INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR

Stamp / Signature of the Invigilator

EXAMINATION (End Semester)	**SEMESTER (Spring 2025-2026)**

Roll Number										**Section**		**Name**	

Subject Number	C	S	2	1	2	0	4	**Subject Name**	*FORMAL LANGUAGE AND AUTOMATA THEORY*

Department / Center of the Student		**Additional sheets**	

Important Instructions and Guidelines for Students

1. You must occupy your seat as per the Examination Schedule/Sitting Plan.
2. Do not keep mobile phones or any similar electronic gadgets with you even in the switched off mode.
3. Loose papers, class notes, books or any such materials must not be in your possession, even if they are irrelevant to the subject you are taking examination.
4. Data book, codes, graph papers, relevant standard tables/charts or any other materials are allowed only when instructed by the paper-setter.
5. Use of instrument box, pencil box and non-programmable calculator is allowed during the examination. However, exchange of these items or any other papers (including question papers) is not permitted.
6. Write on both sides of the answer script and do not tear off any page. **Use last page(s) of the answer script for rough work.** Report to the invigilator if the answer script has torn or distorted page(s).
7. It is your responsibility to ensure that you have signed the Attendance Sheet. Keep your Admit Card/Identity Card on the desk for checking by the invigilator.
8. You may leave the examination hall for wash room or for drinking water for a very short period. Record your absence from the Examination Hall in the register provided. Smoking and the consumption of any kind of beverages are strictly prohibited inside the Examination Hall.
9. Do not leave the Examination Hall without submitting your answer script to the invigilator. **In any case, you are not allowed to take away the answer script with you.** After the completion of the examination, do not leave the seat until the invigilators collect all the answer scripts.
10. During the examination, either inside or outside the Examination Hall, gathering information from any kind of sources or exchanging information with others or any such attempt will be treated as **'unfair means'**. Do not adopt unfair means and do not indulge in unseemly behavior.

Violation of any of the above instructions may lead to severe punishment.

Signature of the Student

To be filled in by the examiner

Question Number	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Total**
Marks Obtained											

Marks obtained (in words)	**Signature of the Examiner**	**Signature of the Scrutineer**

Indian Institute of Technology Kharagpur
Department of Computer Science and Engineering

Formal Language and Automata Theory (CS21204) **Spring 2025-2026**

21-April-2026 **End-Semester Examination** Maximum Marks: 60

Instructions:

- Write your answers in the question paper itself. Be brief and precise. Answer *all* questions.
- Write the answers only in the respective spaces provided. The last three blank pages may be used for rough work or leftover answers.
- In case you may need more space/pages, please ask for additional sheets in the exam hall and attach the same with this booklet while submitting.
- If you use any algorithm / result / formula covered in the class, just mention it, do not elaborate (unless the same thing has been explicitly asked to answer in the question).

— Question Paper Starts from Next Page —

Q1. [Turing Machines and Unrestricted Grammars] **15 marks**

(a) Write an unrestricted grammar to generate the following language:

$$L_1 = \{a^i b^j c^k d^l \mid i = k \text{ and } j = l\}, \quad \text{where } i, j, k, l \text{ are non-negative integers.}$$

Mention the start symbol of your grammar. Use upper-case letters for non-terminal symbols. **(5)**

Solution:

The following grammar with start symbol S generates L_1:

S	$\rightarrow$	$aSC \mid T$	[Generate $a^i TC^k$ with $i = k$]
T	$\rightarrow$	$bTd \mid R$	[Generate $a^i b^j Rd^l C^k$ with $i = k$ and $j = l$]
dC	$\rightarrow$	Cd	[Allow C to move past d]
RC	$\rightarrow$	cR	[C is converted to c after it reaches the correct place]
R	$\rightarrow$	ε	[After R converts all C's to c's, it vanishes]

(b) Show a derivation of the string $a^2b^3c^2d^3$ according to your grammar. Clearly mention the exact production that you have used in each step of your derivation. **(5)**

Solution:

The derivation of $a^2b^3c^2d^3$ and the rules used in the derivation process are given below.

