Week 2: Industrial Training Report

Overview

The second week of the industrial training at **Auribises Technologies Pvt. Ltd.** focused on strengthening our understanding of Python programming through practical implementation of control structures, bitwise operations, looping mechanisms, user-defined functions, and object-oriented programming. Each day involved detailed exercises and small projects to enhance logic-building and problem-solving abilities.

Day 6: Bitwise Shift Operations, Conditional Structures, and Pattern Printing

The sixth day of the training began with learning **bitwise shift operators**, which are widely used in low-level programming and encryption processes. We explored the **right shift (>>) and left shift (<<)** operators and understood their real-world use in **encryption and decryption** in computer networks.

Topics Covered:

- **Bitwise Shift Operators:** Used for manipulating bits in numbers; left shift doubles the value while right shift halves it.
- **Encryption Using Shifts:** Applying bitwise operations to encode and decode data.
- **Conditional Structures:** Implementation of nested if-else and ladder if-else using a promo code example.
- **Linear Search Algorithms:** Searching through data using both for and while loops.
- **Pattern Printing:** Creating a chessboard pattern using nested loops.

Example 1: Bitwise Shift Operation

```
# Left and Right Shift Example
num = 8  # binary 1000
print(num >> 1)  # Right Shift: 4
print(num << 1)  # Left Shift: 16
```

Example 2: Coupon-Based Discount Using Ladder If-Else

```
amount = int(input("Enter your Zomato order amount: "))
if amount > 500:
    print("You get 25% discount!")
elif amount > 300:
    print("You get 15% discount!")
else:
    print("No discount, minimum order not met.")
```

Example 3: Linear Search (For and While Loop)

```
# Linear Search using For Loop
data = [10, 20, 30, 40, 50]
key = 30
found = False
for i in data:
    if i == key:
        found = True
        break
print("Found" if found else "Not Found")

# Linear Search using While Loop
index = 0
while index < len(data):
    if data[index] == key:
        print("Found at position", index)
        break
    index += 1
```

Example 4: Chessboard Pattern

```
for i in range(8):
    for j in range(8):
        if (i + j) % 2 == 0:
            print("⬛", end=" ")
        else:
            print("⬜", end=" ")
    print()
```

Day 7: Advanced Loops, Indexing, and Function Introduction

The seventh day emphasized mastering loop-based problems and understanding indexing mechanisms. We practiced **for loop variations**, explored **positive and negative indexing**, and worked on programs to calculate maximum scores using arrays. The concept of **functions** was introduced to modularize code.

Topics Covered:

- **Loop Practice:** Implementing loops for multiple real-life problems.
- **Finding Maximum in a List:** Using loops and conditional logic.
- **Positive and Negative Indexing:** Accessing list elements from both ends.
- **Introduction to Functions:** Understanding the purpose of defining and calling reusable blocks of code.

Example 1: Maximum Score in IPL

```
scores = [120, 145, 178, 156, 189]
max_score = scores[0]
for score in scores:
    if score > max_score:
        max_score = score
print("Highest IPL Score:", max_score)
```

Example 2: Positive and Negative Indexing

```
players = ["Rohit", "Virat", "Dhoni", "Gill", "Bumrah"]
print(players[0])   # Positive Index: Rohit
print(players[-1])  # Negative Index: Bumrah
```

Example 3: Simple Function Example

```
def greet():
    print("Welcome to Python Training!")

greet()
```

Day 8: Function Creation and Magic Variables

Day eight involved a deep dive into **function creation**, including defining, calling, and returning results. We practiced writing functions for multiplication tables, addition, and squares of numbers. Later, we explored **magic (dunder) variables** and learned how the Python interpreter uses them during program execution.

Topics Covered:

- **Function Components:** Defining, calling, and returning values.
- **Parameter Passing:** Understanding arguments and parameters.
- **Practical Functions:** Creating functions for arithmetic operations.
- **Magic Variables (`__name__`):** Understanding Python's execution flow.

Example 1: Function to Print Multiplication Table

```
def print_table(num):
    for i in range(1, 11):
        print(f"{num} x {i} = {num*i}")

print_table(5)
```

Example 2: Function Returning Square

```
def square(n):
    return n * n

print("Square of 7:", square(7))
```

Example 3: Magic Variable Example

```
def main():
    print("Code is running directly")

if __name__ == "__main__":
    main()
```

Day 9: Object-Oriented Programming (OOP) - Classes and Objects

The ninth day focused on **Object-Oriented Programming** concepts. We learned how to define **classes and objects**, implement constructors using `__init__`, and understand the role of attributes and methods. Examples were practiced through playlist and song-based structures.

Topics Covered:

- **OOP Introduction:** Understanding the concept of objects and classes.
- **Constructors (`__init__`):** Automating object initialization.
- **Attributes and Methods:** Defining class-level and instance-level properties.
- **Default Arguments:** Handling function arguments with default values.

Example 1: Simple Class Example

```
class Song:
    def __init__(self, title, artist):
        self.title = title
        self.artist = artist

    def play(self):
        print(f"Now playing: {self.title} by {self.artist}")

song1 = Song("Perfect", "Ed Sheeran")
song1.play()
```

Example 2: Default Argument Example

```
def add(a=10, b=20):
    return a + b

print("Sum:", add())
print("Sum with arguments:", add(5, 15))
```

Day 10: Mini Project and Application Practice

The final day of Week 2 was dedicated to a small project applying all the learned concepts. We revisited dictionary-based structures and created a **Flight Booking System**, integrating loops, conditions, and user input.

Topics Covered:

- **Project Implementation:** Building a flight booking application.
- **Using Dictionaries and Functions Together.**
- **Applying Conditionals and Loops in Real Contexts.**

Example: Flight Booking System

```
flights = {
    'AI101': {'source': 'Delhi', 'destination': 'Mumbai', 'price': 4500},
    'AI202': {'source': 'Chandigarh', 'destination': 'Bangalore', 'price': 6200}
}

code = input("Enter flight code: ")
if code in flights:
    details = flights[code]
    print(f"Flight from {details['source']} to {details['destination']} costs Rs.{details['price']}")
else:
    print("Invalid flight code.")
```

Summary

Week 2 provided a deeper understanding of Python through applications of bitwise operations, control structures, looping mechanisms, functions, and object-oriented programming. Practical exercises like promo code discounts, IPL score analysis, and the flight booking mini-project enhanced the learning experience by connecting programming logic with real-world examples.