Q.No.	Questions	*Expected Time to complete*
1	**Lab 3: Prompt Engineering: Improving prompt and context management** **Objective:** To explore how variations in prompt structure affect the quality, completeness, and accuracy of responses from a large language model. **Requirements:** • VS Code with GitHub Copilot or Cursor API and/or Google Colab with Gemini • Tasks to be completed are as below. --- **Task 1: Conceptual Understanding in Physics** **Scenario** Suppose that you are a data assistant developer for an EdTech company that uses ChatGPT to answer student queries related to introductory physics. **Tasks to be completed** **1. Baseline Prompt Testing** Choose 5 typical user queries, for example: "Explain, What, Define, Why, What" • Run these prompts in a chat-based AI model and record the raw responses. **2. Prompt Refinement** Rewrite each query using the following strategies: • Add 5 different contexts (school level, competitive exam, real-life application, mathematical focus, conceptual focus). • Make the task explicit (e.g., "List and explain Newton's three laws with one real-world example each."). • Break the query into subtasks (definition → explanation → example). Run these prompts in a chat-based AI model and record the raw responses. **3. Evaluate Outputs** Score AI responses on a scale of 1–5 using: • Completeness • Accuracy • Relevance • Clarity Run these prompts in a chat-based AI model and record the raw responses. Present results in a comparative table. **4. Reflection** Discuss how contextual and structured prompts influenced the depth and correctness of responses. --- **Task 2: Programming Fundamentals (Python)** **Scenario** Suppose that you are a data assistant developer for an EdTech platform that supports beginner programming students. **Tasks to be completed** **1. Baseline Prompt Testing** Choose 5 common user queries, such as: • "What, Explain, How, What, Difference" Run these prompts in a chat-based AI model and record the raw responses.	Week-1 Saturday

2. Prompt Refinement

Rewrite each query by:

- Adding 5 contexts (beginner, exam-oriented, real-world analogy, syntax-focused, performance-focused).
- Making instructions explicit (e.g., "Define a Python loop and show one example for for-loop and while-loop.").
- Breaking into subtasks (definition → syntax → example → use case).

Run these prompts in a chat-based AI model and record the raw responses.

3. Evaluate Outputs

- Evaluate responses using completeness, accuracy, relevance, and clarity.
- Summarize findings in a table.

4. Reflection

Analyze, how explicit subtasks improve code correctness and explanation quality.

Task 3: Data Science and Machine Learning Concepts

Scenario

Suppose that you are a data assistant developer for an EdTech company offering data science courses.

Tasks to be completed

1. Baseline Prompt Testing

Select 5 typical queries, for example:

- "What, Explain, What, Define, What"

Run these prompts in a chat-based AI model and record the raw responses.

2. Prompt Refinement

Refine each query by:

- Adding 5 contexts (academic, industry, beginner-friendly, mathematical, interview-focused).
- Making tasks explicit (e.g., "Define supervised learning and explain it with one real-world example.").
- Breaking into subtasks (definition → types → example → limitation).

Run these prompts in a chat-based AI model and record the raw responses.

3. Evaluate Outputs

Score outputs using the given metrics and present results in a table.

4. Reflection

Reflect on how context management affects conceptual clarity in technical domains.

Task 4: Database and SQL Queries

Scenario

Suppose that you are a data assistant developer supporting students learning database systems.

Tasks to be completed

1. Baseline Prompt Testing

Choose 5 common queries, such as:

- "Explain, What, Difference, where, how"

Run these prompts in a chat-based AI model and record the raw responses.

2. Prompt Refinement

Rewrite each prompt by:

- Adding 5 contexts (theory exam, practical lab, interview prep, real-world

database, optimization focus).

- Making instructions explicit (e.g., "Explain SQL JOIN types with syntax and examples.").
- Breaking into subtasks (definition → syntax → example → use case).

Run these prompts in a chat-based AI model and record the raw responses.

3. Evaluate Outputs

Evaluate responses using the four metrics and summarize results in a comparison table.

4. Reflection

Discuss how refined prompts reduce ambiguity in technical explanations.

Task 5: General Aptitude and Logical Reasoning

Scenario

Suppose that you are a data assistant developer for an EdTech company focused on aptitude and competitive exam preparation.

