Outlines & Practice Questions for Final Exam of Programming Fundamentals 23 Batch AI Department
Subject Teacher: Muhammad Owais Rehmani

Outlines:

1. Programming Fundamentals
    a. Programming language
    b. Types of programming Language
    c. Python Programming Language
    d. Importance of Python in AI

2. Variables, Datatypes, Expressions, & All types of Operators

3. Control Flow (If elif else, nesteed if else, Loops, Nesteed Loops, for and while and both)

4. Python Data structures (Lists and Dictionaries)

5. Python Functions

Conceptual Theoretical Questions:

2. Briefly explain the difference between a compiled and an interpreted programming language. When would you choose one over the other?
3. Compare and contrast imperative, functional, and object-oriented programming paradigms. Give an example of a language in each category.
4. Why Python is considered an ideal language for beginners? Discuss any two features that contribute to this.
5. Explain three reasons why Python is widely used in the field of Artificial Intelligence.
6. What is the difference between variable declaration and initialization in Python? Provide an example of each.
7. Differentiate between arithmetic and logical operators in Python. Give an example of each type with an explanation.
8. Describe the purpose of a nested loop in Python. Provide a scenario where a nested loop would be advantageous over a single loop.
9. Explain the concept of indexing and slicing for accessing elements within a Python list. Demonstrate how they work with an example.
10. Differentiate between a list and a dictionary in Python. When would you use a dictionary over a list?
11. Explain the concept of modularity and how functions promote it in Python code. Give an example of a function definition and call.

Basic Programming Questions:

1. Write a simple program that prompts the user for their name and then prints a greeting message that includes their name.
2. Briefly describe the following terms in the context of programming languages: syntax, semantics, and keywords.
3. Write a Python program that prints the following message to the console: "Hello, World!"
4. List three areas within Artificial Intelligence where Python is commonly used.
5. Declare two variables in Python: one to store your age (integer) and another to store your name (string). Assign appropriate values to each variable and then print them both to the console.
6. Write a Python expression that calculates the area of a rectangle with a width of 5 and a height of 3. Use the appropriate operator(s).
7. Write a Python program that prompts the user for a number and then checks if the number is even or odd. Print an appropriate message based on the result.
8. Write a Python program that prints the numbers from 1 to 10 (inclusive) using a for loop.
9. Create a Python list containing the names of your three favorite fruits. Then, access and print the second element in the list.
10. Create a Python dictionary to store information about a book, including its title, author, and year of publication. Access and print the title of the book from the dictionary.
11. Write a simple Python function that takes two numbers as arguments and returns their product (multiplication). Call the function with specific values and print the returned result.

Medium Level Programming Problems:

1. Budget Calculator:
   - Write a program that asks the user for their monthly income and expenses.
   - Calculate the difference between income and expenses.
   - If the difference is positive, display a message indicating how much the user can save.
   - If the difference is negative, display a message indicating how much the user is overspending.

2. Simple Quiz:
   - Create a program that asks the user 5 multiple-choice questions with three answer options each.
   - Store the correct answers in a separate list.
   - After the user answers all questions, calculate the user's score and display a message indicating their performance.

3. Movie Recommendation System:

- Design a program that asks the user for their favorite movie genre (e.g., comedy, action, drama).
- Based on the user's input, recommend a few movies from that genre using a list of pre-defined movie titles and genres.

4. Shopping Cart:
   - Simulate a basic shopping cart program.
   - Allow the user to add items (e.g., product name, price) to the cart.
   - Provide options to remove items and view the total cost of the cart.

5. Password Generator:
   - Create a program that generates a random password for the user.
   - The password should include a combination of uppercase and lowercase letters, numbers, and special characters.
   - Allow the user to specify the desired password length.

6. Simple Text Analyzer:
   - Write a program that reads a string from the user and analyzes it.
   - Calculate and display the number of characters, words, and sentences in the string.

7. Temperature Converter:
   - Develop a program that allows the user to convert temperatures between Celsius and Fahrenheit.
   - Provide options for the user to choose the conversion direction (Celsius to Fahrenheit or vice versa) and enter the temperature value.

8. BMI Calculator:
   - Design a program that calculates the user's Body Mass Index (BMI) based on their weight and height.
   - Prompt the user to enter their weight (in kilograms) and height (in meters).
   - Calculate the BMI using the formula: BMI = weight / (height * height).
   - Display the calculated BMI and interpret it based on the standard BMI categories (underweight, normal weight, overweight, obese).

