Assignment 2

By: Alexander
UMSL

Coding Journey Summary:

- ⏱️ 1 hour 27 seconds of active typing
- 📅 3 hours 16 minutes 25 seconds total duration
- 🔢 3,496 characters | 107 lines
- 📈 Typing speed: 1 character(s) per second
- 🗄️ Process Data: Showing 188 changes made every 5+ seconds (of 379 changes total)
- 📊 Interactive version of this report is AVAILABLE HERE.

Final Code

If the code contains more than one file, they are concatenated and shown.

```
==+==
FILE: Main.py
==+==

# imports
from decimal import Decimal
import math
from itertools import permutations
from statistics import mean, median, mode, pstdev, pvariance

# some constants
Bday = 22
d = 2 + (Bday % 2)
m = 1 + (Bday // 2)
PI = 3.14159
volume = 0
circumference = 0
area = 0

# begin problem 1
user_input = int(input("Enter a value\n"))
if user_input % m == 0:
    volume = (4/3) * PI * m ** 3
    circumference = 2 * PI * m ** 2
    area = PI * m ** 2
else:
    volume = (4/3) * PI * user_input ** 3
    circumference = 2 * PI * user_input ** 2
    area = PI * user_input ** 2

# begin problem 2
some_number = 4554
```

```
some_number_as_string = str(some_number)
if some_number_as_string == some_number_as_string[::-1]:
    print("Yay its a palindrome")


# begin problem 3 factorial.
fact_user_input = int(input("Enter a non-negative integer\n"))
factorial = 1
for x in range(fact_user_input, 0, -1):
    factorial *= x
print(f'Factorial is {factorial}')


# begin problem 4
principal = Decimal("2000.00")
rate = Decimal("0.07")
for year in range(10, 31, 10):
    amount = principal * (1 + rate) ** year
    print(f'Amount in {year} years {amount:>10.2f}')


# begin problem 5
def square_cube(x):
    """Return square and cube of input x if x is int else return message invalid input
    x -- int
    """
    if x.isdigit():
        # convert to int if it is a digit
        x = int(x)
        return print(f'square value is {math.pow(x, 2)}, cubed value is {math.pow(x, 3)}')
    else:
        print("Wrong type dude.")

square_input = input("Enter a number to be squared and cubed\n")
square_cube(square_input)

# begin problem 6
def anagram(string):
    """Returns all possible anagrams of a given string

    string -- str
    return - tuple of characters
    """
    return permutations(string, len(string))

# anagram returns a tuple so iterate and join into a string
for word in anagram("he"):
    print(''.join(word))

# begin problem 7
fake_stack = []
input_prompt = "Enter a value to add to the stack. Enter -999 to end\n"
stack_input = int(input(input_prompt))
# if initial value is not kill value then append to stack
if stack_input != -999:
    fake_stack.append(stack_input)
while stack_input != -999:
    stack_input = int(input(input_prompt))
```

```
    stack_input = int(input(input_prompt))
    # check here to catch -999 and not enter it to stack
    if stack_input != -999:
        fake_stack.append(stack_input)

# some functions to do some func-y stuff..get it?
def compute_central_tendancies():
    """Compute central tendancy measures mean, median, mode"""
    print(f'Mean {mean(fake_stack)}, median {median(fake_stack)}, mode {mode(fake_stack)}')

def compute_dispersion():
    """Compute dispersion measures - variance and standard deviation"""
    print(f'Variance is {pvariance(fake_stack)}, standard deviation is {pstdev(fake_stack)}')

# invoke measures once user ends input
compute_central_tendancies()
compute_dispersion()

# begin problem 8
def is_perfect_square(num):
    if num == math.isqrt(num) ** 2:
        print(f'{num} is a perfect square!')
    else:
        print(f'{num} is not a perfect square.')

def ask_user():
    times_asked = 0
    user_prompt = "Enter a number to determine if its perfect square\n"
    while times_asked < 5:
        user_number = int(input(user_prompt))
        is_perfect_square(user_number)
        times_asked += 1

ask_user()


# begin problem 9
def is_palindrome(list_of_chars):
    """Return true if list of characters is a palindrome"""
    return "".join(list_of_chars) == "".join(list(reversed(list_of_chars)))

chars = ["h", "e", "h"]
chars_2 = ["h", "e", "l"]

print(is_palindrome(chars))
print(is_palindrome(chars_2))
```

