📚 Day 14 Study Notes

1. Aptitude: Probability

Goal: Understand how to calculate the likelihood of an event occurring.

Core Formula

The probability of an event E occurring is defined as:

$$P(E) = \frac{\text{Number of Favorable Outcomes } (n(E))}{\text{Total Number of Possible Outcomes } (n(S))}$$

- **Range:** $0 \leq P(E) \leq 1$
- **Impossible Event:** $P(E) = 0$
- **Certain Event:** $P(E) = 1$
- **Complementary Event:** $P(\text{not } E) = 1 - P(E)$

Example Problems

1. Dice Roll:

- **Question:** What is the probability of getting an even number when rolling a standard 6-sided die?
- **Sample Space (S):** $\{1, 2, 3, 4, 5, 6\} \rightarrow n(S) = 6$
- **Favorable Event (E):** $\{2, 4, 6\} \rightarrow n(E) = 3$
- **Solution:** $P(E) = \frac{3}{6} = \frac{1}{2}$

2. Cards:

- **Question:** Probability of drawing a King from a pack of 52 cards.
- **Solution:** There are 4 Kings. $P(K) = \frac{4}{52} = \frac{1}{13}$.

2. Programming: Character Frequency

Goal: Count how many times each character appears in a string.

Logic (Hash Map Approach)

We use a dictionary (hash map) where the **Key** is the character and the **Value** is the count.

1. Initialize an empty dictionary.
2. Iterate through every character in the string.
3. If the character is already in the dictionary, increment its value.
4. If not, add it to the dictionary with a value of 1.

- **Time Complexity:** $O(N)$

Code Snippet (Python)

```
def count_char_frequency(text):
    freq_map = {}

    for char in text:
        if char in freq_map:
            freq_map[char] += 1
        else:
            freq_map[char] = 1

    return freq_map

text = "programming"
print(count_char_frequency(text))
# Output: {'p': 1, 'r': 2, 'o': 1, 'g': 2, 'a': 1, 'm': 2, 'i': 1, 'n': 1}
```

3. Concept: Python OOP (Classes & Objects)

Goal: Understand the building blocks of Object-Oriented Programming.

Definitions

- **Class:** The blueprint or template. It defines the properties (attributes) and behaviors (methods) that the objects will have.
- **Object:** An instance of a class. It is the actual entity created from the blueprint.

Key Components

- `__init__` : The constructor method. It initializes the object's attributes when the object is created.
- `self` : A reference to the current instance of the class. It allows access to variables belonging to the object.

Example

```
class Car:
    # Constructor
    def __init__(self, brand, model):
        self.brand = brand  # Attribute
        self.model = model  # Attribute

    # Method
    def display_info(self):
        print(f"This is a {self.brand} {self.model}")

# Creating Objects
car1 = Car("Toyota", "Corolla")
car2 = Car("Tesla", "Model S")
```

```
car1.display_info() # Output: This is a Toyota Corolla
```

4. C/C++ Concept: Structures vs. Classes

Goal: Compare procedural data grouping vs. Object-Oriented encapsulation.

C Structures (`struct`)

- **Procedural:** Primarily used to group data (variables) together.
- **Visibility:** In C, everything is public. In C++, members are **public** by default.
- **Usage:** Best for Plain Old Data (POD) types without complex logic.

C++ Classes (`class`)

- **Object-Oriented:** Designed to bundle data *and* behavior (methods) while hiding implementation details.
- **Visibility:** Members are **private** by default (Encapsulation). You must explicitly verify `public` sections.
- **Usage:** Used for defining complex entities with state and behavior.

Feature	`struct` (C++)	`class` (C++)
Default Access	Public	Private
Default Inheritance	Public	Private
Concept	Data Holder	Data + Behavior + Encapsulation

5. SQL: UPDATE Command

Goal: Modify existing records in a table.

Syntax

```
UPDATE table_name
SET column1 = value1, column2 = value2, ...
WHERE condition;
```

⚠️ Critical Warning

Always use the `WHERE` clause. If you omit it, **ALL** records in the table will be updated.

Examples

1. Update a specific record: Change the city of the employee with ID 101.

```
UPDATE Employees
SET City = 'New York'
WHERE EmpID = 101;
```

2. **Update multiple columns:** Give a raise and change the job title for 'Alice'.

```
UPDATE Employees
SET Salary = 60000, JobTitle = 'Senior Developer'
WHERE Name = 'Alice';
```