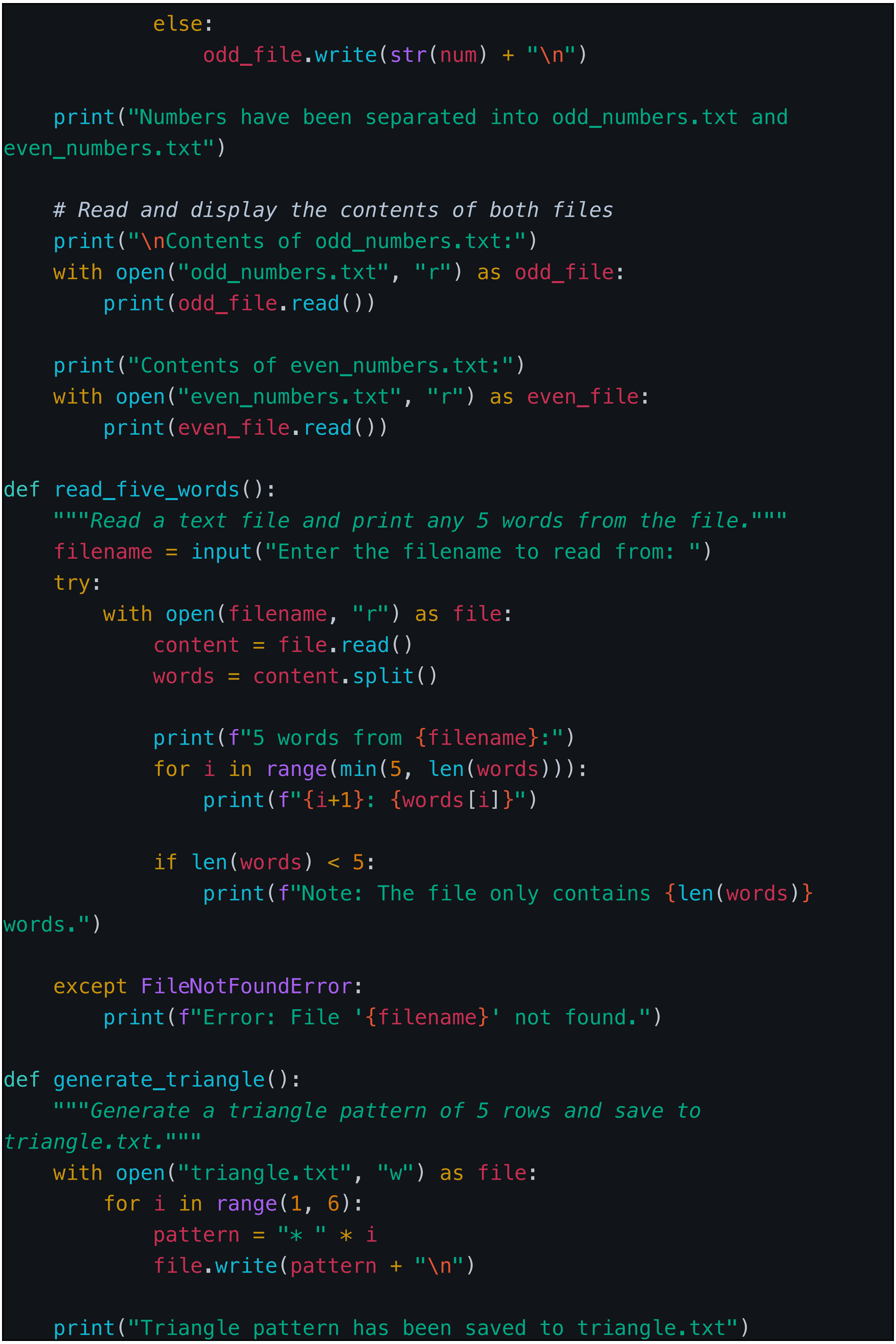
Assignment 8

8

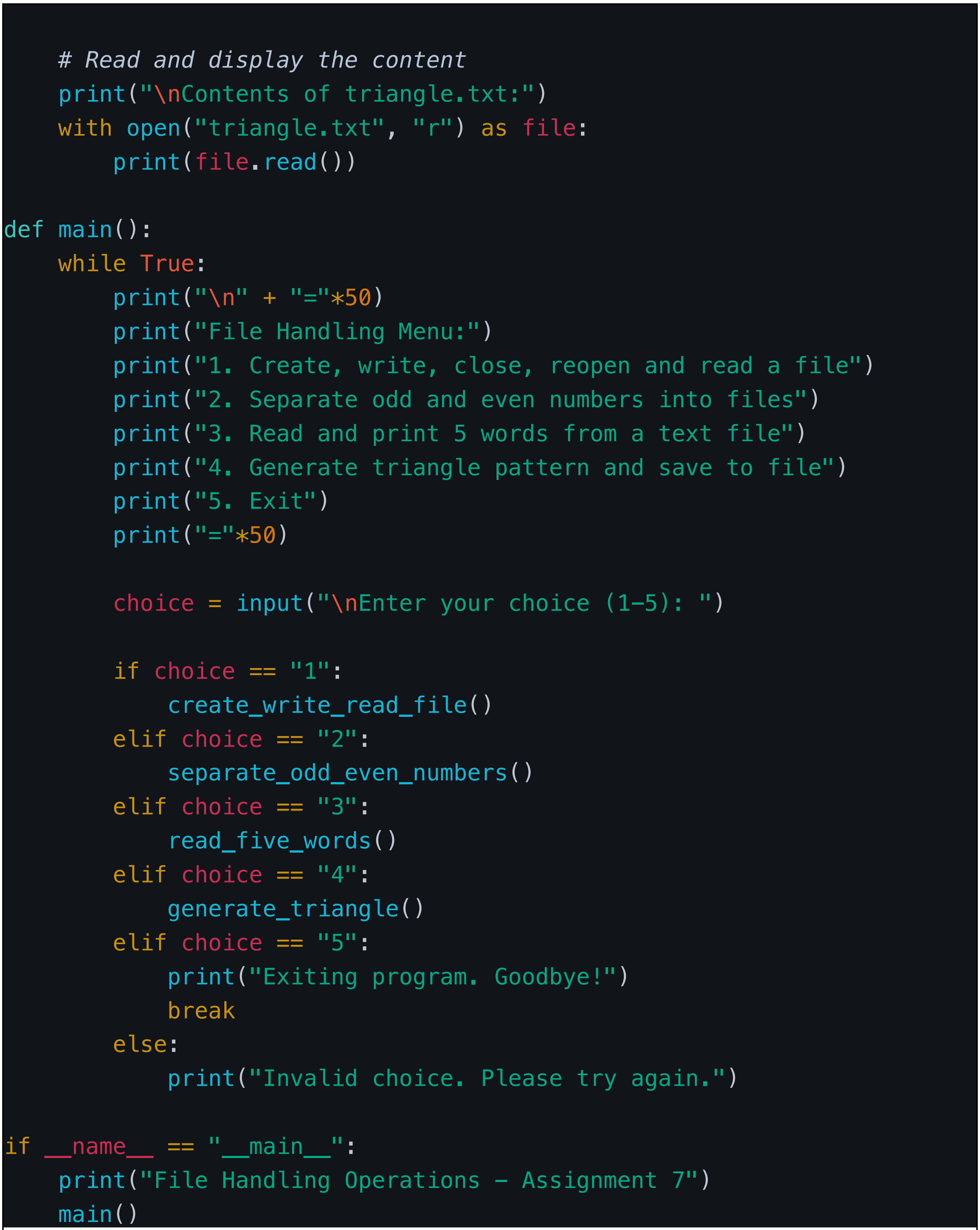
Aim:
Write a program to define a custom exception and raise it in specific scenarios, then handle it using a try-except block.
Write a program to access an element in a list and handle an IndexError if the index is out of range. Display a user-friendly error message and prompt the user to enter a valid index.
Write a program to convert a string to an integer and handle a ValueError if the string is not a valid number. Display a user-friendly error message and prompt the user to enter a valid number

