Task 1: Basic Variables and Printing

Create a variable 'age' and assign it the value 25.
Create a string variable 'name' and assign it your name.
Print both variables.

Task 2: Type Conversion

Create a variable 'length' with value 10.5.
Convert it to an integer and store it in 'int_length'.
Print the type of both variables.

Task 3: Formatted Printing

Create variables 'x', 'y', and 'z' with values 5, 10.5, and 'Hello'.
Print them in a single print statement as 'x: 5, y: 10.5, z: Hello'.

Task 4: Simple Function

Write a function 'greet(name)' that takes a name as input and returns a greeting string.
Call the function with your name and print the result.

Task 5: Conditional Statements

Write a function 'is_even(num)' that returns True if the number is even and False if the number is odd.
Call the function with values 4 and 7, and print the results.

Task 6: Lists and Dictionaries

Create a list fruits = ["apple", "banana", "orange"].
Create a dictionary fruit_colors = {"apple": "red", "banana": "yellow", "orange": "orange"}.
Loop through the fruits list and print the fruit and its corresponding color using the fruit_colors dictionary.

Task 7: List Comprehension

Create a list of squares of numbers from 1 to 10 using list comprehension.
Print the resulting list.

Task 8: For Loop with Conditional

Write a for loop to print all numbers between 1 and 20 that are divisible by 3.

Task 9: Function with Loop and Conditionals

Write a function fizz_buzz(n) that prints numbers from 1 to n.

For multiples of 3, print "Fizz" instead of the number.
For multiples of 5, print "Buzz" instead of the number.
For multiples of both 3 and 5, print "FizzBuzz" instead of the number.
Call the function with n = 20.

Task 10: Error Handling - Division

Write a function that takes two numbers and returns their division.
Use try-except to handle division by zero.
Test the function with various inputs, including a division by zero scenario.

Task 11: Error Handling - List Index

Write a function that takes a list and an index, and returns the value at that index.
Use try-except to handle index out of range errors.
Test the function with a valid index and an out-of-range index.

Task 12: User Input and Error Handling

Write a program that reads input from the user, converts it to an integer, and prints the square of the number.
Use try-except to handle invalid input (non-integer input).
Keep asking for input until a valid integer is provided.

Task 13: File Writing

Write a Python program to create a text file 'hello.txt' and write the text "Hello, World!" into it.
Print a success message after writing to the file.

Task 14: File Reading

Write a Python program to read the contents of 'hello.txt' and print them to the console.
Handle the case where the file might not exist.

Task 15: File Processing

Create a text file 'numbers.txt' with a list of numbers, one per line.
Write a Python program that reads 'numbers.txt', calculates the sum and average of the numbers, and prints the results.
Handle potential errors, such as the file not existing or containing non-numeric data.