Tasks to be completed

1. Baseline Prompt Testing

Select 5 user queries, for example:

- "Explain, What, Difference, where, how"

Run these prompts in a chat-based AI model and record the raw responses.

2. Prompt Refinement

Rewrite each query by:

- Adding 5 contexts (school exams, competitive exams, real-life analogy, formula-based, step-by-step solving).
- Making tasks explicit (e.g., "Define probability and solve one simple numerical example.").
- Breaking into subtasks (definition → formula → example → common mistakes).

Run these prompts in a chat-based AI model and record the raw responses.

3. Evaluate Outputs

- Score responses using completeness, accuracy, relevance, and clarity.
- Present findings in a table.

4. Reflection

Reflect on how structured prompts improve step-by-step reasoning and learner understanding.

Deliverables:

- Table of original vs. refined prompts and responses.
- Evaluation summary with reflection.
- Code or notebook if automated testing is used (optional).

Task 1: Conceptual Understanding in Physics

Given prompt 1 (Newton's First Law) :

Explain Newton's First Law for a high-school student. First define the law, then explain it in simple terms, and finally give one real-life example.

Newton's First Law of Motion

Definition:

Newton's First Law states that an object will remain at rest or continue to move in a straight line at constant speed unless an external force acts on it.

Simple Explanation:

This law means that objects do not change their state of motion by themselves. If something is not moving, it will stay still, and if it is moving, it will keep moving at the same speed and in the same direction—unless a force like a push, pull, or friction makes it change.

Real-Life Example:

When a book is kept on a table, it stays at rest until someone pushes it. Similarly, when a moving bus suddenly stops, passengers move forward because their bodies were in motion and tend to keep moving.

Given Prompt 2 (Force):

Define force for competitive exam preparation. Include its definition, SI unit, formula, and one real-world application.

Force :

Definition:

Force is a physical quantity that can change the shape, size, or state of motion of an object. It is a push or a pull acting on a body.

SI Unit:

The SI unit of force is the **newton (N)**.

Formula:

According to Newton's Second Law of Motion:

$$\text{Force}(F) = \text{mass}(m) \times \text{acceleration}(a)$$

or

$$F = m \times a$$

Real-World Application:

When a football player kicks a ball, the force applied by the foot changes the ball's state of rest and sets it into motion.

Given Prompt(Velocity):

Explain velocity with a mathematical focus. Define it, give the formula, and explain the difference between speed and velocity with an example

Velocity (Mathematical Focus)

Definition:

Velocity is the rate of change of displacement of an object with respect to time. It is a vector quantity, meaning it has both magnitude and direction.

Formula:

$$\text{Velocity}(v) = \frac{\text{Displacement}}{\text{Time}}$$

Difference Between Speed and Velocity (With Example):

- Speed measures how fast an object moves and depends only on distance covered. It is a scalar quantity (magnitude only).
- Velocity measures how fast an object moves in a specific direction, so it depends on displacement.

Example:

If a runner completes one full lap of a circular track and returns to the starting point, the distance covered is not zero, so speed is non-zero. However, the displacement is zero, so the average velocity is zero.

Given prompt (Why objects fall?) :

Explain why objects fall downwards using conceptual physics and real-life examples. Include the role of gravity.

Objects fall downwards because of a force called **gravity**.

Conceptual Explanation:

Gravity is a natural force of attraction that exists between all objects with mass. The Earth has a very large mass, so it exerts a strong gravitational pull on objects near its surface. When an object is released, gravity pulls it toward the center of the Earth, causing it to fall downward. This happens regardless of the object's mass (ignoring air resistance).

Role of Gravity:

Gravity provides a constant downward acceleration to falling objects. Near the Earth's surface, this acceleration is approximately **9.8 m/s²**, which means the object's speed increases as it falls.

Real-Life Examples:

- When you drop a pen from your hand, it falls to the ground because Earth's gravity pulls it downward.
- Rain falls from clouds to the ground due to gravity.

- A ball thrown upward eventually comes back down because gravity slows it, stops it momentarily, and then pulls it back toward the Earth.

Given prompt (Energy):

Define energy for school-level understanding. Explain its types and give one example for each type.

Energy

Definition:

Energy is the ability of an object or system to do work or cause change.