Code:

```
def create_write_read_file():
    """Create a new file, write content to it, close it, and then
reopen to read and display content."""
    # Create and write to file
    with open("sample.txt", "w") as file:
        file.write("Hello, this is some sample content.\n")
        file.write("Python file handling is fun and useful!\n")
        file.write("This file was created as part of Assignment 7.")

    print("File created and content written successfully.")

    # Reopen and read the file
    try:
        with open("sample.txt", "r") as file:
            content = file.read()
            print("\nContent of the file:")
            print(content)
    except FileNotFoundError:
        print("Error: File not found.")

def separate_odd_even_numbers():
    """Read a list of numbers and insert odd numbers into
odd_numbers.txt and even numbers into even_numbers.txt."""
    numbers = input("Enter numbers separated by spaces: ").split()

    # Convert inputs to integers
    numbers = [int(num) for num in numbers]

    # Open files for writing
    with open("odd_numbers.txt", "w") as odd_file,
open("even_numbers.txt", "w") as even_file:
        for num in numbers:
            if num % 2 == 0:
                even_file.write(str(num) + "\n")
```

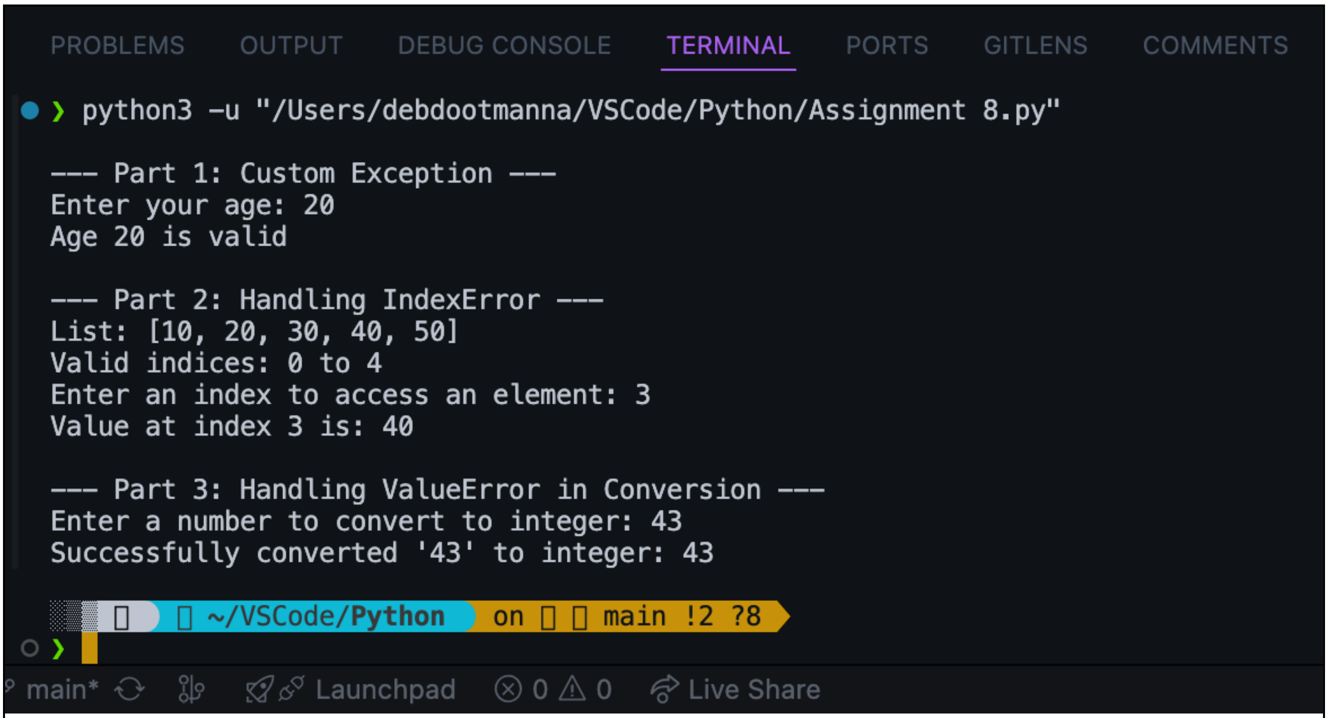


```
            else:
                odd_file.write(str(num) + "\n")

    print("Numbers have been separated into odd_numbers.txt and 
even_numbers.txt")

    # Read and display the contents of both files
    print("\nContents of odd_numbers.txt:")
    with open("odd_numbers.txt", "r") as odd_file:
        print(odd_file.read())

    print("Contents of even_numbers.txt:")
    with open("even_numbers.txt", "r") as even_file:
        print(even_file.read())

def read_five_words():
    """Read a text file and print any 5 words from the file."""
    filename = input("Enter the filename to read from: ")
    try:
        with open(filename, "r") as file:
            content = file.read()
            words = content.split()

            print(f"5 words from {filename}:")
            for i in range(min(5, len(words))):
                print(f"{i+1}: {words[i]}")

            if len(words) < 5:
                print(f"Note: The file only contains {len(words)} 
words.")

    except FileNotFoundError:
        print(f"Error: File '{filename}' not found.")

def generate_triangle():
    """Generate a triangle pattern of 5 rows and save to 
triangle.txt."""
    with open("triangle.txt", "w") as file:
        for i in range(1, 6):
            pattern = "* " * i
            file.write(pattern + "\n")

    print("Triangle pattern has been saved to triangle.txt")
```



```
    # Read and display the content
    print("\nContents of triangle.txt:")
    with open("triangle.txt", "r") as file:
        print(file.read())

def main():
    while True:
        print("\n" + "="*50)
        print("File Handling Menu:")
        print("1. Create, write, close, reopen and read a file")
        print("2. Separate odd and even numbers into files")
        print("3. Read and print 5 words from a text file")
        print("4. Generate triangle pattern and save to file")
        print("5. Exit")
        print("="*50)

        choice = input("\nEnter your choice (1-5): ")

        if choice == "1":
            create_write_read_file()
        elif choice == "2":
            separate_odd_even_numbers()
        elif choice == "3":
            read_five_words()
        elif choice == "4":
            generate_triangle()
        elif choice == "5":
            print("Exiting program. Goodbye!")
            break
        else:
            print("Invalid choice. Please try again.")

if __name__ == "__main__":
    print("File Handling Operations - Assignment 7")
    main()
```

