

Assignment 1

Report

Norzin Wangmo (1 SWE)

02250359

# 1. Introduction:

* This report covers the setup for six utility functions in part A.
* It also includes two games for part B.
* You get the complete Python code for everything.
* There are short notes on the logic behind each one.
* Input checks are explained, too.
* And expected results.
* Screenshots are also pasted.

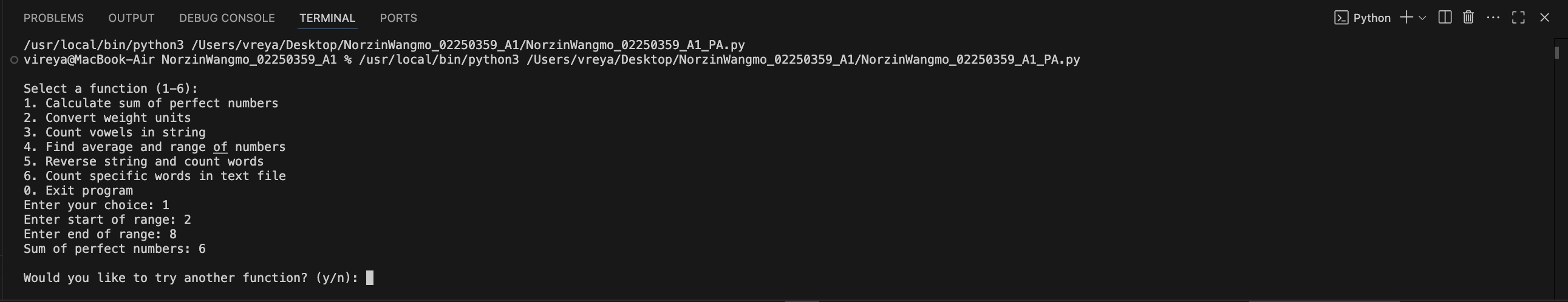
# 2. Part A: Python Functions:

* The program runs from the command line.
* It shows a menu so you can pick from the six functions.
* It keeps going until you choose to quit.

## 2.1 Perfect Number Sum Calculator:

* The perfect number sum calculator comes first.
* It adds up all perfect numbers in a range you set.
* The range includes both ends.
* It makes sure the inputs are integers before doing anything.

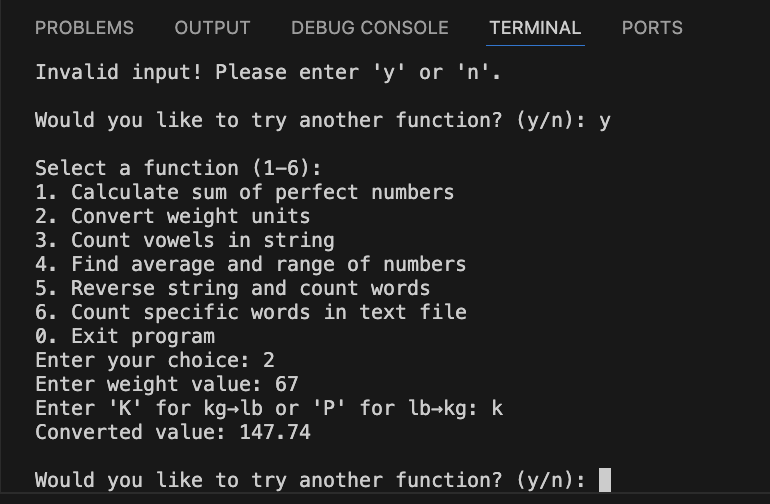
**Output:**



## 2.2 Weight Unit Converter:

* Next is the weight unit converter.
* It switches between kilograms and pounds.
* You enter K to go from kg to lb.
* Or P for the other way.
* If you pick wrong, it gives an error message.