Final Execution Output

Output of the most recent code execution.

```
Enter a value
Yay its a palindrome
Enter a non-negative integer
Factorial is 120
Amount in 10 years    3934.30
Amount in 20 years    7739.37
Amount in 30 years   15224.51
Enter a number to be squared and cubed
square value is 100.0, cubed value is 1000.0
he
eh
Enter a value to add to the stack. Enter -999 to end
Enter a value to add to the stack. Enter -999 to end
Enter a value to add to the stack. Enter -999 to end
Mean 25, median 25.0, mode 20
Variance is 25, standard deviation is 5.0
Enter a number to determine if its perfect square
5 is not a perfect square.
Enter a number to determine if its perfect square
6 is not a perfect square.
Enter a number to determine if its perfect square
7 is not a perfect square.
Enter a number to determine if its perfect square
25 is a perfect square!
Enter a number to determine if its perfect square
30 is not a perfect square.
True
False
```

Inputs to the Code

Most recent user-typed inputs to the program (STDIN).

```
12
5
10
20
30
-999
5
6
7
25
30
```

Most Common Error

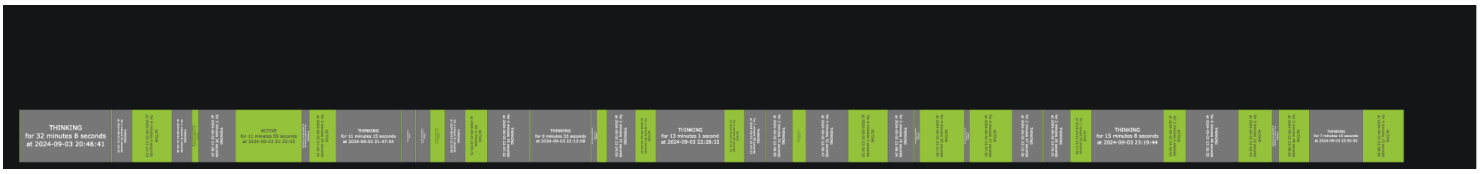
The most common error that occured during the coding process; "null" (if any) indicates no errors.

```
STDERR:
Traceback (most recent call last):
  File "Main.py", line 32, in <module>
    factorial *= x
NameError: name 'factorial' is not defined


Enter a value
Yay its a palindrome
Enter a non-negative integer
```

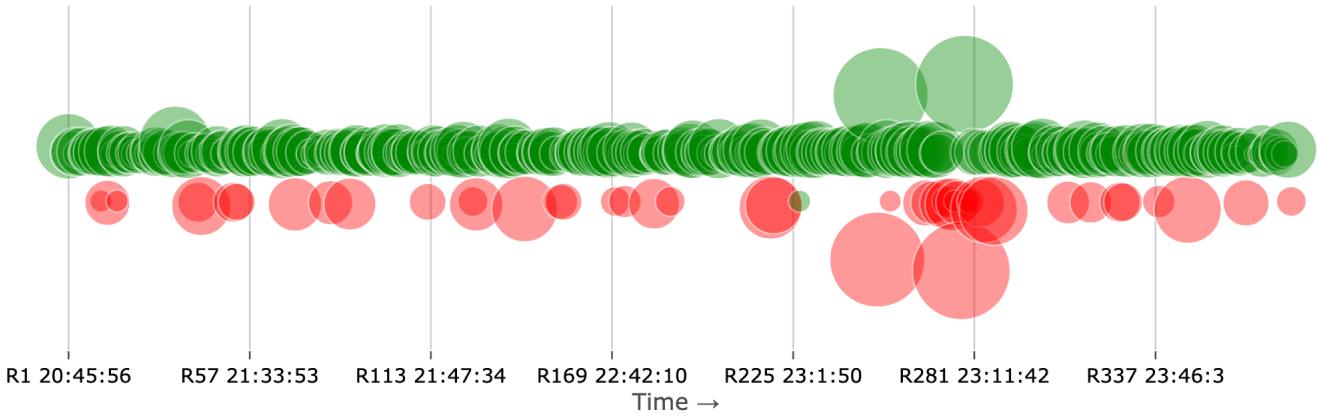
Active Sessions

Active typing sessions while working on the task, shown as green bars.



