Domande Importanti:

1. La classe P è l'insieme di Linguaggi L per cui una MDT Deterministica decide se $x \in L$ in tempo polinomiale
   La classe NP è l'insieme di Linguaggi L per cui una MDT non deterministica decide se $x \in L$ in tempo polinomiale
   (Def. alternativa: La classe NP è l'insieme dei linguaggi L per cui se $x \in L$ allora $\exists\, c(x)$, detto certificato, il quale ha le seguenti caratteristiche
   - $|c(x)|$ è polinomiale rispetto $|x|$
   - Una MT det. con input $c(x)$ e x è in grado di decidere se $x \in L$ in un tempo polinomiale rispetto $|c(x)|$ e $|x|$)
2. $P \neq NP$: Per dimostrare che $P \neq NP$ bisogna trovare un problema che si trova strettamente in NP, per fare ciò serve trovare almeno un problema con Lower Bound più che polinomiale per una macchina deterministica, ma per ora non è stato trovato
3. $P = NP$: Per dimostrare che $P = NP$ bisogna dimostrare che un problema NP-completo ha upper Bound Polinomiale per una macchina deterministica e cioè trovare un algoritmo che risolve il problema in tempo polinomiale

Problemi: NP (Calcolabilità, Vertex-cover, SAT ...)

4. Una MT universale prende in input la descrizione di una MT e l'input della MT da simulare. Una MT universale, quindi, simula la MT descritta perciò termina la computazione se la macchina simulata termina, altrimenti non termina. Il risultato della computazione è il risultato della macchina simulata. La MT universale formalizza il concetto di interprete

5. Un problema si dice indecidibile se non esiste alcun algoritmo che lo risolve.

6. Uso del Pumping Lemma per dimostrare che certi linguaggi non sono regolari:
   1. Sia L il linguaggio di cui si voglia dimostrare la non regolarità;
   2. Si fissi $n \geq 0$;
   3. Si scelga una stringa z (in funzione di n) in L con $|z| \geq n$;
   4. Si consideri un modo qualunque di spezzare la stringa z in u, v, w (cioè $z = uvw$) con $|uv| \leq n$ e $|v| \geq 1$
   5. Si dimostri che $\exists\, i : uv^i w \notin L$, $i \geq 0$

Esempio: $\mathcal{L} = \{ a^n b^n , n > 0 \}$ $\quad K = 4$

$z = \underbrace{aa}_{u}\underbrace{b}_{v}\underbrace{b}_{w}$ $\quad aab^i b$ per $i = 0 \Rightarrow aab \notin L$ Linguaggio non regolare

7. Pumping Lemma per linguaggi context-free: Sia L un linguaggio context-free. È possibile determinare una costante k, dipendente da L, tale che qualunque stringa $z \in L$ con $|z| > k$ si può esprimere come $z = uvwxy$ in cui:

① $|vwx| \leq k$

② $|v| + |x| > 0$

③ $uv^i wx^i y \in L$ per $i \geq 0$

Esempio: $L = \{ a^n b^n c^n \text{ con } n \geq 0 \}$ $\quad k = 4$

$z = \underbrace{aa|bb|cc}_{vwx}$ $\quad aa^i bb c^i c$ per $i = 0 \Rightarrow abbc \notin L$

Linguaggio non context-free

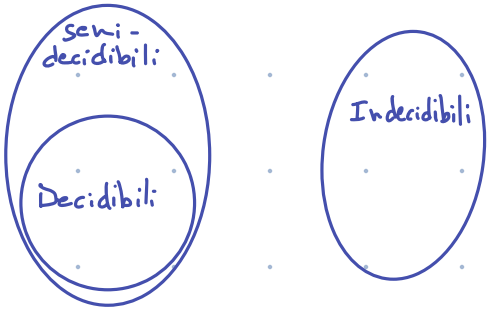
8. Diremo che un problema P ha una delimitazione superiore $O(f(n))$ se $\exists$ un algoritmo A_P che risolve P, con $C_{A_P}(n) \in O(f(n))$. In altre parole, un algoritmo che risolve un determinato problema gli "trasferisce" l'upper bound. Parlando dell'upper bound di un problema, si fa implicitamente riferimento al miglior algoritmo noto, quello con il costo asintotico minore

9. Sia dato un problema P per cui conosciamo diversi algoritmi di soluzione che appartengono tutti a $\Omega(f(n))$, cioè tutti gli algoritmi noti hanno un costo che cresce almeno come

$O(f(n))$. Possiamo affermare che $\Omega(f(n))$ rappresenta una delimitazione inferiore al costo di tutti gli algoritmi noti ma questo non implica che $\Omega(f(n))$ sia una delimitazione inferiore alla complessità del problema. Infatti le conoscenze che abbiamo non sono sufficienti ad escludere che non sia possibile progettare nuovi algoritmi che hanno un costo asintotico migliore.

10. Una macchina di Turing non deterministica è una macchina di Turing a cui non viene imposto il vincolo che, dato uno stato della macchina e un simbolo letto dalla testina, la transizione da eseguire sia univocamente determinata. Quindi, $f(q,s)$ non è una funzione ma per un dato q e s possiamo avere più valori. La macchina termina con successo se almeno una scelta porta a stato finale. La computazione di MdT può essere vista come un albero, detto albero delle computazioni.

11. Insieme linguaggi Decidibili, Indecidibili e semidecidibili:



12. Il linguaggio H, relativo al problema dell'arresto, può essere definito matematicamente come l'insieme delle coppie ordinate di programmi e input tali che il programma termini (si arresti) dato quell'input.
Formalmente, possiamo definire il linguaggio H come segue:
$H = \{(P, x) \mid$ il programma P termina dato l'input $x\}$
Quindi, se una coppia (P, x) appartiene al linguaggio H, significa che il programma P termina correttamente dato l'input x.

13. Riduzione Polinomiale: Un linguaggio L_1 è riducibile a un linguaggio L_2 se $\exists$ una funzione totale calcolabile in tempo polinomiale $f: \{0,1\}^* \to f: \{0,1\}^*$, detta riduzione polinomiale, tale che, per ogni stringa binaria x, $x \in L_1$ se e solo se $f(x) \in L_2$ (x in L_1 se e solo se f(x) in L_2)

14. Una macchina di Turing è un modello di calcolo abbastanza simile agli odierni calcolatori. Essa possiede una CPU e una memoria su cui poter leggere e scrivere. In particolare, la CPU di una macchina di Turing è composta da un registro di stato, contenente lo stato attuale della macchina, e da un programma contenente le istruzioni che essa deve eseguire. La memoria di una macchina di Turing è composta da un nastro infinito, suddiviso in celle e al quale la CPU può accedere

attraverso una testina di lettura/scrittura.
Inizialmente, il nastro contiene la stringa di input preceduta e seguita da una serie infinita di simboli vuoti "□", la testina è posizionata sul primo simbolo della stringa di input e la CPU si trova nello stato iniziale. Sulla base dello stato in cui si trova la CPU e del simbolo letto dalla testina, la macchina esegue un'istruzione del programma. La macchina prosegue nell'esecuzione del programma fino a quando la CPU non viene a trovarsi in uno di un insieme di stati particolari, detti stati finali: (altrimenti ervove). Stringa ottenuta quando la CPU è in uno stato finale = stringa di output.