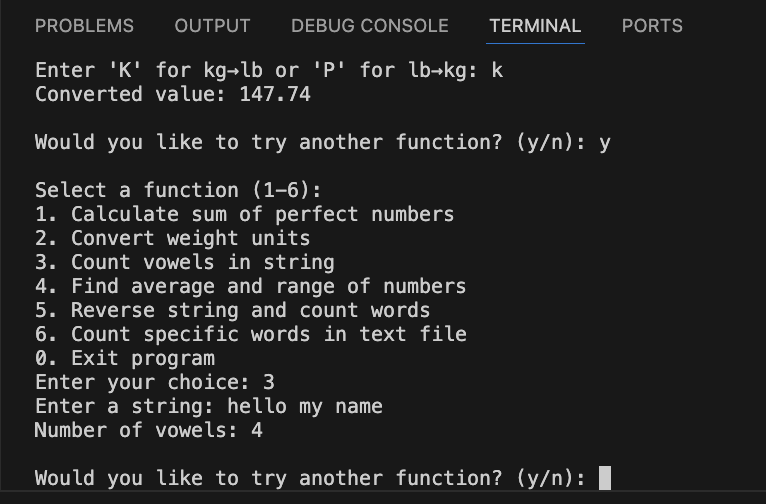
**Output:**



## 2.3 Vowel Counter:

* The vowel counter tallies up "a e i o u” in any string you give it.
* It does not care about upper or lower case.

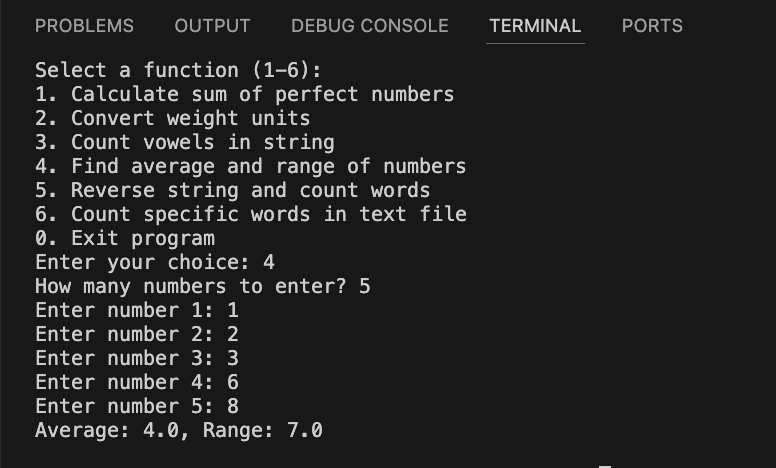
**Output:**



## 2.4 Average and Range Finder:

* It figures the average for a list of numbers.
* Also, the range is max minus min.
* It throws an error if the list is empty or contains no numbers.

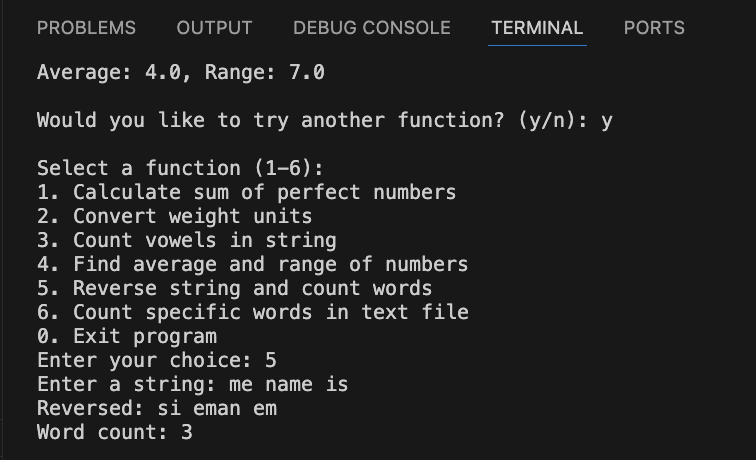
**Output:**



## 2.5 String Reverser with Word Count

* It flips the whole string around.
* It counts words, too.
* It ignores extra spaces between them.

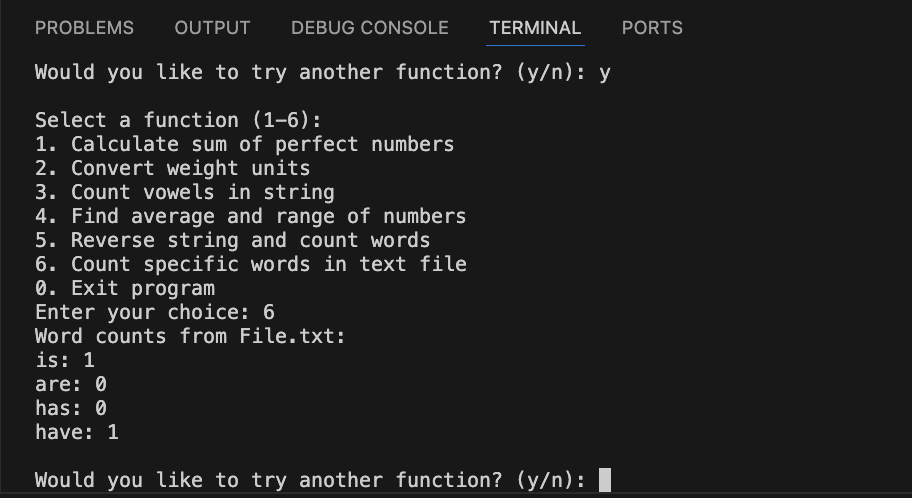
**Output:**



## 2.6 Specific Word Counter

* It looks for “is, are, has, have” in a file called File.txt.
* The file is local.
* If the file is not there, it just returns zeros for everything.