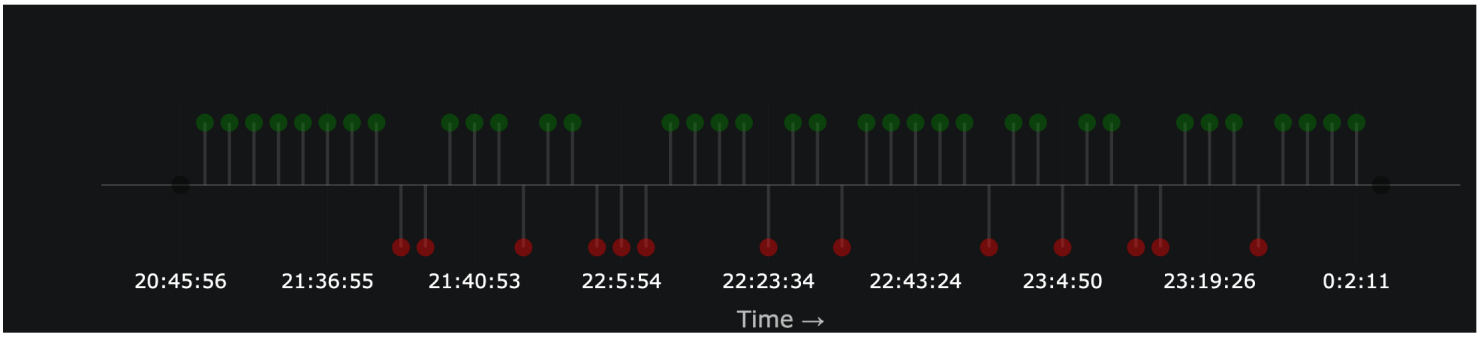
Text Added or Removed Over Time

Green and red circles show text add and removed; size of the circle shows change in the number of characters.



