**Handling and Manipulating Strings**

Kenneth Thomas Barrett

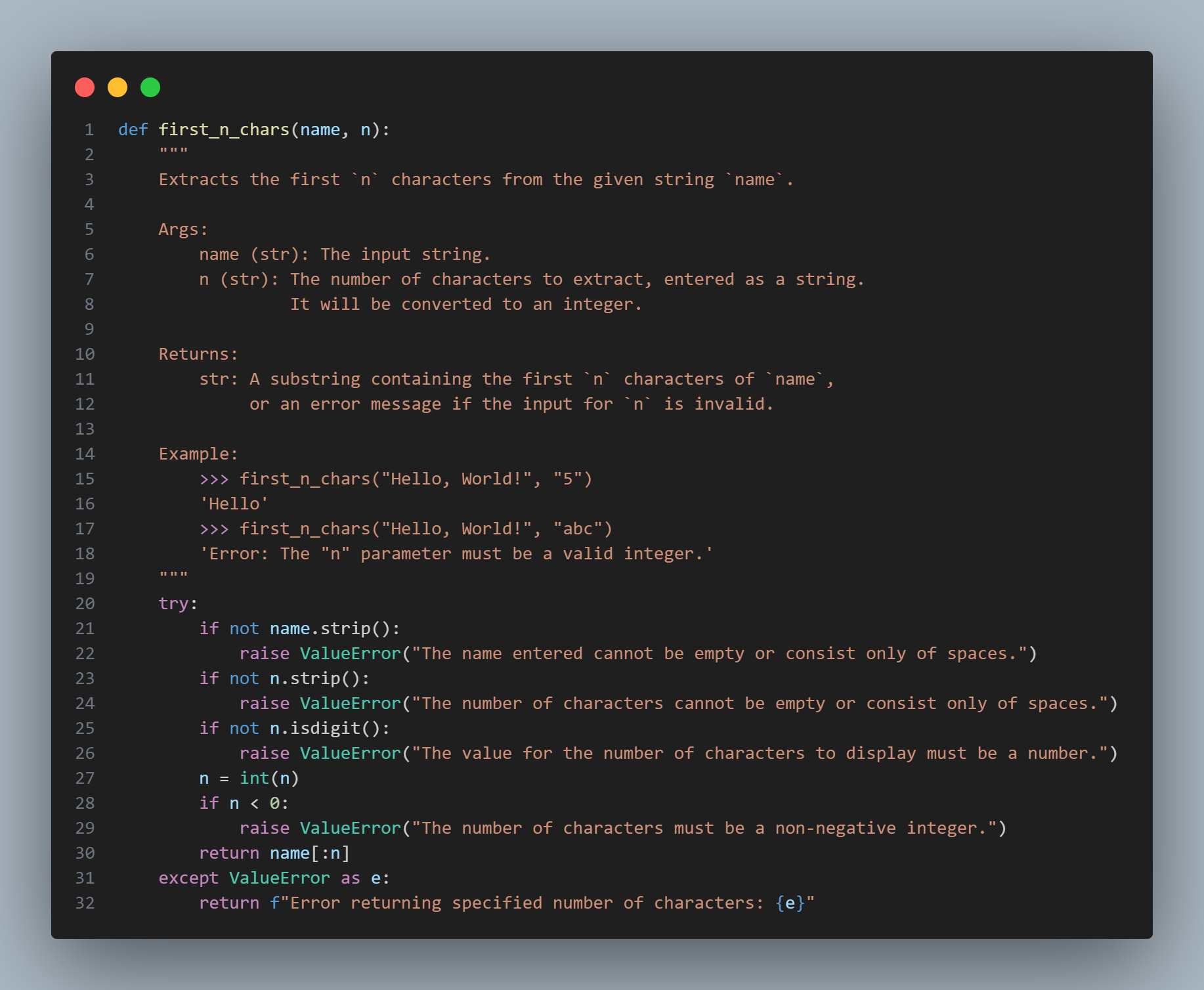
Computer Science Department, University of the People