**Output:**



**Complete Code for Part A :**

# 1. Perfect Number Sum Calculator  
def perfect\_number\_sum(start: int, end: int) -> int:  
 #Return the sum of all perfect numbers within the inclusive range [start, end]  
 def is\_perfect(n: int) -> bool:  
 return n > 1 and sum(i for i in range(1, n) if n % i == 0) == n  
  
 return sum(n for n in range(start, end + 1) if is\_perfect(n))  
  
  
# 2. Weight Unit Converter  
def weight\_converter(value: float, direction: str) -> float:  
 #Convert between kilograms and pounds. Use 'K' for kg→lb and 'P' for lb→kg.  
 if direction.upper() == 'K':  
 return round(value \* 2.205, 2)  
 elif direction.upper() == 'P':  
 return round(value / 2.205, 2)  
 else:  
 raise ValueError("Invalid direction. Use 'K' for kilograms to pounds or 'P' for pounds to kilograms.")  
  
  
# 3. Vowel Counter  
def count\_vowels(text: str) -> int:  
 #Return the number of vowels in the string (case insensitive).  
 vowels = "aeiou"  
 return sum(1 for ch in text.lower() if ch in vowels)  
  
  
# 4. Average and Range Finder  
def average\_and\_range(numbers: list[float]) -> tuple[float, float]:  
 #Return the average and range (max – min) of the numbers.  
 if not numbers:  
 raise ValueError("The list of numbers cannot be empty.")  
 avg = sum(numbers) / len(numbers)  
 rng = max(numbers) - min(numbers)  
 return round(avg, 2), round(rng, 2)  
  
  
# 5. String Reverser with Word Count  
def reverse\_and\_count\_words(text: str) -> tuple[str, int]:  
 #Return the reversed string and the word count (handling multiple spaces).  
 reversed\_str = text[::-1]  
 words = [w for w in text.split(" ") if w.strip() != ""]  
 return reversed\_str, len(words)  
  
  
# 6. Specific Word Counter  
def specific\_word\_counter():  
 # Count the occurrences of specific words ['is', 'are', 'has', 'have'] in File.txt  
 try:  
 # Open and read the local File.txt  
 file = open("File.txt", "r")  
 text = file.read()  
 file.close()  
 except:  
 print("Error: Could not find or read File.txt file.")  
 return {"is": 0, "are": 0, "has": 0, "have": 0}  
  
 # Convert text to lowercase for counting  
 text = text.lower()  
   
 # Split text into individual words  
 words = text.split()  
   
 # Remove punctuation from each word  
 clean\_words = []  
 for word in words:  
 clean\_word = ""  
 for char in word:  
 if char.isalpha(): # Keep only letters  
 clean\_word = clean\_word + char  
 if clean\_word: # Add word if it's not empty  
 clean\_words.append(clean\_word)  
   
 # Count target words  
 target\_words = ["is", "are", "has", "have"]  
 counts = {"is": 0, "are": 0, "has": 0, "have": 0}  
   
 for word in clean\_words:  
 if word in target\_words:  
 counts[word] = counts[word] + 1  
   
 return counts  
  
  
# Main Program  
def main():  
 while True:  
 print("\nSelect a function (1-6):")  
 print("1. Calculate sum of perfect numbers")  
 print("2. Convert weight units")  
 print("3. Count vowels in string")  
 print("4. Find average and range of numbers")  
 print("5. Reverse string and count words")  
 print("6. Count specific words in text file")  
 print("0. Exit program")  
  
 # Loop until valid choice is entered  
 while True:  
 choice = input("Enter your choice: ").strip()  
 if choice in {'0','1','2','3','4','5','6'}:  
 break  
 print("Invalid choice!. Please enter a number between 0 and 6.")  
  