$$
\begin{aligned}
S &\Longrightarrow aSC && [S \to aSC] \\
&\Longrightarrow aaTCC && [S \to T] \\
&\Longrightarrow aabTdCC \Longrightarrow aabbTddCC \Longrightarrow aabbbTdddCC && [T \to bTd] \\
&\Longrightarrow aabbbRdddCC && [T \to R] \\
&\Longrightarrow aabbbRddCdC \Longrightarrow aabbbRdCddC \Longrightarrow aabbbRCdddC && [dC \to Cd] \\
&\Longrightarrow aabbbcRdddC && [RC \to cR] \\
&\Longrightarrow aabbbcRddCd \Longrightarrow aabbbcRdCdd \Longrightarrow aabbbcRCddd && [dC \to Cd] \\
&\Longrightarrow aabbbccRddd && [RC \to cR] \\
&\Longrightarrow aabbbccddd && [R \to \varepsilon] \\
&\equiv a^2b^3c^2d^3
\end{aligned}
$$

(c) Present an informal algorithmic description of a Turing Machine that *decides* L_1. **(5)**

Solution:

Here is an informal description for the (total) Turing Machine:

Step 0: Make a pass to check whether the input string is of the form $a^*b^*c^*d^*$. If not, then *reject and halt*; otherwise proceed to **Step 1**.

Step 1: Repeat the following until no more a's and c's remain to be uncrossed:

- Make a pass to cross every uncrossed a with a corresponding uncrossed c.
- If during this pass, there remains uncrossed a's but no more uncrossed c's or uncrossed c's but no more uncrossed a's, then *reject and halt*, else goto **Step 1**.

Step 2: Repeat the following until no more b and d remain to be uncrossed:

- Make a pass to cross every uncrossed b with a corresponding uncrossed d.
- If during this pass, there remains uncrossed b's but no more uncrossed d's or uncrossed d's but no more uncrossed b's, then *reject and halt*, else goto **Step 2**.

Step 3: If no input symbol is left to be uncrossed in the tape, then *accept and halt*.

Q2. [Decidable Languages] **14 marks**

Prove that the following languages are decidable. Provide only high-level descriptions of deciders.

(a) $\texttt{EMPTY}_{\texttt{DFA}} = \{\langle A \rangle \mid A \text{ is a DFA and } \mathscr{L}(A) = \phi\}$. **(3)**

Solution:

We can design the behavior of a (total) TM, T, as follows:

T = "On input $\langle A \rangle$ where A is a DFA:

- Mark the start state of A.
- Repeat until no new states get marked:
  Mark any state that has a transition coming into it from any state already marked.
- If no accept state is marked, then *accept*; otherwise *reject*.

(b) $\texttt{SUBSET}_{\texttt{DFA}} = \{\langle D1, D2 \rangle \mid D1,\ D2 \text{ are DFA with } \mathscr{L}(D1) \subseteq \mathscr{L}(D2)\}$. **(3)**

Solution:

We have $\mathscr{L}(D1) \subseteq \mathscr{L}(D2)$ if and only if $\mathscr{L}(D1) \setminus \mathscr{L}(D2) = \mathscr{L}(D1) \cap \overline{\mathscr{L}(D2)} = \phi$. Given $D1$ and $D2$, a DFA D can be constructed to recognize $\mathscr{L}(D1) \cap \overline{\mathscr{L}(D2)}$ (recall that regular languages are closed under complementation and intersection). Then feed the description of the DFA D to a decider for $\texttt{EMPTY}_{\texttt{DFA}}$, as proven in **Q2(a)**.

(c) $\texttt{FINITE}_{\texttt{PDA}} = \{\langle P \rangle \mid P \text{ is a PDA with } \mathscr{L}(P) \text{ being finite}\}$. **(5)**

Hint: Use the fact that $A_{PDA} = \{\langle P, w \rangle \mid P \text{ is a PDA that accepts string } w\}$ is decidable.

