For Loops

Overview

This is a tutorial on the programming concept called a for loop. It is intended for people learning their first programming language who do not yet know what for loops are. This tutorial is not language-specific and uses pseudocode instead of actual code. After this tutorial, readers should be able to write for loops in pseudocode that iterate over counters and lists.

Prerequisites

Experience reading and writing **pseudocode** is a recommended prerequisite to this tutorial. **Pseudocode** is an informal, plain English description of code that is not specific to a programming language. Each sample of pseudocode in this tutorial is just one possible way of representing the desired program.

You should also be familiar with **functions** and **if statements** before proceeding. A **function** is a piece of code that performs a specific task. An **if statement** uses a true-false condition to determine whether your program should execute a certain piece of code.

Read the following tutorials before proceeding if pseudocode, functions or if statements are new to you:

What is Pseudocode?

What is an if statement?

What is a function?

What is a for loop?

A **for loop** is a programming structure used to repeat a series of operations. Each repetition is known as an **iteration**. For loops allow us to write one piece of code to perform the same operations repeatedly, such as on different numbers in a sequence or items in a list, instead of writing the same code multiple times.

For loop with counter

Let's begin with a program that repeats just one operation: the print function. Say we want to print each number from 1 through 3 in increasing order. This can be done with the following pseudocode:

```
1. print 1
```

```
2. print 2

3. print 3
```

We can rewrite the above using a for loop as:

```
1. for i from 1 to 3, increase i by 1

2.    print i
```

Line 1 sets up the for loop with the following rules:

- `for i from 1 to 3` means the for loop iterates over values of the variable `i`, which acts as a counter. In this statement:
  - `i` is initialized to 1.
  - The for loop continues while `i` is less than or equal to 3. When i exceeds 3, the loop terminates.
- `increase i by 1` means the value of `i` increases by 1 at the end of each iteration.

Line 2 specifies that at each iteration, the program executes `print i`.

Figure 1 breaks down what happens in each iteration.

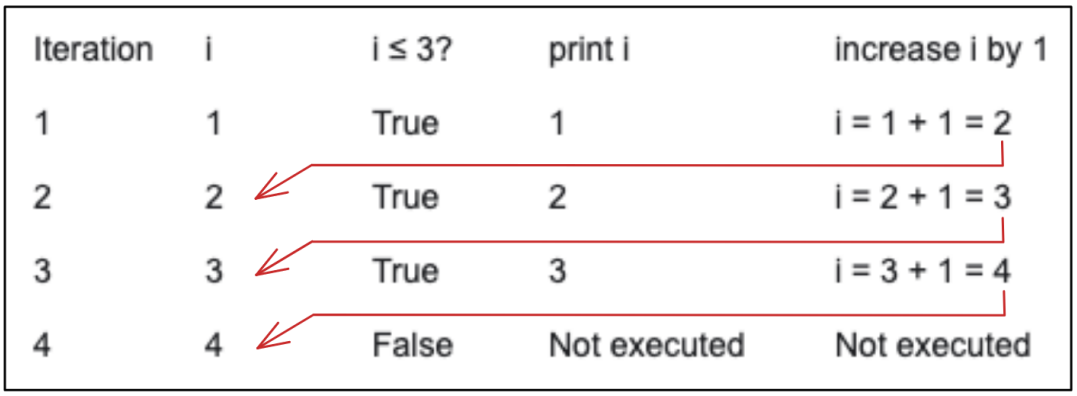

Figure 1. Counter for loop step-by-step.

The output of this program is:

```
1

2

3
```

Practice exercise 1

For loops can use counters that change by any amount. Write pseudocode for a for loop which prints every positive odd number between 4 and 20 in descending order. You may use the above example and Figure 1 as a model.

Possible answer:

```
1. for i from 20 to 4, decrease i by 2

2.    print i
```

For loop with list

Let's say you want to list all the professors teaching a course next semester at your university. You have a course catalog you can use, where entries read "Course Title, Day and Time, Professor Name, Building". Without attempting to code this, here is how you might do it by hand.

The first entry in the catalog reads: "Physics, Monday, taught by Professor Newton in Inertia Hall". You take the professor's name, "Newton", write it down, and proceed to the next item in the catalog. You can imagine repeating this for each item, skipping names you already recorded, until you reach the end of the catalog. The result would be a list of all professors teaching a course next semester.

Now, imagine the **course catalog** is available online and you can create this list of professor names programmatically. We just learned about for loops, so let's try using one.

Imagine we already wrote a function, **getProfessorFor**, that returns the name of the professor teaching any course.

```
1 professors_list = empty list

2 for course in course catalog, step course to next entry:

3     professor = getProfessorFor course

4     if professor is not in professors_list

5           add professor to professors_list

5 print professors_list
```

Line 1 initializes the list of professor names, `professors_list`, to empty at the beginning because we have not started reading the course catalog yet. The for loop begins on line 2 with the following rules:

- `for course in catalog` means the for loop iterates over the variable course.
    - Course is initialized as the first entry.

    - The for loop continues for all courses in the catalog, and terminates after the last entry.
- `set course to next entry` means the value of course is set to the next entry in the catalog at the end of each iteration.

Lines 3-4 comprise the body of our for loop, which first extracts the professor's name from `course`, then checks if the name is already in `professors_list`, and finally adds it to the list if it is not. The complete list of professor names is output at the end of the program on line 5.

Figure 2 breaks down what happens in each iteration for the following course catalog:

1. Physics, Professor Newton, Monday, Inertia Hall
2. Biology, Professor Franklin, Friday, Helix Library
3. Genetics, Professor Franklin, Tuesday, RNA Lab

Note that the notation [a, b, c] represents a list containing elements a, b and c. [ ] represents an empty list.

Iteration	course	course in catalog?	professor = getProfessorFor course	professor not in professors_list?	professors_list	course = next entry
1	Physics 101, Dr. Newton, Monday 10am, Inertia Hall	True	professor = Dr. Newton	Dr. Newton not in [ ] = True	[Dr. Newton]	Biology 200, Dr. Franklin, Friday 3pm, Helix Library
2	Biology 200, Dr. Franklin, Friday 3pm, Helix Library	True	professor = Dr. Franklin	Dr. Franklin not in [Dr. Newton] = True	[Dr. Newton, Dr. Franklin]	Genetics 100, Dr. Franklin, Tuesday 1pm, RNA Lab
3	Genetics 100, Dr. Franklin, Tuesday 1pm, RNA Lab	True	professor = Dr. Franklin	Dr. Franklin not in [Dr. Newton, Dr. Franklin] = False	[Dr. Newton, Dr. Franklin]	None
4	None	False	Not executed	Not executed	Not executed	Not executed

Figure 2. List for loop step-by-step.

The output of this program is:

```
[Dr. Newton, Dr. Franklin]
```

Practice exercise 2

Imagine an **enrollment list** is available online with the name and major of each student at your university. Use pseudocode to write a program that uses a for loop to iterate through the enrollment list and create a list of all students majoring in Math, and then prints that list. Assume you have a function **getMajorFor** a student that returns the major for any student.

Possible answer:

```
1. math_student_list = empty list
2. for student in enrollment_list, step student to next entry:
3.    if getMajorFor student is math:
4.          add student to math_student_list
5. print math_student_list
```