9.12. Accumulator Pattern Strategies

9.12.1. When to Use it

When children first encounter word problems in their math classes, they find it difficult to translate those words into arithmetic expressions involving addition, subtraction, multiplication, and division. Teachers offer heuristics. If the problem says "how many…altogether", that's an addition problem. If it says "how many are left", that's going to be a subtraction problem.

Learning to use the accumulator pattern can be similarly confusing. The first step is to recognizing something in the problem statement that suggests an accumulation pattern. Here are a few. You might want to try adding some more of your own.

Phrase	Accumulation Pattern
how many	count accumulation
how frequently	
total	sum accumulation
a list of	list accumulation
concatenate	string accumulation
join together	

For example, if the problem is to compute the total distance traveled in a series of small trips, you would want to accumulate a sum. If the problem is to make a list of the cubes of all the numbers from 1-25, you want a list accumulation, starting with an empty list and appending one more cube each time. If the problem is to make a comma separated list of all the people invited to a party, you should think of concatenating them; you could start with an empty string and concatenate one more person on each iteration through a list of name.

9.12.2. Before Writing it

Before writing any code, we recommend that you first answer the following questions:

- What sequence will you iterate through as you accumulate a result? It could be a range of numbers, the letters in a string, or some existing list that you have just as a list of names.
- What type of value will you accumulate? If your final result will be a number, your accumulator will start out with a number and always have a number even as it is updated each time. Similarly, if your final result will be a list, start with a list. If your final result will be a string, you'll probably want to start with a string; one other option is to accumulate a list of strings and then use the .join() method at the end to concatenate them all together.

We recommend writing your answers to these questions in a comment. As you encounter bugs and have to look things up, it will help remind you of what you were trying to implement. Sometimes, just writing the comment can help you to realize a potential problem and avoid it before you ever write any code.

9.12.3. Choosing Good Accumulator and Iterator Variable Names

The final piece of advice regarding accumulation strategies is to be intentional when choosing variable names for the accumulator and iterator variables. A good name can help remind you of what the value is assigned to the variable as well as what you should have by the end of your code. While it might be tempting at first to use a short variable name, such as `a` or `x`, if you run into any bugs or look at your code later, you may have trouble understanding what you intended to do and what your code is actually doing.

For the accumulator variable, one thing that can help is to make the variable name end with "so_far". The prefix can be something that helps remind you of what you're supposed to end up with. For example: count_so_far, total_so_far, or cubes_so_far.

As mentioned previously in a previous Way of the Programmer segment, Naming Variables in For Loops, the iterator variable should be a singular noun. It should describe what one item in the original sequence, not what one item in the final result will be. For example, when accumulating the cubes of the numbers from 1-25, don't write for cube in range(25):. Instead, write for num in range(25):. If you name the iterator variable cube you run the risk of getting confused that it has already been cubed, when that's an operation that you still have to write in your code.

Check Your Understanding

seqmut-11-1: Does the following prompt require an accumulation pattern? If so, what words indicate that? For each string in `wrds`, add 'ed' to the end of the word (to make the word past tense). Save these past tense words to a list called `past_wrds`.

- (•) A. Yes; "save... to a list"
- ( ) B. Yes; "add 'ed' to the end of the word"
- ( ) C. No

Check me | Compare me

✔ Correct!

Activity: 1 -- Multiple Choice (question8_11_1)

seqmut-11-2: Does the following prompt require an accumulation pattern? If so, what words indicate that? Write code to sum up all of the numbers in the list `seat_counts`. Store that number in the variable `total_seat_counts`.

- (•) A. Yes; "to sum up"
- ( ) B. Yes; "numbers in the list"
- ( ) C. No

Check me | Compare me

✔ Correct!

Activity: 2 -- Multiple Choice (question8_11_2)

seqmut-11-3: Does the following prompt require an accumulation pattern? If so, what words indicate that? Write code to print out each character of the string `my_str` on a separate line.

- A. Yes; "print out each"
- B. Yes; "on a separate line"
- C. No

Check me | Compare me

✔ Correct!

Activity: 3 -- Multiple Choice (question8_11_3)

seqmut-11-4: Does the following prompt require an accumulation pattern? If so, what words indicate that? Write code that will count the number of vowels in the sentence `s` and assign the result to the variable `num_vowels`.

- A. Yes; "vowels in the sentence"
- B. Yes; "code that will count"
- C. No

Check me | Compare me

✔ Correct!

Activity: 4 -- Multiple Choice (question8_11_4)

seqmut-11-5: What type should be used for the accumulator variable in the following prompt? Write code that will count the number of vowels in the sentence `s` and assign the result to the variable `num_vowels`.

- A. string
- B. list
- C. integer
- D. none, there is no accumulator variable.

Check me | Compare me

✔ Yes, because we want to keep track of a number.

Activity: 5 -- Multiple Choice (question8_11_5)

seqmut-11-6: What sequence will you iterate through as you accumulate a result in the following prompt? Write code that will count the number of vowels in the sentence `s` and assign the result to the variable `num_vowels`.

