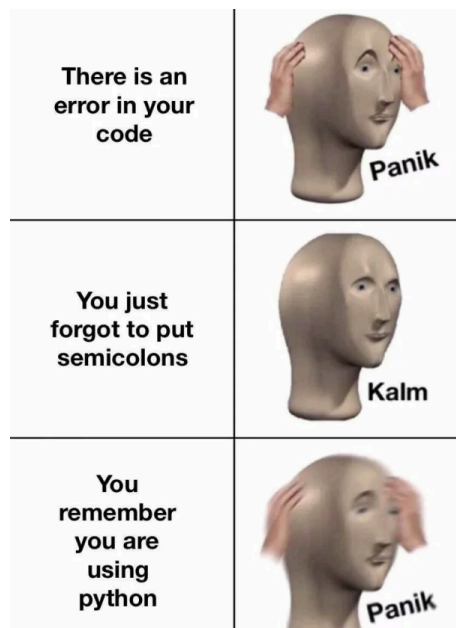


Meme:



3.1

- Explanation:

This Python code splits a string `my_str` into a list of words and then extracts every second word starting from index 1. It initializes an empty list `res`, sets `first = 1` (starting index), `second = len(my_str)` (total number of words), and `third = 2` (step size). The while loop iterates while `first < second`, appending the word at index `first` to `res` and incrementing `first` by `third` (moving two steps forward each time). As a result, `res` will contain words from `my_str` at indices 1, 3, 5, etc.

- one liner:

```
res = my_str.split()[1::2]
```

3.2

- Explanation:

This Python code creates a dictionary, `my_dictionary`, where the keys are numbers from 100 down to 1 (decreasing by 3), and the values are formatted strings describing whether each number is divisible by `n` or not, showing its remainder when divided by `n`. It iterates over `range(100, 0, -3)`, checking if each number is evenly divisible by `n` (`x % n == 0`). If it is, the corresponding dictionary value states that the number is divisible by `n`, otherwise, the value shows the remainder. Finally, `print(*my_dictionary.values())` prints all the dictionary values in a formatted way.

- one liner:

```
print(*{x: f'{x} is divided by {n}.\n' if x % n == 0 else f'the remainder of {x} divided by {n} is: {x % n}.\n' for x in range(100, 0, -3)}.values(), sep='')
```

3.3

-explanation:

This Python code iterates through ASCII values from 0 to the maximum of `ord('9')`, `ord('z')`, and `ord('Z')`, checking if each value corresponds to an alphanumeric character (a letter or a digit). If the character is alphabetic or numeric, it prints a formatted string displaying the ASCII number and its corresponding character. The `chr()` function converts the ASCII value to a character, while `isalpha()` and `isdigit()` ensure only letters and digits are included in the output. As a result, the program lists ASCII codes and their corresponding alphanumeric characters in a structured format.

- one liner:

```
print("\n".join(f'The ASCII number {i} represents the char '{chr(i)}' for i in range(0, max(ord('9'), ord('z'), ord('Z'))+1) if chr(i).isalpha() or chr(i).isdigit()))
```

3.4

-explanation:

This code initializes an empty string `tmp_chr` and then starts iterating over `list_c` item by item. For each number in `list_c`, it uses the `chr()` function to get the corresponding ASCII (or Unicode) character and appends it to `tmp_chr`. As a result, `tmp_chr` gradually builds a string that represents the sequence of characters corresponding to the values in `list_c`. After the loop completes, `print(tmp_chr)` outputs the final constructed string.

-oneliner:

```
print("".join(chr(num) for num in [80, 121, 104, 111, 110, 32, 105, 115, 32, 102, 117, 110, 33]))
```