Types of Energy (with Examples):

1. **Kinetic Energy**
   - Energy possessed by an object due to its motion.
   - *Example:* A moving car has kinetic energy.
2. **Potential Energy**
   - Energy stored in an object due to its position or shape.
   - *Example:* A book kept on a shelf has potential energy.
3. **Thermal (Heat) Energy**
   - Energy related to the temperature of an object.
   - *Example:* Hot water contains thermal energy.
4. **Chemical Energy**
   - Energy stored in chemical bonds.
   - *Example:* Food stores chemical energy that our body uses.
5. **Electrical Energy**
   - Energy due to the flow of electric charges.
   - *Example:* Electricity used to light a bulb.

Conclusion:

Energy exists in different forms, and it can be converted from one form to another but cannot be created or destroyed.

Comparision Table :

Query Topic	Prompt Type	Completeness	Accuracy	Relevance	Clarity
Newton's First Law	Baseline	3	4	4	3
Newton's First Law	Refined	5	5	5	5
Force	Baseline	3	4	4	3
Force	Refined	5	5	5	4
Velocity	Baseline	2	4	3	3
Velocity	Refined	5	5	5	5
Gravity (Falling Objects)	Baseline	3	4	4	3
Gravity (Falling Objects)	Refined	5	5	5	4
Energy	Baseline	3	4	4	3
Energy	Refined	5	5	5	5

Task 2: Programming Fundamentals (Python) :

Given prompt (Variables) :
Explain Python variables for a beginner. First define what a variable is, then show basic syntax, give one simple example, and explain where variables are used in real programs.

What is a Variable in Python?
A variable is a name used to store data in a program. It allows us to save a value so we can use or change it later.

Basic Syntax
In Python, you create a variable by **assigning a value** to a name using the = symbol:

variable_name = value

Simple Example
age = 18
print(age)

Here, age is a variable that stores the value 18.

Where Are Variables Used in Real Programs?

Variables are used everywhere in programs to store:

- User input (like name, age, marks)
- Calculated results (such as total, average, score)
- Changing values in loops and conditions

For example, in a login system, variables store the username and password. In games, they track scores and levels.

Given prompt (Loops):

Explain Python loops with a beginner focus. Define loops, explain for-loop and while-loop syntax, show one example of each, and mention when loops are used.

What is a Loop in Python?

A **loop** is used to repeat a block of code multiple times without writing it again and again. Loops help save time and reduce errors.

for Loop (Syntax and Explanation)

A for loop is used when you know **how many times** the loop should run.

Syntax:

```
for variable in range(start, end):
    # code to repeat
```

Example:

```
for i in range(1, 6):
    print(i)
```

This prints numbers from 1 to 5.

while Loop (Syntax and Explanation)

A while loop is used when you want to repeat code **as long as a condition is true**.

Syntax:

```
while condition:
    # code to repeat
```

Example:

```
count = 1
while count <= 5:
    print(count)
    count += 1
```

This also prints numbers from 1 to 5.

When Are Loops Used?

Loops are used to:

- Print patterns or sequences
- Process lists of data
- Repeat tasks like checking user input
- Automate repetitive work in programs

Loops are essential in almost every real-world Python program.

Given prompt (If-Else):

Explain how if-else works in Python. Start with the concept, show syntax, give a simple example, and explain a real-world analogy.

Concept of if-else in Python

The **if-else statement** is used to make decisions in a program. It allows the program to choose between two actions based on whether a condition is true or false.

Basic Syntax

```
if condition:
    # code runs if condition is true
else:
    # code runs if condition is false
```

Simple Example

```
age = 18

if age >= 18:
    print("You are eligible to vote")
else:
    print("You are not eligible to vote")
```

If the condition age >= 18 is true, the first message is printed; otherwise, the second message is printed.

Given prompt (Functions):

Explain Python functions for exam-oriented learning. Define a function, show syntax, give one example, and explain why functions improve performance and reuse.

What is a Function in Python?

A **function** is a block of code that performs a specific task and runs only when it is called. Functions help organize code and are commonly tested in exams.

Syntax of a Function

```
def function_name(parameters):
    # code to perform task
    return value
```

Simple Example

```
def add(a, b):
    return a + b

result = add(5, 3)
print(result)
```

This function adds two numbers and returns the result.