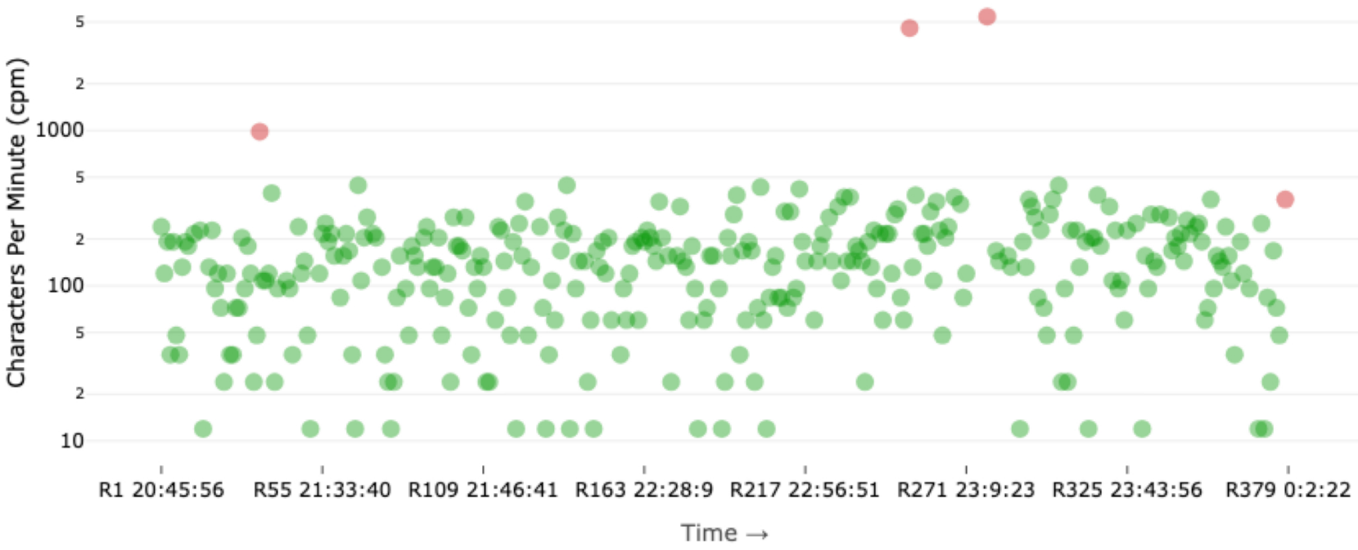
Code Executions

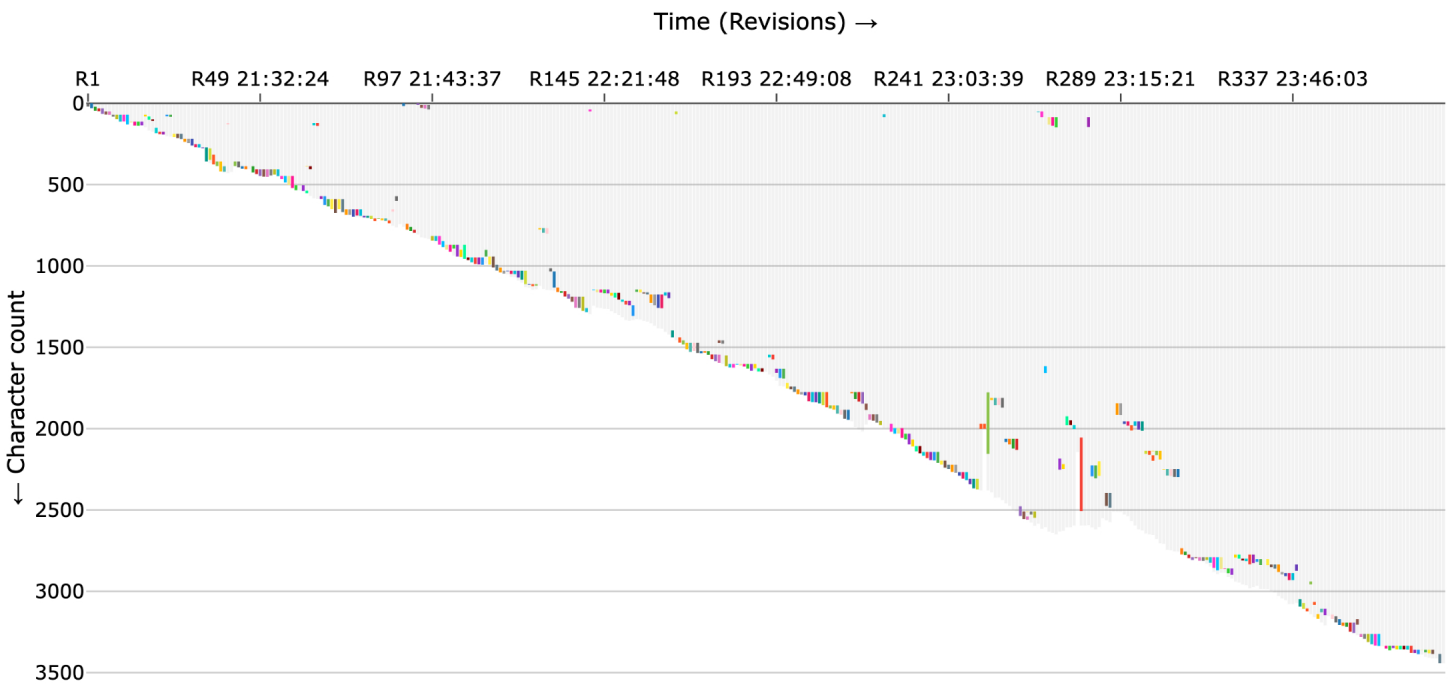
Green and red dots correspond to successful and failed executions.



Typing Speed

Typing speed can be negative when text is deleted.



Edited Line Locations

Each edited line block is assigned a color; highlighted bars show the line(s) edited at that revision.