- A. num_vowels
- B. s
- C. the prompt does not say

Check me | Compare me

✔ Yes, that is the sequence you will iterate through!

Activity: 6 -- Multiple Choice (question8_11_6)

seqmut-11-7: What type should be used for the accumulator variable in the following prompt? For each string in `wrds`, add 'ed' to the end of the word (to make the word past tense). Save these past tense words to a list called `past_wrds`.

- A. string
- B. list
- C. integer
- D. none, there is no accumulator variable.

Check me | Compare me

✔ Yes, because we want a new list at the end of the code.

Activity: 7 -- Multiple Choice (question8_11_7)

seqmut-11-8: What sequence will you iterate through as you accumulate a result in the following prompt? For each string in `wrds`, add 'ed' to the end of the word (to make the word past tense). Save these past tense words to a list called `past_wrds`.

- A. wrds
- B. past_wrds

C. the prompt does not say

Check me Compare me

✔ Yes, that is the sequence you will iterate through!

Activity: 8 -- Multiple Choice (question8_11_8)

seqmut-11-9: What type should be used for the accumulator variable in the following prompt? Write code to sum up all of the numbers in the list `seat_counts`. Store that number in the variable `total_seat_counts`.

A. string
B. list
C. integer
D. none, there is no accumulator variable.

Check me Compare me

✔ Yes, because we want to keep track of a number.

Activity: 9 -- Multiple Choice (question8_11_9)

seqmut-11-10: What sequence will you iterate through as you accumulate a result in the following prompt? Write code to sum up all of the numbers in the list `seat_counts`. Store that number in the variable `total_seat_counts`.

A. seat_counts
B. total_seat_counts
C. the prompt does not say

Check me Compare me

✔ Yes, that is the sequence you will iterate through!

Activity: 10 -- Multiple Choice (question8_11_10)

seqmut-11-11: What type should be used for the accumulator variable in the following prompt? Write code to print out each character of the string `my_str` on a separate line.

A. string
B. list
C. integer
D. none, there is no accumulator variable.

Check me Compare me

✔ Correct, because this prompt does not require an accumulator pattern

Activity: 11 -- Multiple Choice (question8_11_11)

seqmut-11-12: What sequence will you iterate through as you accumulate a result in the following prompt? Write code to print out each character of the string `my_str` on a separate line.

A. my_str
B. my_str.split()
C. the prompt does not say

Check me Compare me

✔ Yes, that is the sequence you will iterate through!

Activity: 12 -- Multiple Choice (question8_11_12)

seqmut-11-13: Which of these are good alternatives to the accumulator variable and iterator variable names for the following prompt? For each string in `wrds`, add 'ed' to the end of the word (to make the word past tense). Save these past tense words to a list called `past_wrds`.

A. Accumulator Variable: wrds_so_far ; Iterator Variable: wrd
B. Accumulator Variable: wrds_so_far ; Iterator Variable: x
C. Accumulator Variable: changed_wrds ; Iterator Variable: ed

Check me Compare me

✔ Yes, this is the most clear combination of accumulator and iterator variables.

Activity: 13 -- Multiple Choice (question8_11_13)

seqmut-11-14: WWhich of these are good alternatives to the accumulator variable and iterator variable names for the following prompt? Write code that will count the number of vowels in the sentence `s` and assign the result to the variable `num_vowels`.

A. Accumulator Variable: count_so_far ; Iterator Variable: l

B. Accumulator Variable: total_so_far ; Iterator Variable: letter

C. Accumulator Variable: n_v ; Iterator Variable: letter

Check me Compare me

✔ Yes! Both the accumulator and iterator variable are clear.

Activity: 14 -- Multiple Choice (question8_11_14)

seqmut-11-15: Which of these are good alternatives to the accumulator variable and iterator variable names for the following prompt? Write code to sum up all of the numbers in the list `seat_counts`. Store that number in the variable `total_seat_counts`.

A. Accumulator Variable: total_so_far ; Iterator Variable: seat

B. Accumulator Variable: total_seats_so_far ; Iterator Variable: seat_count

C. Accumulator Variable: count ; Iterator Variable: n

Check me Compare me

✔ Yes, this is the most clear combination.

Activity: 15 -- Multiple Choice (question8_11_15)

seqmut-11-16: Which of these are good alternatives to the accumulator variable and iterator variable names for the following prompt? Write code to print out each character of the string `my_str` on a separate line.

A. Accumulator Variable: character_so_far ; Iterator Variable: char

B. Accumulator Variable: no variable needed ; Iterator Variable: c

C. Accumulator Variable: no variable needed ; Iterator Variable: char

Check me Compare me

✔ Yes, there is no accumulator variable needed and the iterator variable is clear (char is a common short form of character)

Activity: 16 -- Multiple Choice (question8_11_16)

You have attempted 17 of 16 activities on this page

9.11. The Accumulator Pattern with Strings"> 9.13. Don't Mutate A List That You Are Iterating Through">

e Accumulator Pattern with Strings">

9.13. Don't Mutate A List That You Are Iterating Through">Next Section - 9.13. Don't Mutate A List That You Are Iterating Through

| Back to top