Given prompt (List vs Tuple):

Explain the difference between list and tuple in Python. Define both, show syntax, compare mutability, and give one use case for each.

List in Python

A **list** is a collection used to store multiple values in a single variable. Lists are **mutable**, meaning their values can be changed.

Syntax:

```
my_list = [1, 2, 3, 4]
```

Tuple in Python

A **tuple** is also a collection used to store multiple values, but tuples are **immutable**, meaning their values cannot be changed after creation.

Syntax:

```
my_tuple = (1, 2, 3, 4)
```

Mutability Comparison

- **List:** Mutable (elements can be added, removed, or modified)
- **Tuple:** Immutable (elements cannot be changed)

Example:

```
my_list[0] = 10     # Allowed
# my_tuple[0] = 10 # Error
```

Evaluation of AI Outputs:

Topic	Prompt Type	Completeness	Accuracy	Relevance	Clarity
Variables	Baseline	3	4	4	3
Variables	Refined	5	5	5	5
Loops	Baseline	3	4	4	3
Loops	Refined	5	5	5	4
If-Else	Baseline	3	4	4	3
If-Else	Refined	5	5	5	5
Functions	Baseline	3	4	4	3
Functions	Refined	5	5	5	4
List vs Tuple	Baseline	2	4	3	3
List vs Tuple	Refined	5	5	5	5

Reflection:

Explicit subtasks significantly improved the quality of AI responses.
Baseline prompts produced correct but shallow answers that lacked examples and syntax clarity.
Refined prompts guided the AI to explain concepts step by step, reducing ambiguity.
Breaking tasks into definition, syntax, example, and use case improved both correctness and understanding.
This shows that structured prompts are essential for teaching beginner-level programming effectively.

Task 3: Data Science and Machine Learning Concepts

Given prompt (Data Science) :

Define data science in an academic and beginner-friendly context. Explain its main components, give one industry example, and mention one limitation.

Definition of Data Science:

Data science is an interdisciplinary field that focuses on collecting, cleaning, analyzing, and interpreting data to extract meaningful insights and support decision-making.
For beginners, it can be understood as **using data and computers to find patterns, trends, and useful information**.

Main Components of Data Science:

1. **Data Collection** – Gathering data from sources such as databases, sensors, or websites
2. **Data Cleaning** – Removing errors, duplicates, and missing values
3. **Data Analysis** – Exploring data using statistics and visualizations
4. **Machine Learning** – Building models that learn patterns from data

5. **Communication** – Presenting insights using charts, reports, or dashboards

Industry Example:
In the **e-commerce industry**, data science is used to recommend products to customers based on their browsing and purchase history (e.g., Amazon recommendations).

One Limitation:
A major limitation of data science is that **results depend heavily on data quality**. Poor or biased data can lead to incorrect or misleading conclusions.

Given prompt (Machine Learning) :
Explain machine learning for beginners and interviews. Define it, list main types, give one real-world example, and state one limitation

Definition of Machine Learning
Machine learning (ML) is a branch of artificial intelligence where computers learn patterns from data and make predictions or decisions without being explicitly programmed.
For beginners, it means **teaching computers to learn from experience (data)**.

Main Types of Machine Learning:
1. **Supervised Learning** – Uses labeled data (e.g., spam email detection)
2. **Unsupervised Learning** – Finds patterns in unlabeled data (e.g., customer clustering)
3. **Reinforcement Learning** – Learns through rewards and penalties (e.g., game-playing AI)

Real-World Example:
In **email spam filtering**, machine learning models learn from past emails to classify new emails as spam or not spam.

One Limitation
Machine learning models require **large amounts of quality data**, and poor data can lead to inaccurate predictions.

Given prompt (Supervised Learning) :
Define supervised learning with a mathematical and industry focus. Explain types, give one real-world example, and mention one limitation.

Definition of Supervised Learning

Supervised learning is a type of machine learning where a model is trained using **labeled data**, meaning each input is paired with a known output.
Mathematically, the model learns a function $f(x)$that maps input variables $x$to an output $y$by minimizing an error or loss function.

