Solution: Check for Understanding: For and While Loops

Solution: What type of loop should we use?

Question: You need to write a loop that takes the numbers in a given list named num_list:

```
num_list = [422, 136, 524, 85, 96, 719, 85, 92, 10, 17, 312, 542,
87, 23, 86, 191, 116, 35, 173, 45, 149, 59, 84, 69, 113, 166]
```

Your code should add up the odd numbers in the list, but only up to the first 5 odd numbers together. If there are more than 5 odd numbers, you should stop at the fifth. If there are fewer than 5 odd numbers, add all of the odd numbers.

Our solution:

We would write a `while` loop to write this code for the following reasons:

1. We don't need a `break` statement that a `for` loop will require. Without a `break` statement, a `for` loop will iterate through the whole list, which is not efficient.

2. We don't want to iterate over the entire list, but only over the required number of elements in the list that meets our condition.

3. It is easier to understand because you explicitly control the exit conditions for the loop.

Here's the code we wrote:

```python
num_list = [422, 136, 524, 85, 96, 719, 85, 92, 10, 17, 312, 542, 87, 23, 86, 191, 116, 35, 173, 45, 149, 59, 84, 69, 113, 166]

count_odd = 0
list_sum = 0
i = 0
len_num_list = len(num_list)

while (count_odd < 5) and (i < len_num_list):
    if num_list[i] % 2 != 0:
        list_sum += num_list[i]
        count_odd += 1
    i += 1

print ("The numbers of odd numbers added are: {}".format(count_odd))
print ("The sum of the odd numbers added is: {}".format(list_sum))
```

Consider this: If the question was to identify if *each* number in the list is an odd or even number, then a `for` loop makes better sense. In that case, you need to loop through each element in the list. However, in the question above, as long as you have the sum of the first five odd numbers (the condition), you can stop going through the list and don't need to go through the rest of the elements.

Previous   Next →   Give Page Feedback