CS1101 - Programming Fundamentals

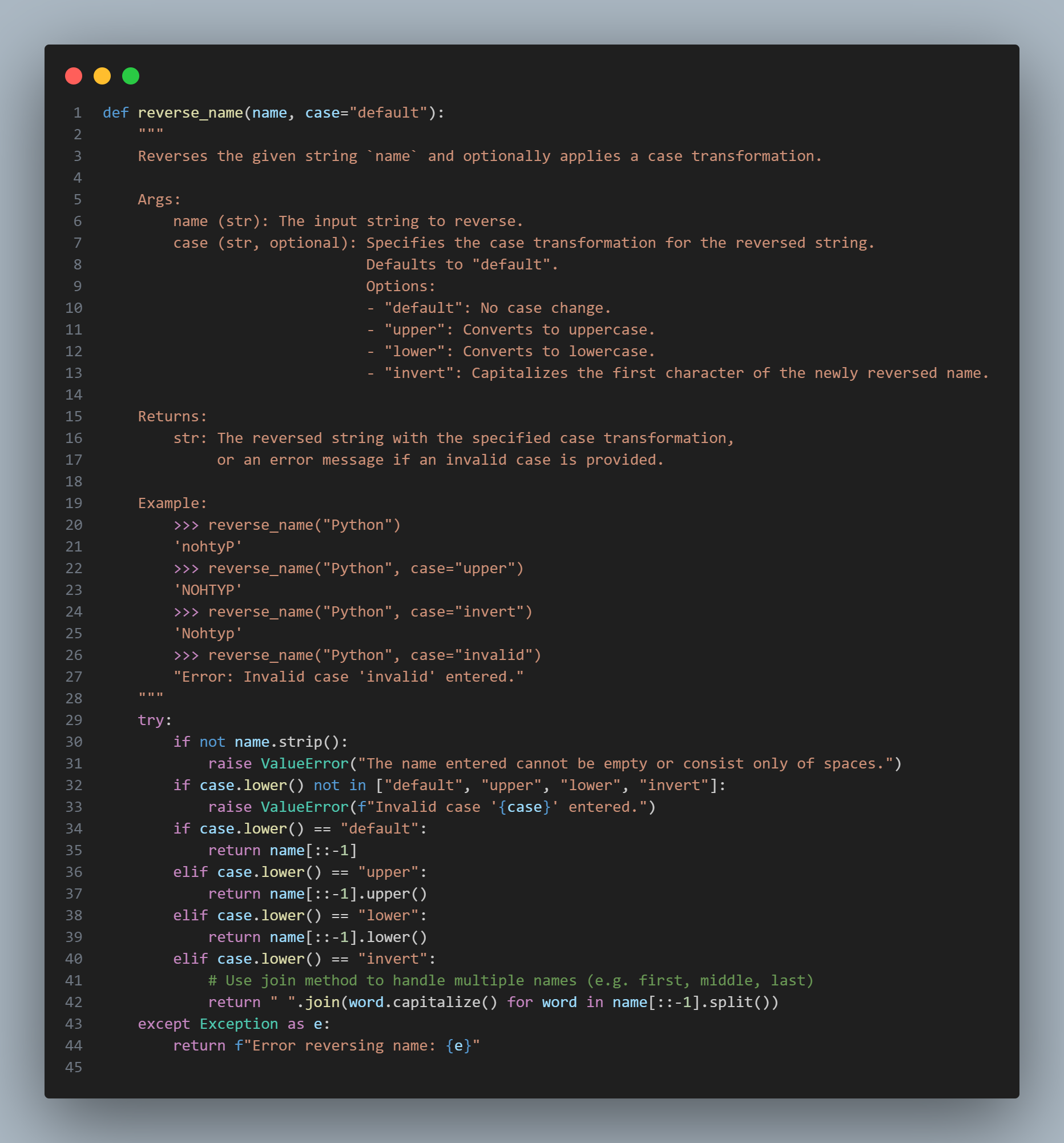
Ola Fatai Ayanleke Ayanleke

December 16, 2024

### Return ‘n’ Characters from String

In my function to return the first ‘n’ characters from the name input by the user, I began by creating a function called first\_n\_chars with two parameters - ‘name’ and ‘n’. I made a try-catch block and raised multiple ValueErrors in the event the user enters a value for either parameter that is empty or consists of only spaces. Another ValueError was raised if the user assigned a non-numeric value to ‘n’. Next, I converted the value assigned to ‘n’ from a string to an integer, raising a ValueError if the value entered is not a positive, whole number. The function utilizes string slicing to return the ‘n’ number of characters of the provided name, or the appropriate error message if one occurs.