Types of Supervised Learning

1. **Classification** – Output is a category or class
    - Example: Spam vs Not Spam
2. **Regression** – Output is a continuous numerical value
    - Example: House price prediction

Industry Example
In the **banking industry**, supervised learning is used for **credit risk assessment**, where customer data is labeled as "loan approved" or "loan rejected" to train prediction models.

One Limitation
A key limitation of supervised learning is the **need for large labeled datasets**, which are expensive and time-consuming to create.

<u>Given prompt (Overfitting)</u> :
Explain overfitting in machine learning for beginners. Define it, explain why it occurs, give an example, and mention how to avoid it.

What is Overfitting?
Overfitting in machine learning happens when a model learns the training data too well, including noise and small details, and therefore performs poorly on new, unseen data.

Why Does Overfitting Occur?
Overfitting usually occurs when:

- The model is **too complex** for the amount of data
- There is **very little training data**
- The model is trained for **too many iterations**
- The data contains noise or outliers

Simple Example:
Imagine memorizing answers for a test instead of understanding the concepts.
You may score well on practice questions (training data) but perform poorly in the actual exam (new data).
This is similar to overfitting.

How to Avoid Overfitting?

- Use **more training data**
- Apply **regularization techniques**
- Use **simpler models**
- Apply **cross-validation**

Given prompt (Accuracy) :

Define accuracy in machine learning with a mathematical and interview-focused approach. Show the formula, explain its meaning, and mention one drawback.

Definition of Accuracy (Machine Learning):

Accuracy is an evaluation metric used to measure how many predictions made by a machine learning model are correct.

In interviews, it is commonly defined as the **ratio of correct predictions to total predictions**.

Mathematical Formula:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}}$$

Meaning of Accuracy:

Accuracy tells us how often the model makes the right decision overall.

For example, if a model predicts 80 correct outputs out of 100 inputs, its accuracy is **0.8 (or 80%)**.

One Drawback

Accuracy can be **misleading for imbalanced datasets**.

For instance, if 95% of data belongs to one class, a model predicting only that class can still achieve high accuracy without truly learning.

Topic	Prompt Type	Completeness	Accuracy	Relevance	Clarity
Data Science	Baseline	3	4	4	3
Data Science	Refined	5	5	5	5
Machine Learning	Baseline	3	4	4	3
Machine Learning	Refined	5	5	5	4
Supervised Learning	Baseline	3	4	4	3
Supervised Learning	Refined	5	5	5	5
Overfitting	Baseline	2	4	3	3

Overfitting	Refined	5	5	5	4
Accuracy	Baseline	3	4	4	3
Accuracy	Refined	5	5	5	5

Reflection:

Context management significantly improves conceptual clarity in data science topics.
Baseline prompts produced correct but shallow answers.
Refined prompts guided the AI to explain concepts step by step with examples and limitations.
Adding academic, industry, and interview contexts made responses more practical and accurate.
This shows that structured prompts are essential for teaching complex technical domains effectively.

Task 4: Database and SQL Queries :

What is a Database?

A **database** is an organized collection of data that is stored electronically so it can be easily accessed, managed, and updated.

For beginners, a database can be thought of as a **digital storage system** that keeps information in an orderly way.

Purpose of a Database

The main purpose of a database is to:

- Store large amounts of data efficiently
- Allow quick searching, updating, and retrieval of data
- Maintain data accuracy and consistency

Databases help manage information better than manual records or simple files.

Real-World Example

A **college management system** uses a database to store student details such as names, roll numbers, marks, and attendance.

Given prompt(SQL):

Define SQL for interview and practical lab preparation. Explain what SQL is, show basic syntax, give one example query, and mention where SQL is used.

What is SQL? (Interview + Lab Oriented)

SQL (Structured Query Language) is a standard language used to **store, retrieve, modify, and manage data** in relational databases.

In interviews, SQL is described as the language that communicates with databases to perform operations like SELECT, INSERT, UPDATE, and DELETE.

Basic SQL Syntax

```
SELECT column_name
FROM table_name
WHERE condition;
```

Example Explanation

- SELECT chooses the columns to display
- FROM specifies the table
- WHERE filters rows based on a condition

In the example above, SQL fetches names and marks of students who scored more than 80.

Where Is SQL Used?