 if choice == '0':  
 print("Goodbye!")  
 break  
  
 try:  
 if choice == '1':  
 start = int(input("Enter start of range: "))  
 end = int(input("Enter end of range: "))  
 print(f"Sum of perfect numbers: {perfect\_number\_sum(start, end)}")  
  
 elif choice == '2':  
 val = float(input("Enter weight value: "))  
 direction = input("Enter 'K' for kg→lb or 'P' for lb→kg: ").strip()  
 print(f"Converted value: {weight\_converter(val, direction)}")  
  
 elif choice == '3':  
 text = input("Enter a string: ")  
 print(f"Number of vowels: {count\_vowels(text)}")  
  
 elif choice == '4':  
 n = int(input("How many numbers to enter? "))  
 nums = []  
 for i in range(n):  
 nums.append(float(input(f"Enter number {i+1}: ")))  
 avg, rng = average\_and\_range(nums)  
 print(f"Average: {avg}, Range: {rng}")  
  
 elif choice == '5':  
 text = input("Enter a string: ")  
 rev, wc = reverse\_and\_count\_words(text)  
 print(f"Reversed: {rev}\nWord count: {wc}")  
  
 elif choice == '6':  
 counts = specific\_word\_counter()  
 print("Word counts from File.txt:")  
 for word, count in counts.items():  
 print(f"{word}: {count}")  
  
 except Exception as e:  
 print(f" ! Error: {e}")  
  
 while True:  
 again = input("\nWould you like to try another function? (y/n): ").strip().lower()  
 if again == 'y':  
 break  
 elif again == 'n':  
 print("Goodbye! :) ")  
 return  
 else:  
 print("Invalid input! Please enter 'y' or 'n'.")  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

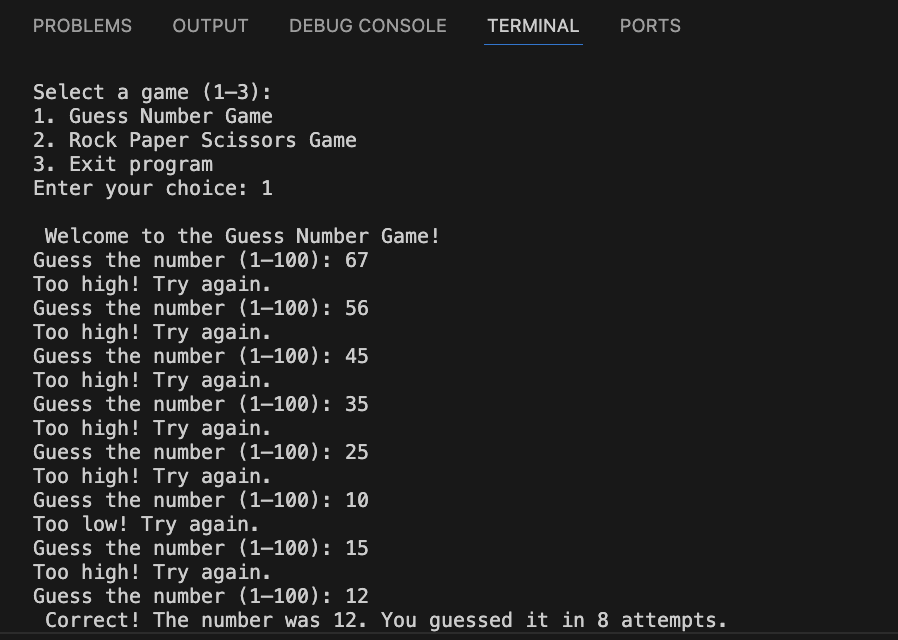
# 3. Part B: Games

* It has two games.
* Both use text inputs.
* They check what you type.
* It provides clear feedback.

## 3.1 Guess the Number:

* A random number is generated from 1 to 100.
* You need to guess the number.
* It has to be a number, or it shows an error.
* Hints are provided to tell you whether it is higher or lower.
* It counts your tries.
* It stops when you get the number right.