Real-World Scenario Based Programming Questions:

1. **Restaurant Menu Management:** Develop a program for a restaurant to manage its menu items. It should allow adding, removing, and updating menu items with details like name, price, description, and category (e.g., appetizer, main course, dessert). The program should also display the complete menu and filter items based on category.
2. **Library Management System:** Create a program for a library to manage its books and users. It should allow adding new books with details like title, author, ISBN, and availability. The program should also register new users, track borrowed and returned books, and display available books based on various criteria like title, author, or genre.
3. **Event Registration System:** Design a program to manage registrations for an event. It should allow users to register by providing their name, email, and any additional required information. The program should track the number of registered participants, allow organizers to view registrations, and send confirmation emails to attendees.
4. **Simple Online Shop:** Develop a program to simulate a basic online shop for selling products. It should display a list of available products with details like name, price, and quantity. The program should allow users to add items to their cart, view cart contents, and calculate the total cost.
5. **Task Management App:** Create a program to help users manage their tasks. It should allow adding new tasks with details like title, description, due date, and priority level. The program should display a list of tasks, allow marking tasks as completed, and filter tasks based on different criteria like due date or priority.
6. **Simple Travel Booking System:** Design a program to help users book flights. It should allow users to search for flights based on origin, destination, and travel dates. The program should display available flights with details like airline, departure and arrival times, and price. Users should be able to choose a flight and book it (simulating the process).
7. **Basic Music Player:** Develop a program to act as a simple music player. It should allow users to browse and select music files from their device. The program should play the selected music file, pause or stop playback, and adjust the volume.
8. **Simple Quiz with Timer:** Create a program for a timed quiz. It should display a set of questions with multiple-choice answers and a timer for each question. Users should select their answers within the time limit. The program should calculate the score and display the results upon completion.
9. **Simple Social Media Feed:** Design a program to represent a basic social media feed. It should allow users to post messages and view posts from other users (simulated data). Users should be able to like and comment on posts.
10. **Basic Expense Tracker:** Develop a program to help users track their daily expenses. It should allow adding new expenses with details like date, category (e.g., food, transportation, entertainment), and amount. The program should categorize expenses, calculate total spending for different categories, and display statistics over a specified period.

Scenario-Based Programming Questions:

1. Movie Recommendation System:

Imagine you work for a streaming service that wants to personalize movie recommendations for its users. Develop a program that:

- Asks the user for their age and favorite movie genre (e.g., comedy, action, drama).
- Based on the user's age and genre preference, recommends at least three specific movies from the service's library using pre-defined data (including title, genre, and release year).
- If the user has no specific genre preference, recommend a mix of popular movies from different genres.

2. Restaurant Order Management System:

Design a program for a restaurant's wait staff to manage customer orders. It should allow:

- Selecting a table number to access the order for that specific customer.
- Adding items from a pre-defined menu (including name, price, and category) to the order.
- Modifying or removing items from the order.
- Calculating the total cost of the order, including any applicable taxes.
- Printing a receipt for the customer.

3. Personal Budget Tracker:

Develop a program to help users manage their personal budget. It should allow:

- Setting up categories for different types of expenses (e.g., rent, groceries, transportation).
- Adding income and expenses with details like category, date, and amount.
- Generating reports that:
    - Track income and expenses over a chosen period (e.g., monthly, weekly).
    - Analyze spending by category to identify areas where the user can save.
    - Compare income and expenses to calculate the user's overall financial health.

4. Smart Home Automation System:

Imagine you are creating a program for a smart home device. It should be able to:

- Control various smart home features based on user input or pre-programmed schedules, such as:
    - Adjusting lighting (on/off, brightness, color)
    - Setting room temperature

    - Locking/unlocking doors
- Respond to user voice commands for controlling these features.

5. E-commerce Product Search and Filtering:

Develop a program for an e-commerce website's product search functionality. It should allow users to:

- Search for products using keywords or browsing by category (e.g., electronics, clothing, books).
- Filter search results based on various criteria, such as:
    - Price range
    - Brand
    - Product features
    - Customer ratings

6. Simple Fitness Tracker:

Design a program to work with a wearable fitness tracker. It should:

- Track daily steps, distance walked, and calories burned based on user activity data.
- Allow users to set daily activity goals and monitor their progress.
- Generate reports that:
    - Compare daily activity levels over a period of time (e.g., weekly, monthly).
    - Calculate trends in user's fitness progress.

7. Online Learning Platform:

Imagine you are creating a program for an online learning platform. It should allow:

- Users to register and create accounts.
- Users to enroll in different courses offered on the platform (including title, description, and instructor information).
- Accessing course materials such as video lectures, quizzes, and assignments.
- Submitting completed assignments and receiving feedback from instructors.

8. Basic Customer Service Chatbot:

Develop a program to simulate a simple chatbot for a customer service application. It should be able to:

- Greet users and understand their basic inquiries through natural language processing.
- Provide answers to frequently asked questions based on a predefined knowledge base.
- Offer basic troubleshooting steps for common issues.

- Route users to a human customer service representative for more complex inquiries.

9. Smart Traffic Light System:

Imagine you are creating a program for a smart traffic light system. It should be able to:

- Monitor traffic flow on different roads using sensors.
- Adjust traffic light timings dynamically based on real-time traffic congestion to optimize traffic flow and reduce wait times.
- Prioritize emergency vehicles by giving them immediate right-of-way.

10. Social Media Sentiment Analysis:

Develop a program that analyzes public sentiment on social media platforms. It should be able to:

- Collect tweets or posts related to a specific topic or brand.
- Analyze the sentiment of the collected text data (positive, negative, or neutral) using natural language processing techniques.
- Generate reports that:
    - Summarize the overall sentiment towards the topic or brand.
    - Identify common themes and opinions expressed in the text data.