SQL is used in:

- **Web applications** (user data, login systems)
- **Business analytics** (reports and dashboards)
- **Banking systems** (transactions and accounts)
- **Data science** (data extraction and preprocessing)

Given prompt(DELETE vs TRUNCATE):

Explain the difference between DELETE and TRUNCATE with optimization focus. Define both, show syntax, compare behavior, and give one use case for each

Definitions

DELETE

DELETE is a SQL command used to remove **specific rows** from a table using a WHERE clause.

It is a **DML (Data Manipulation Language)** operation.

TRUNCATE

TRUNCATE is a SQL command used to remove **all rows** from a table quickly.

It is a **DDL (Data Definition Language)** operation.

Feature	**DELETE**	**TRUNCATE**
Row removal	Selected rows	All rows
WHERE clause	Supported	Not supported
Speed	Slower (row-by-row)	Faster (bulk operation)
Rollback	Possible (depends on DB)	Not possible
Triggers	Fired	Not fired
Table space reset	No	Yes

Use Cases

- **DELETE:**
  Used when you need to remove **specific records**, such as deleting inactive users.
- **TRUNCATE:**
  Used when you need to **quickly clear an entire table**, such as resetting log or temporary tables for better performance.

Given prompt(WHERE Clause):

Explain the SQL WHERE clause for practical labs. Define it, show syntax, give an example query, and explain why it is used.

Definition

The **WHERE clause** in SQL is used to **filter records** in a table by specifying conditions.
It ensures that only rows matching the given condition are selected, updated, or deleted.

Syntax

SELECT column1, column2
FROM table_name
WHERE condition;
The WHERE clause can also be used with UPDATE and DELETE.

Example Query

SELECT name, marks
FROM students
WHERE marks >= 75;

Explanation:

This query retrieves only those students whose marks are **75 or higher** from the students table.

Why the WHERE Clause Is Used

- To **retrieve specific data** instead of the entire table
- To **improve query efficiency** by reducing unnecessary data processing
- To **apply conditions** in SELECT, UPDATE, and DELETE operations
- To **ensure accuracy** in database operations during practical labs

Given prompt(SQL JOIN):

Explain SQL JOIN types for interviews and real-world databases. Define JOIN, list types, show basic syntax, give one example, and mention a use case

SQL JOIN Types (Interview + Real-World Explanation)

What is JOIN?

A **JOIN** in SQL is used to **combine rows from two or more tables** based on a related column between them (usually a primary key–foreign key relationship).

Types of SQL JOINs

1. **INNER JOIN**
    - Returns only the rows that have **matching values in both tables**.
2. **LEFT JOIN (LEFT OUTER JOIN)**
    - Returns **all records from the left table** and matching records from the right table.
    - If no match exists, NULL values are returned for right table columns.
3. **RIGHT JOIN (RIGHT OUTER JOIN)**
    - Returns **all records from the right table** and matching records from the left table.
    - If no match exists, NULL values are returned for left table columns.
4. **FULL JOIN (FULL OUTER JOIN)**
    - Returns **all records when there is a match in either table**.
    - Non-matching rows from both tables are included with NULLs.

Basic Syntax

```
SELECT columns
FROM table1
JOIN table2
ON table1.common_column = table2.common_column;
```

Example Explanation

In the given query:

- employees and departments tables are joined using dept_id
- Only employees who are assigned to a department are shown
- This is a classic **INNER JOIN** example

Real-World Use Case

- **Company Databases:**
  Combine employee details with department information
- **E-commerce Systems:**
  Join orders and customers tables to see who placed which order
- **Banking Systems:**
  Join accounts and transactions to track customer activity

Topic	Prompt Type	Completeness	Accuracy	Relevance	Clarity
Database	Baseline	3	4	4	3
Database	Refined	5	5	5	5
SQL	Baseline	3	4	4	3
SQL	Refined	5	5	5	4
DELETE vs TRUNCATE	Baseline	2	4	3	3
DELETE vs TRUNCATE	Refined	5	5	5	5
WHERE Clause	Baseline	3	4	4	3
WHERE Clause	Refined	5	5	5	4
SQL JOIN	Baseline	2	4	3	3
SQL JOIN	Refined	5	5	5	5

Reflection

Refined prompts significantly reduce ambiguity in technical explanations.
Baseline prompts resulted in correct but shallow answers that lacked structure and practical relevance.
By adding context and explicit subtasks, the AI produced clearer, more complete, and exam-ready responses.
Structured prompts helped ensure inclusion of syntax, examples, and real-world use cases.
This demonstrates that prompt refinement is essential for effective learning in database and SQL education.