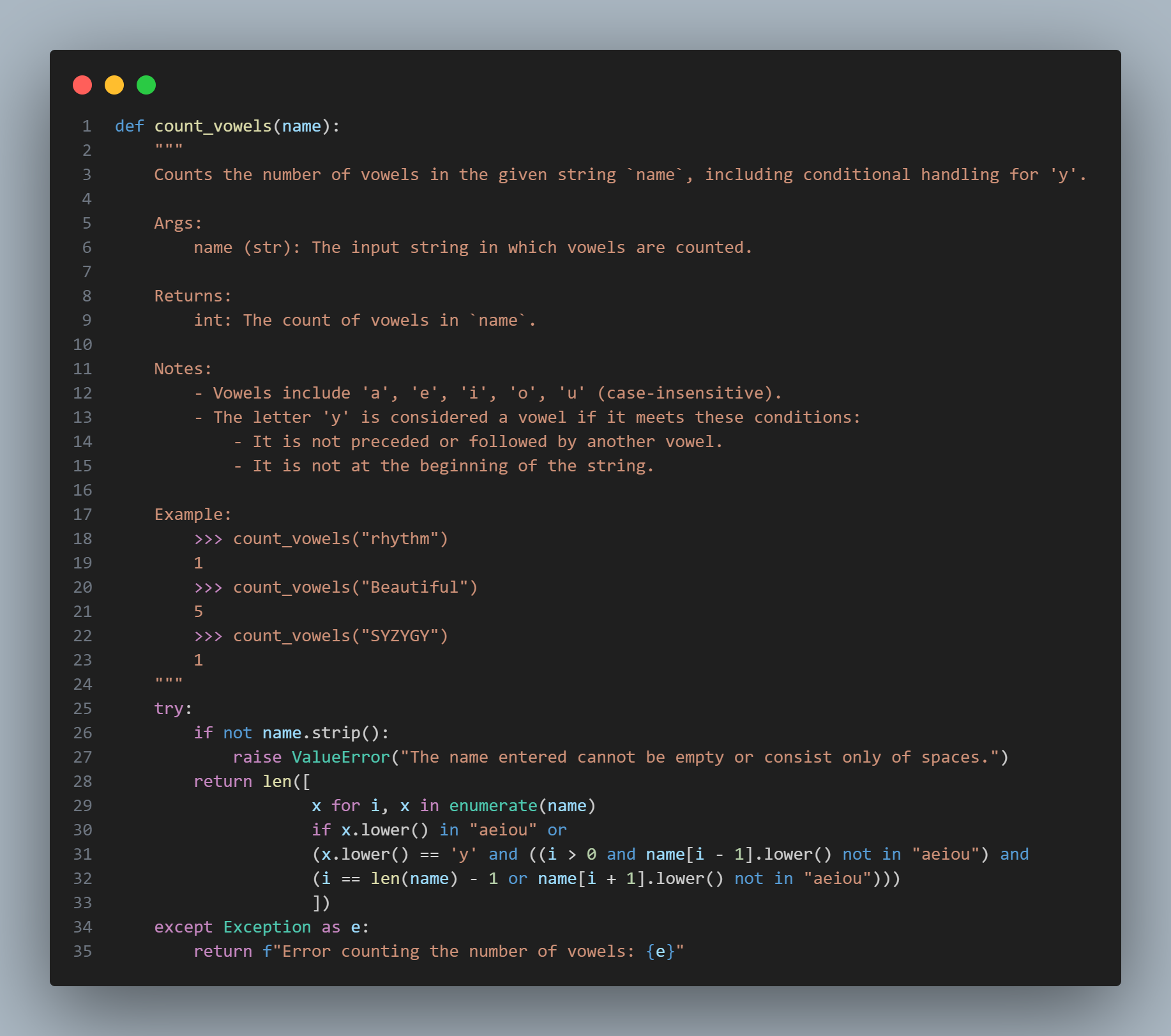
### Reverse Name

My reverse\_name function accepts two parameters: ‘name’ and an optional parameter, ‘case’. I once again began by creating a try-catch block. I then used the strip() method, which removes any leading and trailing spaces from a string. This allowed me to ensure that I could raise a ValueError if the name entered was blank, either being only spaces or having no value at all. Another ValueError was raised if ‘case’ was assigned a value not included in the pre-defined options available. If the user did not specify a case, ‘default’ is selected as the value, which directly reverses the string using a slice that steps backwards ([::-1]. Selecting “upper” or “lower” for ‘case’ reverses the string and converts it to uppercase or lowercase, respectively. Using “invert” splits the reversed string into a list of words using .split(), capitalizes the first letter of each reversed word using .capitalize(), and rejoins them into a single string (for example, “John Smith” will become “Htims Nhoj”. The function returns the processed string or an error message if one occurs.