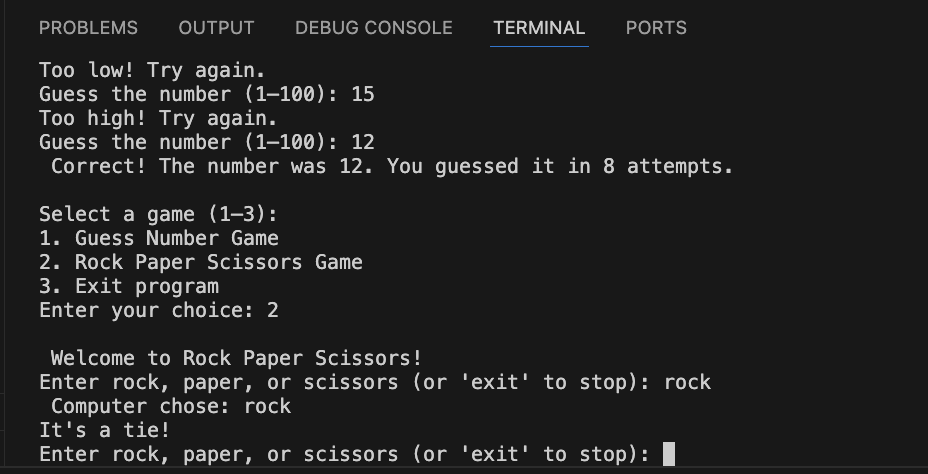
**Output:**

****

## 3.2 Rock Paper Scissors Game

* You play against the computer.
* The computer picks its stand randomly.
* The user enters rock, paper, or scissors.
* It checks if it is a valid input or not.
* Keeps score for wins and losses.
* When you exit, it sums up the final scores and gives you the output.

**Output:**

****

**Complete Code for Part B:**

import random  
  
# 1. Guess Number Game  
def guess\_number\_game():  
 #Play a number guessing game between 1 and 100.  
 print("\n Welcome to the Guess Number Game!")  
 number = random.randint(1, 100)  
 attempts = 0  
  
 while True:  
 try:  
 guess = int(input("Guess the number (1–100): "))  
 if guess < 1 or guess > 100:  
 print("exit Please enter a number between 1 and 100.")  
 continue  
  
 attempts += 1  
 if guess < number:  
 print("Too low! Try again.")  
 elif guess > number:  
 print("Too high! Try again.")  
 else:  
 print(f" Correct! The number was {number}. You guessed it in {attempts} attempts.")  
 break  
 except ValueError:  
 print("! Invalid input. Please enter a valid integer.")  
  
  
# 2. Rock Paper Scissors Game  
def rock\_paper\_scissors():  
 """Play a rock-paper-scissors game against the computer."""  
 print("\n Welcome to Rock Paper Scissors!")  
 choices = ["rock", "paper", "scissors"]  
 user\_wins = 0  
 computer\_wins = 0  
 rounds = 0  
  
 while True:  
 user\_choice = input("Enter rock, paper, or scissors (or 'exit' to stop): ").lower().strip()  
  
 if user\_choice == 'exit':  
 print("\n Final Results:")  
 print(f"You won {user\_wins} times.")  
 print(f"Computer won {computer\_wins} times.")  
 print(f"Total rounds played: {rounds}\n")  
 break  
  
 if user\_choice not in choices:  
 print("Invalid choice. Please enter rock, paper, or scissors.")  
 continue  
  
 computer\_choice = random.choice(choices)  
 print(f" Computer chose: {computer\_choice}")  
 rounds += 1  
  
 if user\_choice == computer\_choice:  
 print("It's a tie!")  
 elif (  
 (user\_choice == "rock" and computer\_choice == "scissors") or  
 (user\_choice == "scissors" and computer\_choice == "paper") or  
 (user\_choice == "paper" and computer\_choice == "rock")  
 ):  
 print(" You win this round!")  
 user\_wins += 1  
 else:  
 print(" Computer wins this round!")  
 computer\_wins += 1  
  
  
# Main Menu  
def main():  
 while True:  
 print("\nSelect a game (1–3):")  
 print("1. Guess Number Game")  
 print("2. Rock Paper Scissors Game")  
 print("3. Exit program")  
  
 choice = input("Enter your choice: ").strip()  
  
 if choice == '1':  
 guess\_number\_game()  
 elif choice == '2':  
 rock\_paper\_scissors()  
 elif choice == '3':  
 print("Goodbye! ")  
 break  
 else:  
 print(" Invalid choice. Please enter 1, 2, or 3.")  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

# 4. Summary:

* All the menu options get checked.
* It rejects bad numbers or weird strings.
* Conversions for numbers round off properly.
* Errors for empty lists or no File.txt are handled nicely.
* Prompts stay clear.

# 5. Conclusion:

The whole thing wraps up with solid input handling. The output format is clean. Code stays modular for functions and games. The menu makes testing easy. You can demo it without hassle.

# 6. GitHub Repository URL:

[Github Repository](https://github.com/norzin-wangmo/NorzinWangmo_02250359_A1.git)