Problem Set 1

- Due date: **23 Feb, 2025** (released on 08 Feb, 2025).
- The points for each problem is indicated on the side. The total for this set is **70** points
- The problem set has a fair number of questions so please do not wait until close to the deadline to start on them. Try and do one question every couple of days.
- Turn in your problem sets electronically (PDF; either LaTeXed or scanned etc.) via email. If you submit a LaTeXdocument, you will get an additional **10 points**.
- Collaboration with other students taking this course is encouraged, but collaboration with others is not allowed. Irrespective of this, all writeups must be done individually and must include names of all collaborators (if any).
- Referring to sources other than the text book and class notes is STRONGLY DISCOURAGED. But if you do use an external source (eg.,other text books, lecture notes, or any material available online), ACKNOWLEDGE all your sources (including collaborators) in your writeup. This will not affect your grades. However, not acknowledging will be treated as a serious case of academic dishonesty.

---

0. **[Create homepage]** (7)

   Create a personal homepage for yourself and give the URL. (It doesn't have to be hosted on the STCS / TIFR servers. You are welcome to use Google Sites, or Github Pages, or Bitbucket Pages.)

1. **[Classification of problems]** (5)

   For each of these problems, mention (with justification) if they are in P, or not (known to be) in P, or in NP, or in coNP, or is NP-hard, or is coNP-hard etc.

   (a) Factoring $= \left\{(1^n, 1^k) \ : \ n \text{ has a prime factor less than } k\right\}$.

   (b) Contradiction $= \{\langle\varphi\rangle \ : \ \langle\varphi\rangle$ encodes a formula that is false for every assignment.$\}$ (For example, $\langle x_1 \wedge \neg x_1\rangle$ is in the language.)

   (c)

   $$\text{EquivalentFormulas} = \left\{ (\langle\varphi_1\rangle, \langle\varphi_2\rangle) \ : \ \begin{array}{c} \langle\varphi_1\rangle, \langle\varphi_2\rangle \text{ encode two formulas such that} \\ \text{for all } x \text{ we have } \varphi_1(x) = \varphi_2(x). \end{array} \right\}$$

2. **[Properties of reductions]** (5)

   We'll use $L_1 \leq_m^{\text{poly}} L_2$ to denote that there is a polynomial-time many-one reduction from L_1 to L_2.

   Answer each of the questions with True/False with brief justifications.

   (a) If $L_1 \leq_m^{\text{poly}} L_2$, then $\overline{L_2} \leq_m^{\text{poly}} \overline{L_1}$.

   (b) If $L_1 \leq_m^{\text{poly}} L_2$, then $\overline{L_1} \leq_m^{\text{poly}} \overline{L_2}$.

   (c) If L is NP-hard, then $\overline{L}$ is coNP-hard.

   (d) If $L_1 \leq_m^{\text{lin}} L_2$ and $L_2 \leq_m^{\text{lin}} L_3$, then $L_1 \leq_m^{\text{lin}} L_3$. (Here, $\leq_m^{\text{lin}}$ refers to linear-time many-one reductions).

(e) If $L_1 \leq_m^{\text{quad}} L_2$ and $L_2 \leq_m^{\text{quad}} L_3$, then $L_1 \leq_m^{\text{quad}} L_3$. (Here, $\leq_m^{\text{quad}}$ refers to quadratic-time many-one reductions).

3. **[Verifier perspective of NP]** (5)

In class, we gave a *machine-based* definition of the class NP. Often times, the following alternate definition is used.

> A language $L \subseteq \Sigma^*$ is said to be in NP if and only if there is a language $V_L \in \mathsf{P}$, and constants c, n_0 such that for any $x \in \Sigma^*$ with $|x| > n_0$, we have $x \in \mathsf{L}$ if and only if there is a string $w \in \Sigma^*$ with $|w| \leq |x|^c$ such that $(x, w) \in V_L$.

Formally prove that the above definition is equivalent to the definition we saw in class. (That is, any language that satisfies the above property is computable by a non-deterministic TM running in poly(n) time, and vice-versa.)

4. **[Operations on languages]** (2 + 2 + 2)

(a) If L_1, L_2 are two languages in NP, show that the languages $L_1 \cap L_2$ and $L_1 \cup L_2$ are in NP as well.

(b) For any three languages L_1, L_2, L_3,

$$\text{Maj}(L_1, L_2, L_3) = \{x \ : \ x \text{ is in at least two of the } L_i\text{'s}\}.$$

Show that, if $L_1, L_2, L_3 \in \mathsf{NP}$, then the language $\text{Maj}(L_1, L_2, L_3)$ is also in NP.

(c) For two languages L_1, L_2, let $L_1 \oplus L_2 = \{x \in \Sigma^* \ : \ x \text{ is in exactly one of } L_1, L_2\}$. If $L_1, L_2 \in \mathsf{NP} \cap \mathsf{coNP}$, show that $L_1 \oplus L_2 \in \mathsf{NP} \cap \mathsf{coNP}$ as well.