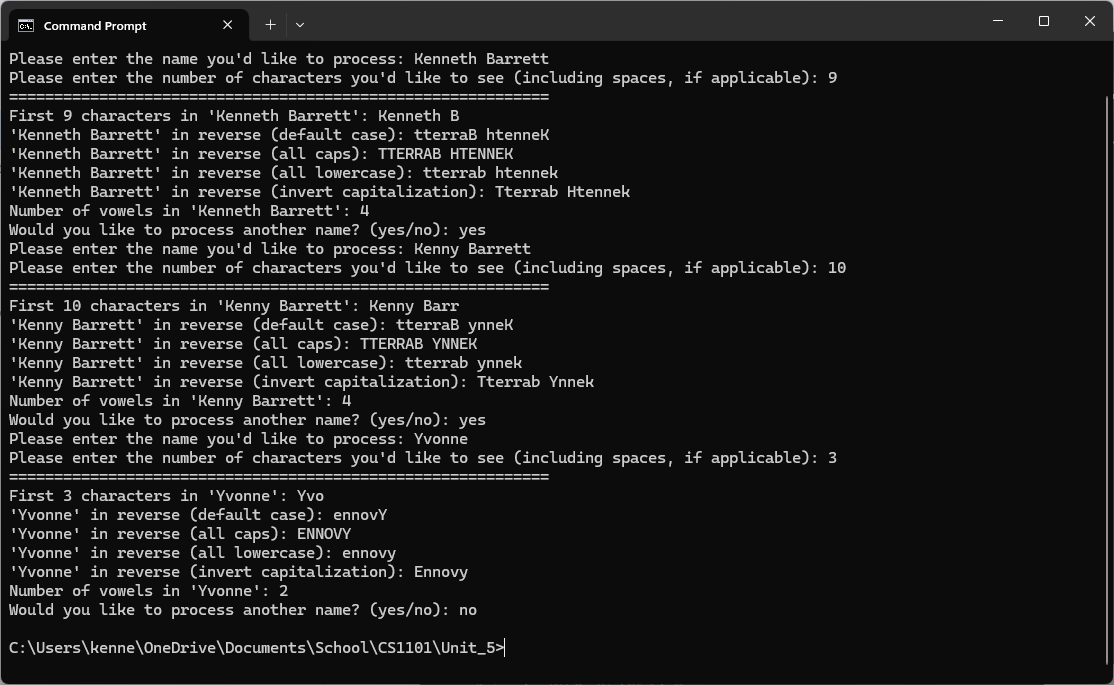
### Count Vowels

My count\_vowels function counts the number of vowels in the given string ‘name’, considering both standard vowels (a, e, i, o, and u), as well as conditional handling for the letter ‘y’. The function starts by checking if the input is empty or consists only of spaces, and raises a ValueError if so. It thehn iterates through each character in the string using a list comprehension with enumerate() to track the index (i) as well as the character (x). Because the letter ‘y’ is only treated as a vowel under specific conditions, the following conditions were considered before including it in the count:

* The letter ‘y’ is not at the beginning (i > 0)
* The letter ‘y’ is not preceded by a vowel (name[i - 1].lower() not in “aeiou”)
* The letter ‘y’ is not followed by a vowel (name[i + 1].lower() not in “aeiou”)

Characters in the name provided that are standard vowels or meet the conditional rules for ‘y’ are added to the list comprehension. The function then returns the length of the list, representing the total number of vowels. The function returns this value, or an error message if one occurred.

### Output Sample



### Code for Reproduction

def first\_n\_chars(name, n):

"""

Extracts the first `n` characters from the given string `name`.

Args:

name (str): The input string.

n (str): The number of characters to extract, entered as a string.

It will be converted to an integer.

Returns:

str: A substring containing the first `n` characters of `name`,

or an error message if the input for `n` is invalid.

Example:

>>> first\_n\_chars("Hello, World!", "5")

'Hello'

>>> first\_n\_chars("Hello, World!", "abc")

'Error: The "n" parameter must be a valid integer.'