Output Screenshot:



```
❯ python3 -u "/Users/debdootmanna/VSCode/Python/Assignment 8.py"

--- Part 1: Custom Exception ---
Enter your age: 20
Age 20 is valid

--- Part 2: Handling IndexError ---
List: [10, 20, 30, 40, 50]
Valid indices: 0 to 4
Enter an index to access an element: 3
Value at index 3 is: 40

--- Part 3: Handling ValueError in Conversion ---
Enter a number to convert to integer: 43
Successfully converted '43' to integer: 43

   ~/VSCode/Python  on   main !2 ?8
❯
```

Conclusion/Summary:

In this assignment, I demonstrated three important exception handling concepts:

1. Creating and using custom exceptions:
   - Defined InvalidAgeError to validate age inputs
   - Used try-except blocks to handle both custom and built-in exceptions

2. Handling IndexError:
   - Protected the program from crashing when accessing invalid list indices
   - Provided user-friendly error messages and reprompted for valid input

3. Handling ValueError during type conversion:
   - Safely converted string input to integers
   - Implemented proper error handling with descriptive messages

Exception handling is a critical aspect of writing robust programs that can gracefully handle unexpected situations without crashing. By anticipating potential errors and providing meaningful feedback, we create a better user experience while maintaining program stability.

Student Signature & Date	**Marks:**	**Evaluator Signature & Date**