Solution:

Let P be a PDA, $L = \mathscr{L}(P)$, and n a pumping lemma constant for L. If L contains a string α of length $\geq n$, then the pumping lemma on α gives an infinite collection of strings each of which belongs to L. In order that L is finite we then require L to consist of no strings of length $\geq n$. However, we cannot check that this condition is satisfied by simulating the PDA P on all strings of length $\geq n$, since there are infinitely many such strings and the sequence of simulation does not halt. Assume that L is infinite and l is the minimum length of a string in L of length $\geq n$. We claim that $n \leq l \leq 2n-1$. Assume not, that is, $l \geq 2n$. Let α be a string of length l in L. The pumping lemma gives a decomposition $\alpha = \alpha_1\alpha_2\alpha_3\alpha_4\alpha_5$ so that $\beta = \alpha_1\alpha_3\alpha_5$ is in L too. We have $1 \leq |\alpha_2\alpha_4| \leq n$ by the pumping lemma. So β is again a string in L of length $\geq n$. This contradicts the choice of l (and α).

So, it suffices to check only the strings of length between n and $2n-1$. There are finitely many of them. Since A_{PDA} is decidable, a TM can check in finite time whether each of these strings belongs to $\mathscr{L}(P)$. Finally, note that the pumping lemma constant n can be computed from the description of P. For example, we may take $n = b^{|V|+2}$, where V is the set of non-terminals and b is the maximum number of symbols on the right side of a rule in a CFG equivalent to P.

(d) $\texttt{MOVE}_{\texttt{TM}} = \{\langle M, n \rangle \mid M \text{ is a TM that makes at least } n \text{ moves on some input}\}$. **(3)**

Solution:

In n moves a TM M can scan at most n cells starting from the left end. So irrespective of what the length of the input string is, M makes at least n moves if and only if it does so on a string of length $\leq n$. So simulate M for at most n steps on each input string α of length $\leq n$. If any string of length $\leq n$ is found on which M does not halt before making n moves, then *accept*, else *reject*.

Q3. [Semi-decidable Languages and Problem Reduction] **11 marks**

Consider the following language, $L_2 = \{\langle M\rangle \mid M \text{ is a TM that halts on the input } 010101\}$.

Prove the following assertions:

(a) Prove that L_2 is not recursive (i.e., not Turing-decidable). **(5)**

Hint: Reduce from membership problem, $\texttt{MP} = \{\langle N, w\rangle \mid N \text{ is a TM which accepts the input string } w\}$.

Solution:

Let us reduce non-recursive MP to L_2, that is, we convert $\langle N, w\rangle$ to an instance $\langle M\rangle$ such that M halts on 010101 if and only if N accepts w. Here is a description of M.

M = "On input v":

- If $v \neq 010101$, then *halt* (after accepting v).
- If $v = 010101$, then
  - Simulate N on v.
  - If N accepts v (and hence halts), then *halt* (after accepting v).
  - If N rejects v after halting, then go into infinite loop.

It follows that M halts on every input other than 010101. If the input is 010101, then there are three possibilities: N accepts w (after halting), N rejects w after halting, N goes to an infinite loop on w (and hence implicitly rejects w). Only in the first case, M halts on 010101. In the second case, M enters a forced infinite loop. In the third case, the simulation of N on w by M never terminates.

(b) Prove that L_2 is recursively enumerable (i.e., Turing-recognizable). **(3)**

Solution:

Simulate M on 010101.

If M halts (after accepting or rejecting), then accept.

If M does not halt on 010101, then the simulation does not stop and so $\langle M \rangle$ is anyway not accepted.

(c) Consider the complement of L_2:

$$\overline{L_2} = \left\{ \langle M' \rangle \;\middle|\; M' \text{ is a TM which does not halt on the input } 010101 \right\}.$$

Prove that $\overline{L_2}$ is not recursively enumerable (i.e., not Turing-recognizable). **(3)**

Solution:

If $\overline{L_2}$ were recursively enumerable (Turing-recognizable), then Part (b) would imply that L_2 is recursive (Turing-decidable), leading to a contradiction to Part **Q3(a)**.

Q4. [Undecidable Languages and Problem Reduction] **10 marks**

Answer the following questions using the problem reduction paradigm.