"""

try:

if not name.strip():

raise ValueError("The name entered cannot be empty or consist only of spaces.")

if not n.strip():

raise ValueError("The number of characters cannot be empty or consist only of spaces.")

if not n.isdigit():

raise ValueError("The value for the number of characters to display must be a number.")

n = int(n)

if n < 0:

raise ValueError("The number of characters must be a non-negative integer.")

return name[:n]

except ValueError as e:

return f"Error returning specified number of characters: {e}"

def reverse\_name(name, case="default"):

"""

Reverses the given string `name` and optionally applies a case transformation.

Args:

name (str): The input string to reverse.

case (str, optional): Specifies the case transformation for the reversed string.

Defaults to "default".

Options:

- "default": No case change.

- "upper": Converts to uppercase.

- "lower": Converts to lowercase.

- "invert": Capitalizes the first character of the newly reversed name.

Returns:

str: The reversed string with the specified case transformation,

or an error message if an invalid case is provided.

Example:

>>> reverse\_name("Python")

'nohtyP'

>>> reverse\_name("Python", case="upper")

'NOHTYP'

>>> reverse\_name("Python", case="invert")

'Nohtyp'

>>> reverse\_name("Python", case="invalid")

"Error: Invalid case 'invalid' entered."

"""

try:

if not name.strip():

raise ValueError("The name entered cannot be empty or consist only of spaces.")

if case.lower() not in ["default", "upper", "lower", "invert"]:

raise ValueError(f"Invalid case '{case}' entered.")

if case.lower() == "default":

return name[::-1]

elif case.lower() == "upper":

return name[::-1].upper()

elif case.lower() == "lower":

return name[::-1].lower()

elif case.lower() == "invert":

# Use join method to handle multiple names (e.g. first, middle, last)

return " ".join(word.capitalize() for word in name[::-1].split())

except Exception as e:

return f"Error reversing name: {e}"

def count\_vowels(name):

"""

Counts the number of vowels in the given string `name`, including conditional handling for 'y'.

Args:

name (str): The input string in which vowels are counted.

Returns:

int: The count of vowels in `name`.

Notes:

- Vowels include 'a', 'e', 'i', 'o', 'u' (case-insensitive).

- The letter 'y' is considered a vowel if it meets these conditions:

- It is not preceded or followed by another vowel.

- It is not at the beginning of the string.

Example:

>>> count\_vowels("rhythm")

1

>>> count\_vowels("Beautiful")

5

>>> count\_vowels("SYZYGY")

1

"""

try:

if not name.strip():

raise ValueError("The name entered cannot be empty or consist only of spaces.")

return len([

x for i, x in enumerate(name)

if x.lower() in "aeiou" or

(x.lower() == 'y' and ((i > 0 and name[i - 1].lower() not in "aeiou") and

(i == len(name) - 1 or name[i + 1].lower() not in "aeiou")))

])

except Exception as e:

return f"Error counting the number of vowels: {e}"

if \_\_name\_\_ == "\_\_main\_\_":

while True:

# Input processing

name = input("Please enter the name you'd like to process: ").strip()

n = input("Please enter the number of characters you'd like to see (including spaces, if applicable): ").strip()

print("=" \* 60) # Formatting for ease when reading output.

#first\_n\_chars

FIRST\_N\_RESULT = first\_n\_chars(name, n)

if "Error" in FIRST\_N\_RESULT:

print(FIRST\_N\_RESULT)

else:

print(f"First {n} characters in '{name}': {FIRST\_N\_RESULT}")

# reverse\_name

REVERSE\_RESULT = reverse\_name(name)

if "Error" in REVERSE\_RESULT:

print(REVERSE\_RESULT)

else:

print(f"'{name}' in reverse (default case): {reverse\_name(name)}")

print(f"'{name}' in reverse (all caps): {reverse\_name(name, 'upper')}")

print(f"'{name}' in reverse (all lowercase): {reverse\_name(name, 'lower')}")

print(f"'{name}' in reverse (invert capitalization): {reverse\_name(name, 'invert')}")

# count\_vowels

COUNT\_RESULT = str(count\_vowels(name))

if "Error" in COUNT\_RESULT:

print(COUNT\_RESULT)

else:

print(f"Number of vowels in '{name}': {COUNT\_RESULT}")

# Ask if the user wants to repeat the process

while True:

repeat = input("Would you like to process another name? (yes/no): ").strip().lower()

if repeat == "yes":

break

elif repeat == "no":

exit()

else:

print("Invalid input. Please enter 'yes' or 'no'.")