Guidelines:

Please work by yourself and do not discuss this problem set with anyone until after December 15.

Please submit your assignment to the TA by email or put it in his COS mailbox. Solutions should be typed, drawings can be done by hand.

You may consult the course notes and the text (Sipser).

The assignment questions may be similar to questions on problem sets from past offerings of this course or courses at other universities. Using any preexisting solutions from these or any other sources is strictly prohibited.

The total number of points for this problem set is a 100. Please attempt all problems. Please provide full proofs, unless stated otherwise.

Question 1 (20 points).

1. (5 points) Prove that \( \text{BPP} \) is closed under union and intersection. That is, show that if \( L_1, L_2 \in \text{BPP} \) then \( L_1 \cup L_2 \in \text{BPP} \) and \( L_1 \cap L_2 \in \text{BPP} \).

2. (15 points) Define the following complexity classes:

   **Definition (The class RP, Randomized Polynomial time):** The class of languages \( L \) for which there is a polynomial time Turing Machine \( M(x, r) \) and a polynomial \( p : \mathbb{N} \rightarrow \mathbb{N} \) such that:

   \[
   x \in L \implies \Pr_{r \in \{0,1\}^{p(|x|)}} [M(x, r) = 1] \geq 2/3
   \]

   \[
   x \notin L \implies \Pr_{r \in \{0,1\}^{p(|x|)}} [M(x, r) = 1] = 0
   \]

   **Definition (The class coRP):** The class of languages \( L \) such that \( \overline{L} \in \text{RP} \).

   **Definition (The class ZPP, Zero-error Probabilistic Polynomial time):** The class of languages \( L \) for which there is a probabilistic Turing Machine \( M \) running in
expected polynomial time and correctly decides \( L \). That is, there exist a Turing machine \( M(x, r) \) and a polynomial \( q : \mathbb{N} \to \mathbb{N} \) such that \( M(x, r) \) accepts if and only if \( x \in L \), and the expected running time of \( M(x, r) \) (over the selection of the random string \( r \)) is at most \( q(|x|) \).

Show that \( \text{ZPP} = \text{RP} \cap \text{coRP} \).

Hint: Show \( \text{ZPP} \subseteq \text{RP} \cap \text{coRP} \) and \( \text{RP} \cap \text{coRP} \subseteq \text{ZPP} \).

For the first, it is important to make sure that the machine terminates in worst-case polynomial time. That is, you want the machine to always halt after at most a polynomial number of steps. When should the machine halt? How can you ensure that the error is only one sided?

You may want to use Markov’s inequality: If \( X \) is a nonnegative random variable and \( a > 0 \), then

\[
\Pr[X \geq a] \leq \frac{E[X]}{a}.
\]

For the latter, execute the \( \text{RP} \) and the \( \text{coRP} \) machines in alternation. When are you certain about the answer? How many executions (in expectation) would it take to reach this ‘certainty’?

Remark: This is the analogue of the claim “\( \text{P} = \text{NP} \cap \text{coNP} \)” in the randomized complexity world. The claim for \( \text{P} \) is believed to be false. However, the randomized version does hold.

Question 2 (15 points).

Let \( L, M \in \Sigma^* \) be languages on alphabet \( \Sigma \). For strings \( x = x_1x_2\cdots x_n \) and \( y = y_1y_2\cdots y_m \), define \( x \bowtie y \) to be (assuming \( n < m \), the analogous definition for \( n \geq m \))

\[
x \bowtie y = x_1y_1x_2y_2\cdots x_ny_ny_{n+1}\cdots y_m.
\]

Extend this to define \( L \bowtie M = \{ x \bowtie y \mid x \in L, y \in M \} \). Prove that \( L \bowtie M \) is regular if \( L \) and \( M \) are.

Hint: First solve the case \( n = m \) by constructing an automaton whose states are pairs. How would you generalize to any \( n, m \) using the equivalence between DFAs and NFAs?

Question 3 (20 points).

Recall that a function \( f : \mathbb{N} \to \mathbb{N} \) is computable if there exists a Turing Machine that on input \( 1^n \) terminates with \( 1^{f(n)} \) on its tape.

Recall that a function \( f : \mathbb{N} \to \mathbb{N} \) is computable if there exists a Turing Machine that on input \( 1^n \) terminates with \( 1^{f(n)} \) on its tape.
**Definition (Busy beavers):** Let $n \in \mathbb{N}$. Denote by $\mathcal{M}(n)$ the set of Turing Machines with $n$ states and tape alphabet $\Gamma = \{0, 1, \text{blank}\}$. When the machines in $\mathcal{M}(n)$ get no input (at the beginning of the execution the tape contains only blanks), they halt with the accepting state.  

The *busy beaver function* $BB: \mathbb{N} \to \mathbb{N}$ defines $BB(n)$ to be the maximum number of ones that a machine in $\mathcal{M}(n)$ that is run with no input has on its tape when halting.

1. (5 points) Prove that $BB(m) > BB(n)$ if $m > n$.  
2. (5 points) Argue that for every computable function $f$, there exists a constant $c$ such that for all $n$, we have $BB(c + n) \geq f(BB(n))$.  
3. (5 points) Show that $BB(n) \geq 2n - d$ for some constant $d$.  
4. (5 points) Conclude that $BB(n)$ is not a computable function.

**Question 4 (15 points).**  
Consider the following solitaire game. You are given a $k \times \ell$ board, for some $k, \ell \in \mathbb{N}$. Each one of the $k\ell$ positions may be empty or occupied by either a red stone or a blue stone. Initially, some configuration of stones is placed on the board. Then, for each column you must remove either all of the red stones in that column or all of the blue stones in that column. (If a column already has only red stones or only blue stones in it then you do not have to remove any further stones from that column.) The objective is to leave at least one stone in each row. Finding a solution that achieves this objective may or may not be possible depending upon the initial configuration. Let  

$$\text{solitaire} = \{G : G \text{ is a game configuration with a solution}\}.$$  

Prove that solitaire is NP-complete.  

*Hint:* Red and blue correspond to $x_i$ and $\bar{x}_i$. 

**Question 5 (15 points).**  
1. (10 points) Show that $\text{SPACE}(n)$ is not closed under polynomial-time reductions. That is, when $A \preceq_P B$ for languages $B \in \text{SPACE}(n)$ and some other language $A$, it may not be the case that $A \in \text{SPACE}(n)$ as well.  

*Hint:* Recall the Space Hierarchy Theorem: $\text{SPACE}(f(n)) \subsetneq \text{SPACE}(f^2(n))$. 

Given a language $A$, consider the padded version of $A$, where an input in $A$ is concatenated with $|x|^2 - |x|$ many # symbols,  

$$PAD_A = \{x \circ \#^{|x|^2 - |x|} : x \in A\}.$$
2. (5 points) Show that $\text{NP} \neq \text{SPACE}(n^k)$ for any $k \in \mathbb{N}$.

**Question 6 (15 points).**

For every integer $n$, consider the streaming complexity of the problem of deciding whether a graph on $n$ vertices, given by a stream of edges, is connected. That is, for a set of $n$ vertices $V$, our alphabet $E = \{(v, u) : v \neq u \in V\}$ is all possible (undirected) edges between these vertices and our stream is a sequence of these edges. If we call the set of all edges in this stream $E$, then $G = (V, E)$ is the undirected graph defined by it.

We want a streaming algorithm that takes the stream and computes whether or not the graph $G$ defined by it is connected. Show that every streaming algorithm for this problem requires $\Omega(n)$ bits of memory.