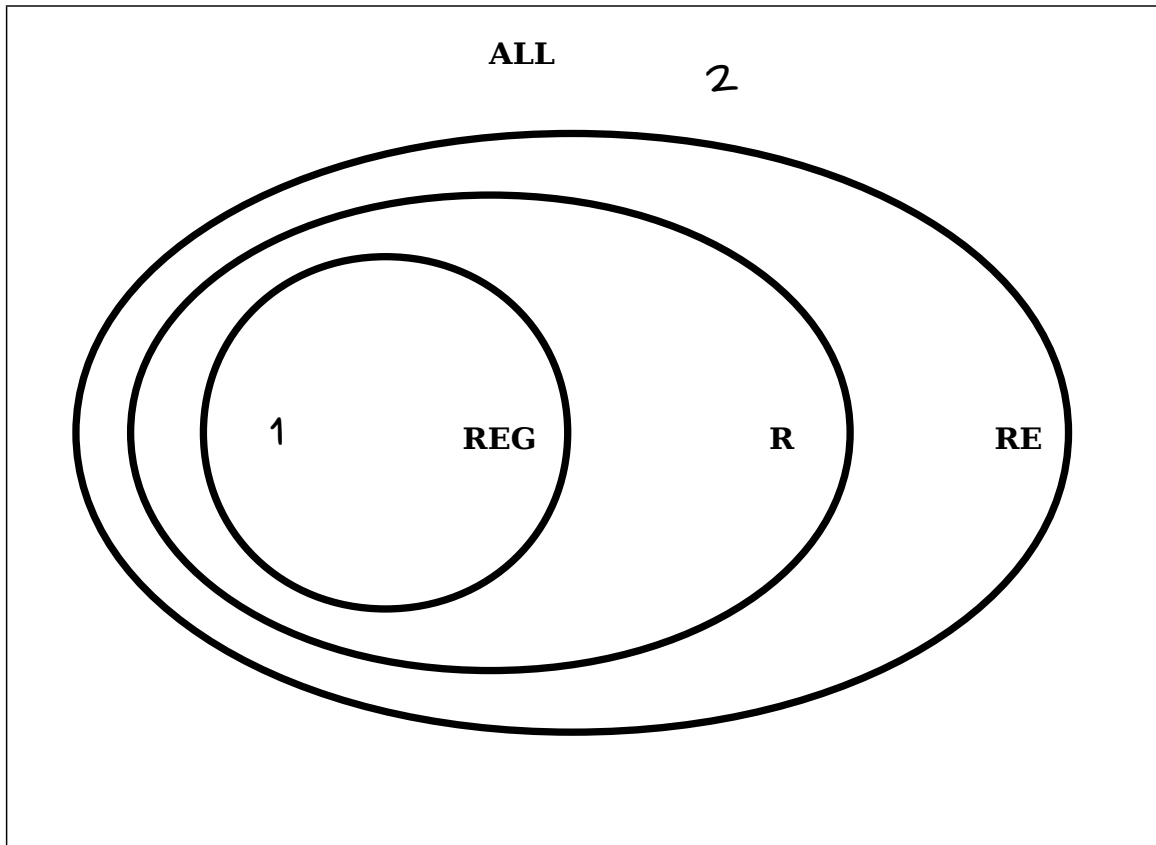
(We recommend spending about 20 minutes on this problem.)

Over the past two weeks you've seen your fair share of undecidable problems. If a problem is undecidable, then there's no way to build a TM for that problem that always halts and gives the right answer. However, it's still possible to write a program for an undecidable problem that can get the right answer on many possible inputs – just not *all* of them.

Here are two new definitions. If L is a language over Σ , then a **sound approximation** to L is a language S over Σ such that $S \subseteq L$ and a **complete approximation** of L is a language C over Σ such that $L \subseteq C$.

- i. **(3 Points)** Briefly explain why *every* language has a decidable sound approximation and a decidable complete approximation. This shows that even if a language is undecidable, it's still possible to build a TM that can provide the right answers on some number of strings.

- ii. **(7 Points)** Below is a Venn diagram showing the overlap of different classes of languages we've studied so far. We have also provided you a list of numbered languages. For each of those languages, draw where in the Venn diagram that language belongs. As an example, we've indicated where Language 1 and Language 2 should go. No proofs or justifications are necessary, and there is no penalty for an incorrect guess.



1. Σ^*
2. L_D
3. $\{ a^n b^m \mid n \in \mathbb{N} \text{ and } m \in \mathbb{N} \}$
4. $\{ \langle T \rangle \mid T \text{ is a tournament and the players in } T \text{ are the CS103 staff members} \}$
5. $\{ \langle P \rangle \mid P \text{ is a syntactically correct Java program whose source code contains the string quokka} \}$
6. $\{ \langle P \rangle \mid P \text{ is a syntactically correct Java program that, when run, at some point prints the string quokka} \}$
7. $\{ \langle P \rangle \mid P \text{ is a syntactically correct Java program that, when run, never prints the string quokka} \}$
8. The intersection of languages (5) and (6).
9. $\{ w \mid w \in L_D \}$

Problem Six: P and NP Languages**(3 Points)**

(We recommend spending about 5 minutes on this problem.)

Here's a quick series of true/false questions about the **P** and **NP** languages. Each correct answer is worth one point, and there is no penalty for an incorrect guess. You do not need to justify your answers.

i. If $\text{SAT} \in \mathbf{P}$, then $\mathbf{P} = \mathbf{NP}$.

True

False

ii. Every verifiable language is in **NP**.

True

False

iii. There is a decidable **NP**-hard language.

True

False

We have one final question for you: do *you* think $\mathbf{P} = \mathbf{NP}$? Let us know in the space below. There are no right or wrong answers to this question – we're honestly curious to hear your opinion!

I think $\mathbf{P} = \mathbf{NP}$

I think $\mathbf{P} \neq \mathbf{NP}$