Python Basics Assignment

Objective: Create a Python script for each question. Write clear comments to explain your code. Test your code to ensure it works as expected. Write your scripts as `.py` files.

---

1. Variables and Data Types

Question 1: Variable Manipulation

- Assign your birth year, your height in centimeters, your name, and a boolean indicating whether you are employed to variables.
- Print each variable along with its data type.

File Name: `variable_manipulation.py`

---

2. Control Structures

Question 2: Number Analysis

- Ask the user for a number.
- Use an `if` statement to determine if the number is even or odd.
- Use a `for` loop to print all even numbers up to the user's number.
- Use a `while` loop to sum the digits of the number.

File Name: `number_analysis.py`

---

3. Functions

Question 3: Temperature Conversion

- Write a function `convert_to_fahrenheit` that takes a temperature in Celsius and returns it in Fahrenheit.
- Write a function `convert_to_celsius` that takes a temperature in Fahrenheit and returns it in Celsius.
- Create a script to test these functions with various temperatures.

File Name: `temperature_conversion.py`

---

4. Lists and Dictionaries

Question 4: Shopping Cart

- Create a list of items you need to buy and print it.
- Add three more items to the list and print the updated list.
- Create a dictionary representing a product in an online store (name, price, in_stock) and print it.
- Add a key-value pair for a discount and print the updated dictionary.

File Name: `shopping_cart.py`

---

5. Error Handling

Question 5: Safe Division

- Write a function `safe_divide` that takes two numbers and returns their division.
- Use `try-except` to handle potential division by zero errors.

- Prompt the user for two numbers and use the `safe_divide` function. Print a message if an error occurs.

File Name: `safe_divide.py`

---

6. Strings

Question 6: String Manipulation

- Write a function `string_analysis` that takes a string and returns the number of vowels, consonants, and words in it.
- Test the function with various strings.

File Name: `string_manipulation.py`

---

7. Files

Question 7: File Operations

- Write a script that reads a text file and prints the content.
- Write a function `count_lines_words` that reads a text file and returns the number of lines and words.
- Create a text file and test your functions.

File Name: `file_operations.py`

---

8. Classes

Question 8: Person Class

- Create a class `Person` with attributes `name`, `age`, and `email`.
- Add a method `display_info` that prints a person's details.
- Create an instance of the `Person` class and display its information.

File Name: `person_class.py`

---

9. Modules

Question 9: Math Utilities

- Write a module `math_utils` that contains functions for calculating the factorial, greatest common divisor, and least common multiple.
- Create a script to import and test these functions.

File Name: `math_utilities.py`

---

10. Regular Expressions

Question 10: Email Validation

- Write a function `validate_email` that checks if a given string is a valid email using regular expressions.
- Create a script to test this function with various email addresses.

File Name: `email_validation.py`

---

11. Recursion

Question 11: Recursive Sum

- Write a recursive function `recursive_sum` that calculates the sum of numbers from 1 to `n`.
- Test the function with different values of `n`.

File Name: `recursive_sum.py`

---

12. List Comprehensions

Question 12: List Transformation

- Create a list of numbers from 1 to 20.
- Use a list comprehension to generate a new list with the squares of the numbers.
- Print the new list.

File Name: `list_transformation.py`

---

13. Lambda Functions

Question 13: Sorting with Lambda

- Write a script that sorts a list of tuples containing names and ages in descending order of age using a lambda function.
- Print the sorted list.

File Name: `lambda_sorting.py`

---

14. Decorators

Question 14: Execution Time

- Write a decorator `execution_time` that prints the time taken by a function to execute.
- Apply this decorator to a function that finds the sum of all numbers from 1 to 1000000.

File Name: `execution_time.py`

---

15. Comprehensions and Generators

Question 15: Prime Numbers Generator

- Write a generator `prime_generator` that yields prime numbers up to a given limit.
- Create a script to print the first 10 prime numbers using this generator.

File Name: `prime_generator.py`

Submission Instructions:

- Format: Submit each part of the assignment as a separate `.py` file.
- File Names: Use the specified names for each script.
