MTH 3300 Problem Set 3

Due date: Feb 23, 2025 at 11:59PM

For the programming assignment, please follow the following naming convention for your Python files.
`mth3300_<lastname>_<firstname>_pset3_part<#>.py`

As with previous problem sets, the written portion should be a separate PDF.

What will the following code snippets output? (15 points)

For each of the following, add an explanation in plain English to justify the output

1. ```
   s = "lorem ipsum"
   i, j = 0, len(s) - 1

   aux = ""
   while i < j:
     aux += f"{s[j]}{s[i]}"
     i += 1
     j -= 1

   print(aux)
   ```

   The output is going to be `mluosrpeim`. On every iteration, we look at the characters in the front and the back. We then concatenate those elements to `aux` with the character from the back going first and then the character from the front. We keep performing the above operation while increment `i` by 1 and decrementing `j` by 1 until `i == j` and then we print the value from `aux`.

2. ```
   s = "CodeMaster"
   result = ""

   for i in range(len(s) // 2):
     result += s[i] + s[-(i + 1)]

   print(result)
   ```

   The output is going to be `CroedtesMa`. Using a for-loop, we iterate over half the length of the string. On every iteration, we take the current character at index `i` and at the index `-i - 1` and append those values to `result`. Note that the characters at `-i - 1` represent the elements as if we were iterating over the string backwards.

3. ```
   text = "hello"
   new_text = ""

   for i in range(len(text)):
       new_char = chr(((ord(text[i]) - ord('a') + i) % 26) + ord('a'))
       new_text += new_char

   print(new_text)
   ```

   The output is going to be `hfnos` On every iteration, we take the ASCII value of the character at index `i` and normalize it such that it fits in the range $[0, 25]$. We then increment that normalized value by the index `i`. We use the mod operator to make sure that the incremented value still fits within the range $[0, 25]$. We then add `ord('a')` in order to denormalize the normalized ASCII value and apply `chr` so that we get the character back.

While-loops (30 points)

Note that for questions in this section, your solution should use a while-loop

1. Write a program that reads from the user a positive integer (in a decimal representation) and prints its binary (base 2) representation.

   ```
   Enter an integer: 76
   The binary representation of 76 is 1001100
   ```

   ```
   n = int(input("Enter an integer: "))
   s = ""

   while n > 0:
       s = str(n % 2) + s
       n //= 2
   ```
   ```

```
print(s)
```

2. Write a program that reads a sequence of positive integers from the user, calculates their geometric mean and prints it out to the user.

   A geometric mean of a dataset is given by the following formula:

   $$\sqrt[n]{a_1 \cdot a_2 \cdot a_3 \cdot \ldots \cdot a_n}$$

   In order to calculate the `nth` root of a number, you need to use the exponentiation operator `**`.

   Keep reading the stream of positive integers until the user enters -1.

```
Please enter a non-empty sequence of positive integers End your sequence by typing -1:
1
2
3
-1
The geometric mean is 1.8171
```

```
product = 1

n = int(
    input(
        "Please enter a non-empty sequence of positive integers; End your sequence by typing -1:\n"
    )
)
count = 0

while n != -1:
    count += 1
    product *= n
    n = int(input())

print(f"The geometric mean is {product ** (1 / count):.4f}")
```

For-loops (30 points)

Note that for questions in this section, your solution should use a for-loop

1. Write a program that asks the user to input a positive integer n, and prints the following image of an hourglass made of $2n - 1$ lines with asterisks.

   For example, if $n = 4$, the program should print:

```
*******
 *****
  ***
   *
  ***
 *****
*******
```

```
n = int(input("Enter the number of stars: "))

max_stars = 2 * n - 1
for i in range(max_stars + 1):
    if i == n:
        continue

    if i < n:
        print(" " * i + "*" * (max_stars - 2 * i))
    else:
        print(" " * (max_stars - i) + "*" * (2 * i - max_stars))
```

2. Write a program that takes a string from the user that contains at least 2 space-separated words. We want to construct a string such that the following is true:
   - Every word in the string is reversed
   - The order of the words in the sentence should remain the same

   Note that you cannot use the string `split` function here

```
Enter a string: hello world
olleh dlrow

Enter a string: Python is fun
nohtyP si nuf
```

```python
s = input("Enter a string: ")

current_word = ""
result = ""

for i in range(len(s)):
    if s[i] != " ":
        current_word += s[i]
    else:
        if len(result) > 0:
            result = f"{result} {current_word[::-1]}"
        else:
            result = current_word[::-1]

        current_word = ""

if len(current_word) > 0:
    if len(result) > 0:
        result = f"{result} {current_word[::-1]}"
    else:
        result = current_word

print(result)
```