Task 5: General Aptitude and Logical Reasoning

Prompt (Probability):

Define probability for school and competitive exams. Show the formula. Solve a simple real-life example step-by-step. Explain one common mistake students make while calculating probability.

Definition of Probability

Probability is a measure of how likely an event is to occur. In other words, it quantifies the chance of an event happening.

- School/Exam Definition: Probability is the ratio of favorable outcomes to the total possible outcomes.
- Mathematical expression:

$$P(E) = \frac{\text{Number of favorable outcomes}}{\text{Total number of possible outcomes}}$$

Where:

- $P(E)$= probability of event E
- Favorable outcomes = outcomes we are interested in
- Total outcomes = all possible outcomes

Formula

$$\boldsymbol{P(E) = \frac{\text{favorable outcomes}}{\text{total outcomes}}}$$

- The probability ranges from 0 to 1:
  - $P(E) = 0 \rightarrow$ event cannot happen
  - $P(E) = 1 \rightarrow$ event will definitely happen

Step-by-Step Example (Real-Life)

Problem: A single six-sided die is rolled. What is the probability of getting a 4?

Step 1: Identify total outcomes

- A die has 6 faces → total outcomes = {1, 2, 3, 4, 5, 6} = 6

Step 2: Identify favorable outcomes

- We want a 4 → favorable outcome = {4} → count = 1

Step 3: Apply the formula

$$P(\text{getting 4}) = \frac{\text{favorable outcomes}}{\text{total outcomes}} = \frac{1}{6}$$

Step 4: Interpretation

- There is a 1 in 6 chance of rolling a 4.
- Probability as a decimal: 0.1667
- Probability as a percentage: 16.67%

4. Common Mistake

- Mistake: Counting outcomes incorrectly. For example, thinking there are 5 possible outcomes on a die because one number is "special" or forgetting one outcome.
- Tip: Always list all possible outcomes to avoid mistakes.

Prompt (Permutation vs Combination):

Explain permutation and combination for competitive exams. Give the formulas. Solve a numerical example step-by-step. Highlight one real-life scenario for each. Mention a common error students make.

Definition

- **Permutation (P):** Arrangement of objects **where order matters**.
- **Combination (C):** Selection of objects **where order does not matter**.

2. Formulas

- **Permutation:**

$$nPr = \frac{n!}{(n-r)!}$$

- **Combination:**

$$nCr = \frac{n!}{r!\,(n-r)!}$$

3. Numerical Example

Problem: From 3 students (A, B, C), choose 2.

- **Permutation (order matters):**

$$3P2 = \frac{3!}{(3-2)!} = \frac{6}{1} = 6$$

Arrangements: AB, BA, AC, CA, BC, CB

- **Combination (order doesn't matter):**

$$3C2 = \frac{3!}{2!\,(3-2)!} = \frac{6}{2} = 3$$

Selections: {A,B}, {A,C}, {B,C}

4. Real-Life Scenario

- **Permutation:** Assigning **first and second prizes** in a competition.
- **Combination:** Choosing **2 friends for a team** from a group of 3.

5. Common Error

- Confusing **order matters vs order doesn't matter**.
- **Tip:** Ask yourself: "Does swapping change the outcome?" → Permutation if yes, Combination if no.

Prompt (Average Speed):

Define average speed for school and competitive exams. Show the formula. Solve a simple example (car travels 60 km in 1 hour and 90 km in 1.5 hours). Explain one mistake students often make and why.

Definition

Average speed is the **total distance traveled divided by the total time taken**.

Formula

$$\text{Average Speed} = \frac{\text{Total Distance}}{\text{Total Time}}$$

Example (Step-by-Step)

Problem: A car travels 60 km in 1 hour and 90 km in 1.5 hours. Find the average speed.