5. **[Not-all-equal-SAT]** (5 + 3 + 2)

The 'not-all-equal' function $\text{NAE}(x_1, \ldots, x_k)$ is the Boolean function that is true when not all the values of $x_1, \ldots, x_k$ are equal (that is, it is false only on the inputs $000\cdots0$ and $111\cdots1$).

NAE-SAT is the constrait satisfaction problem where each constraint is the above NAE-function. An instance of NAE-SAT is satisfiable if there is an assignment to the variables that satisfy all the constraints. (Similar to how CNF-SAT had the contraints as the 'OR' function.)

(a) Consider the following purported reduction from CNF-SAT to NAE-SAT:

> Consider a new variable z. For each clause of the form, $(x_i \vee x_j \vee x_k)$ in the CNF, add the constraint $\text{NAE}(x_i, x_j, x_k, z)$ to the NAE instance.

Prove that this is a legitimate reduction from CNF-SAT to NAE-SAT.

(b) If we begin with an instance of 3 CNF-SAT, then we end up with an instance of 4 NAE-SAT in the above reduction. Give a polynomial-time many-one reduction from 4 NAE-SAT to 3 NAE-SAT.

(c) What is the 2 NAE-SAT problem? Do you know it by a different name? Is it in P?

6. **[Equations that have integer solutions]** (5 + 2)

(a) Consider the following language

$$L = \left\{ (a, b) \ : \ a, b \in \mathbb{Z}\,, \ \begin{array}{c}\text{there exists integers } x, y \\ \text{such that } x^2 + ay + b = 0\end{array} \right\}$$

Here, the inputs a, b are provided in binary.

Prove that $L \in \mathsf{NP}$.

(b) Consider the following language

$$L' = \left\{ (a,b,c) \ : \ a,b,c \in \mathbb{Z} \ , \quad \begin{array}{c} \text{there exists integers } x,y,z \\ \text{such that } x^3 + ay^2 + bz + c = 0 \end{array} \right\}.$$

I don't know if this problem is in NP. Why do you think that is so?

7. **[Block-respecting TMs]** (10)

Hat-tip: This problem is from Ryan O'Donnell's complexity course

Let $B : \mathbb{N} \to \mathbb{N}$ be a "reasonable" function (increasing, time-constructible, yada yada). A deterministic Turing machine M is said to be *B-block-respecting* if it has the following property:

> Consider a length n input x. The tape(s) of the Turing machine are split into contiguous blocks of length $B(n)$ each. Each head of the Turing Machine crosses a block-boundary only at time-steps that are integer multiples of $B(n)$.
> (That is, within each block of $B(n)$ time steps, each head operates only within a particular block.)

Given a deterministic Turing machine M that runs in time $T(n)$, and a "reasonable" function $B : \mathbb{N} \to \mathbb{N}$, construct a deterministic B-block-respecting Turing machine M' for the same language while ensuring that the running time of M' is $O(T(n))$.

	siht	hasi	ftni	htro	orpe
this	isah	intf	orth	epro	blem
hasi	ftni	htro	orpe	melb	

[Hint: Suppose the machine M uses k tapes, you could perhaps make use of the above picture and use $3k$ tapes.]

8. **[Improving the time-hierarchy theorem]** (15)

For this problem, you may assume that any 'reasonable-looking' function is time-constructible. And whenever you see functions like $n^2 \log^{3/4}(n)$, assume that there is an implicit ceiling to make sure this is an integer etc. (Basically don't worry about technicalities!)

In class we proved that the deterministic time hierarchy theorem that stated the following:

> Suppose $t_1, t_2 : \mathbb{N} \to \mathbb{N}$ are non-decreasing time-constructible functions with $t_1(n), t_2(n) \geq n$. If we have $t_1(n) \log t_1(n) = o(t_2(n))$, then we have $\mathsf{DTIME}(t_1) \subsetneq \mathsf{DTIME}(t_2)$.

(a) Let $t_1, t_2, f : \mathbb{N} \to \mathbb{N}$ be time-constructible non-decreasing functions that satisfy $t_1(n), t_2(n), f(n) \geq n$. Show that $\mathsf{DTIME}(t_1(n)) = \mathsf{DTIME}(t_2(n))$ implies

$$\mathsf{DTIME}(t_1(f(n))) = \mathsf{DTIME}(t_2(f(n))).$$

[Hint: Padding.]

(b) Show that $\mathsf{DTIME}(n^2) \subsetneq \mathsf{DTIME}(n^2 \log^{3/4}(n))$.

[Hint: You may have to use the above part multiple times.]

(c) Extend this to show that for any rational number a, ε satisfying $a > 1$ and $0 < \varepsilon < 1$, we have

$$\mathsf{DTIME}(n^a) \subsetneq \mathsf{DTIME}(n^a (\log n)^\varepsilon).$$