(a) Consider the following language:

$$L_3 = \{\langle M \rangle \mid M \text{ is a TM which halts on every input}\}.$$

Show a reduction from $\overline{L_2}$ to L_3 to prove that L_3 is not recursively enumerable (i.e., not Turing-recognizable). Note that the language $\overline{L_2}$ is introduced already in **Q3(c)**. **(5)**

Solution:

Let us propose a reduction from $\overline{L_2}$ to L_3 that maps $\langle M' \rangle$ to $\langle M \rangle$ such that M halts on every input if and only if M' does not halt on 010101. Here is a description of M.

M = "On input v":

- Determine the length n of the input v.
- Simulate M' on 010101 for exactly n steps.
- If the simulation halts (after accepting or rejecting 010101) within n steps, then enter an infinite loop.
- Otherwise, if the simulation does not halt, then stop the simulation and *halt* (after either accepting or rejecting v).

If M' does not halt on 010101, then irrespective of the length n of v, the simulation of M' on 010101 for n steps does not reach a halting configuration. In this case, M simply halts after aborting the simulation. On the other hand, if M' halts on 010101 after the m-th step (for $m < \infty$), then for any input v of length $n \geq m$, M enters an infinite loop and fails to halt.

Since $\overline{L_2}$ is already proven in **Q3(c)** to be not recursively enumerable, it follows that L_3 is also not recursively enumerable (i.e., not Turing-recognizable).

(b) Consider the complement of L_3:

$$\overline{L_3} = \left\{ \langle \overline{M} \rangle \;\middle|\; \overline{M} \text{ is a TM which does not halt on every input} \right\}.$$

Show a reduction from $\overline{L_2}$ to $\overline{L_3}$ to prove that $\overline{L_3}$ is also not recursively enumerable (i.e., not Turing-recognizable). Note that the language $\overline{L_2}$ is introduced already in **Q3(c)**. **(5)**

Solution:

Let us describe a reduction from $\overline{L_2}$ to $\overline{L_3}$ that maps $\langle M' \rangle$ to $\langle \overline{M} \rangle$ such that $\bar{M}$ does not halt on some input string v if and only if M' does not halt on 010101. It is natural to take $v = 010101$, so that $\overline{M}$ can simply simulate M' on input 010101. A description of $\overline{M}$ now follows:

$\bar{M}$ = "On input v":

- If $v \neq 010101$, then halt (after accepting or rejecting v).
- If $v = 010101$, then
  - Simulate M' on 010101.
  - If the simulation halts, then *halt* (after accepting or rejecting v).

Evidently, $\overline{M}$ halts on every input other than 010101. On the other hand, $\overline{M}$ halts on the input 010101 if and only if M' does so on the same input. Thus the reduction is as desired.

Finally, since $\overline{L_2}$ is already proven (in **Q3(c)**) to be not recursively enumerable, it follows that $\overline{L_3}$ too is not recursively enumerable (i.e., not Turing-recognizable).

Q5. [Rice's Theorem and Properties of R.E. Languages] **10 marks**

Which of the following are properties of recursively enumerable sets? First, supply an answer Yes/No. If the answer is *Yes*, mention whether the property is trivial and/or monotone. If the answer is *No*, supply a one-line justification. In each case, the property is specified by a Turing machine M. **(5 × 2)**

(a) M accepts ε.

Solution:

Yes, non-trivial, monotone.

(b) M (explicitly) rejects ε.

Solution:

No. We can construct examples of $\mathscr{L}(M_1) = \mathscr{L}(M_2)$ with M_1 rejecting ε, and M_2 looping on ε.

(c) M halts on ε.

Solution:

No. We can construct examples of $\mathscr{L}(M_1) = \mathscr{L}(M_2)$ with M_1 halting on ε (rejection), and M_2 not.

(d) $\mathscr{L}(M)$ is a context-free language.

Solution:

Yes, non-trivial, non-monotone.

(e) $\mathscr{L}(M)$ contains a context-free language.

Solution:

Yes, trivial, monotone.

— Question Paper Ends Here —