**Week 1: Introduction to Computational Thinking & Python**

1. Which of the following is *not* part of computational thinking?
   1. Memorization
   2. Abstraction
   3. Pattern Recognition
   4. Decomposition
2. What does print("Hello, world!") output?
3. Write a program that prints your name and favorite hobby.
4. Write a Python program that prints your name, your school, and your favorite subject.
5. Break the problem of making a sandwich into four computational steps. Write them as comments in Python.
6. Write a program that prints your name 3 times.

**Week 2: Variables, Data Types, and Input/Output**

1. What is the result of type(3.14)?
2. What does the input/print combo do?

age = input("Enter your age: ")

print("Next year you will be", int(age) + 1)

1. Write a program that greets the user using input.
2. Write a program that asks the user for their favorite number and multiplies it by 2.
3. Ask the user for their name and then print a custom welcome message.
4. Ask the user for their height in inches and convert it to centimeters (1 inch = 2.54 cm).

**Week 3: Operators and Expressions**

1. What is 3 + 4 \* 2?
2. What is 10 > 5 and 2 < 1?
3. Write a triangle area calculator using base and height.
4. Write a program to compute the average of three test scores.
5. Ask the user for two numbers and compute the remainder.
6. Determine whether a number is even or odd.

**Week 4: Conditional Statements**

1. Purpose of elif?
2. Predict result of conditional structure

x = -3

if x > 0:

print("Positive")

elif x == 0:

print("Zero")

else:

print("Negative")

1. Write a letter grade calculator.
2. Ask for a number and determine if it's positive, negative, or zero.
3. Convert numeric score to letter grade (A–F).
4. Check if a year is a leap year.

**Week 5: Loops - Part 1 (While loops)**

1. Which loop continues until a condition is false?
   * A. for
   * B. while
   * C. if
   * D. def
   * **Answer:** B

2. What does this code do?

i = 5

while i > 0:

print(i)

i -= 1

* **Answer:** Prints 5 down to 1

3. Write a guessing game where the user guesses a secret number.

**Week 6: Loops - Part 2 (For loops)**

1. What does range(1, 6) produce?
   * A. [1, 2, 3, 4, 5]
   * B. [0, 1, 2, 3, 4, 5]
   * C. [1, 2, 3, 4, 5, 6]
   * D. [0, 1, 2, 3, 4]
   * **Answer:** A

2. What does this do?

for char in "hi":

print(char)

* **Answer:** Prints h then i

3. Write a program that prints all even numbers from 1 to 20.

**Week 7: Functions and Modular Programming**

1. What is a benefit of using functions?
   * A. Makes code faster
   * B. Eliminates variables
   * C. Improves readability and reuse
   * D. Makes loops unnecessary
   * **Answer:** C

2. What does this return?

def square(n):

return n \* n

print(square(4))

* **Answer:** 16

3. Write a function that returns True if a number is even.

**Week 9: Problem Solving with Lists**

1. What does list.count(x) do?
2. Output of list comprehension? [x\*x for x in [1, 2, 3]] → [1, 4, 9]
3. Write a program to find all elements greater than 10 in a list.
4. Write a function that counts how many times an element appears in a list.
5. Use list comprehension to make a list of squares from 1 to 20.
6. Write a program to find the second-largest number in a list.

**Week 10: Strings and Text Processing**

1. len("Hello")?
2. "Ada".upper()?
3. Count vowels in a sentence.
4. Count how many vowels are in a user-input sentence.
5. Format a user's full name as "Last, First".
6. Replace all spaces in a string with dashes.

**Week 11: Introduction to Dictionaries**

**Practice Problems**

1. What is a dictionary?
2. Output of phone["Bob"]?
3. Use a dictionary to count word frequency.
4. Create a phone book that stores and retrieves names and numbers.
5. Count the frequency of each word in a user-input sentence.
6. Write a program that checks if a key exists in a dictionary.

**Week 12: Algorithms and Algorithmic Thinking**

1. What is linear search?
2. Output of search([1,2,3],2)?
3. Write pseudocode for bubble sort.
4. Implement linear search.
5. Write pseudocode for a sorting algorithm (e.g., selection sort).
6. Ask the user to enter numbers and sort them using Python’s sorted().

**Week 13: Debugging and Testing**

1. What is a NameError?
2. prnt("Hi")
3. Write a test case for add(a, b).
4. Create a buggy version of a program and fix the errors.
5. Write unit tests for a function that multiplies two numbers.
6. Use assert to test correctness of a function.

**Week 14: Introduction to Recursion**

1. What is recursion? — B (Function calling itself)
2. fact(3)? — 6
3. Recursive Fibonacci.
4. Write a recursive factorial function.
5. Write a recursive function to compute the nth Fibonacci number.
6. Use recursion to sum the numbers from 1 to n.