Step 1: Total distance

$$60 + 90 = 150 \text{ km}$$

Step 2: Total time

$$1 + 1.5 = 2.5 \text{ hours}$$

Step 3: Average speed

$$\text{Average Speed} = \frac{150}{2.5} = 60 \text{ km/h}$$

Common Mistake

- **Mistake:** Taking the **simple mean of speeds**: (60 + 90)/2 = 75 km/h → **wrong**.
- **Reason:** Average speed depends on **total distance and total time**, not just the speeds.

Prompt (Seating Arrangement):

Explain seating arrangement problems for competitive exams. Give types (line, circle). Solve one example step-by-step with clues. Give one real-life analogy and a common pitfall.

Definition

Seating arrangement problems test **logical thinking** by arranging people/items based on given clues.

2. Types

1. **Linear arrangement:** People sit in a straight line.
2. **Circular arrangement:** People sit around a circle.

3. Example (Step-by-Step)

Problem: 4 friends — A, B, C, D — sit in a row.

Clues:

1. A sits **left of B**.

2. C sits **at the right end**.

Step 1: Draw 4 seats

_ \ _ \ _ \ _

Step 2: Place C at right end

_ \, _ \, _ \, C

Step 3: Place A and B (A left of B)

- Options: (A,B,,*C) or (*,A,B,C)

Step 4: Check possibilities

- Option 1: A,B,D,C
- Option 2: D,A,B,C

Step 5: Conclusion

- Two valid arrangements: **A,B,D,C** or **D,A,B,C**

4. Real-Life Analogy

- Arranging **students in a classroom** or **guests at a dinner table** based on preferences.

5. Common Pitfall

- **Mistake:** Ignoring the order of "left/right" or "next to" clues.
- **Tip:** Always **start with fixed positions** first, then fill others step-by-step.

Prompt (Logical Reasoning Applications):

Explain logical reasoning. Give formulaic/structured approach to identify reasoning patterns. List 3 examples in real life and competitive exams. Provide stepwise method to solve a sample puzzle. Mention common mistakes students make.

Definition

Logical reasoning = ability to analyze, deduce, and solve problems using structured thinking.

Structured Approach / Formula

1. Understand the problem/clues
2. Identify the reasoning pattern (sequence, syllogism, puzzle, analogy)
3. Organize info (tables, diagrams)
4. Deduce step-by-step
5. Verify solution

Examples

- Real life: Planning daily schedule, debugging code, making decisions
- Competitive exams: Puzzle seating arrangement, syllogisms, blood relation questions

Sample Puzzle (Stepwise)
Puzzle: 3 friends A, B, C sit in a row. A left of B. C at right end.
Step 1: Draw 3 seats: _ _ _
Step 2: Place C at right end: _ _ C
Step 3: Place A left of B: A B C

5. Common Mistakes
- Ignoring order of clues
- Making assumptions not given
- Not verifying all possibilities

Evaluate Outputs:

Query	Completeness (1-5)	Accuracy (1-5)	Relevance (1-5)	Clarity (1-5)	Notes
Baseline: Probability	2	5	3	3	Lacks examples and mistakes.
Refined: Probability	5	5	5	5	Covers definition, formula, example, mistake, tip.
Baseline: Perm vs Comb	2	5	3	3	Minimal explanation.
Refined: Perm vs Comb	5	5	5	5	Step-by-step example and context included.
Baseline: Avg Speed	2	5	3	3	No stepwise calculation shown.
Refined: Avg Speed	5	5	5	5	Stepwise example, common mistake explained.
Baseline: Seating Arrangement	2	5	3	3	Generic advice, no stepwise solving.
Refined: Seating Arrangement	5	5	5	5	Full step-by-step solution + analogy + pitfall.
Baseline:	2	5	3	3	Very brief, lacks

Logical Reasoning					examples.
Refined: Logical Reasoning	5	5	5	5	Structured method, real-life examples, common mistakes.

Reflection

- Baseline prompts: Quick, factual, sometimes correct but minimal understanding.
- Refined prompts: Provide step-by-step reasoning, context, examples, and highlight mistakes → enhances learning and exam readiness.
- Key Insight: Adding contexts + explicit subtasks guides AI to teach rather than just answer, making explanations suitable for learners preparing for exams or real-life problem-solving.

Conclusion:

Structured, stepwise prompts result in AI outputs that are educational, clear, and exam-relevant, improving comprehension and learner confidence.