Flow Control

Chapter 04

Kibret Zewdu (MSc.)

Department of Computer Science
College of Informatics
University of Gondar

January 26, 2023

Outline

Introduction

- Python has only three flow control structures.
  - There is one conditional and two iteration structures.
- A **conditional structure** (**if**) determines, after an expression evaluation, whether a block of code is executed or not.
- **Iteration structures** (**for** and **while**) allow multiple executions of the same code portion.
- How many times is the code associated with an iteration structure executed?
  - A **for** cycle executes a code block many times as elements are available in a specified iterable element,
  - **while** the code under a while cycle is executed until a given condition turns false.

IF-ELSE

- **If** evaluates an expression.
- If the expression is true, the block of code just after the if clause is executed. Otherwise, the block under ***else*** is executed.
- A basic schema of an if-else condition,

```
if EXPRESSION:
        BLOCK1
else:
        BLOCK2
```

EXPRESSION must be an expression that returns **True** or **False**
Example:

```
age = input("How old are you? )
if age >= 18:
        print('You are allowed to vote!')
else:
        print('You are not allowed to vote!')
```

IF-ELSE

- To evaluate more than one condition, use **elif**:

```
if EXPRESSION1:
        BLOCK1
elif EXPRESSION2:
        BLOCK2
elif EXPRESSION3:
        BLOCK3
else:
        BLOCK4
```

IF-ELSE

What is True

- Nonempty data structures (lists, dictionaries, tuples, strings, sets).
- Empty data structures count as False.
- 0 and None count as False (while other values count as True).
- Keyword True is True and False is False.
- If you have a doubt if an expression is True or False, use `bool()`:

```
>>> bool(1=='1')
False
```

IF-ELSE

Example

```
if x > y:
        print(x, ' is larger number')
elif x < y:
        print(y, ' is larger number')
else:
        print('Both numbers are equal')
```

IF-ELSE

Example

Nested If

```
dna = input('Enter your DNA sequence: ')
seqsize = len(dna)
if seqsize == 0:
        print('You must enter something!')
elif 0 < seqsize < 10:
        print('Your primer must have at least ten nucleotides')
elif seqsize < 25:
        print('This size is OK')
else:
        print('Your primer is too long')
```

IF-ELSE

Conditional Expressions

Special syntax to write an if condition in one line
expression1 if condition else expression2
This line will take the value of expression1, if condition is true; otherwise, it will take the value of expression2.

```
>>> total = 5
>>> items = 2
>>> print('Average = {0}'.format(total/items if items != 0 else 'N/
Average = 2.5
```

FOR LOOP

This control structure allows code to be repeatedly executed while keeping a variable with the value of an iterable object.
The generic form of a for loop:

```
for var in ITERABLE:
        BLOCK
```

For example:

```
for each_item in some_list:
        # Do something with each_item
        print(each_item)
```

FOR LOOP

In the following code, ***for*** walks through a list (bases) with four elements. On each iteration, ***x*** takes the value of one of the elements in the list.

```
>>> bases = ['C', 'T', 'G', 'A']
>>> for x in bases:
        print(x)

C
T
G
A
```

Note: The most common iterable objects are lists, tuples, strings, and dictionaries.

FOR LOOP

To know the position on the iterable you are iterating, the method enumerate will return the index of the iterable along with the value.

```
>>> bases = ['C', 'T', 'G', 'A']
>>> for n, x in enumerate(bases):
        print(n, x)

0 C
1 T
2 G
3 A
```

FOR LOOP

The for loop is used to allow a block of code to run a number of times while changing a counter variable.

```
>>> for n in [0, 1, 2, 3, 4]:
        print(n)

0
1
2
3
4
```

Another alternative to iterate thought a list of numbers, is to generate them with the built-in function `range(n)`. This function returns an iterable object. Which each time you call it, returns a number, from 0 to the first parameter entered in the function minus one (that is, n-1).

FOR LOOP

- Another alternative to iterate thought a list of numbers, is to generate them with the built-in function `range(n)`.
- This function returns an iterable object.
- Which each time you call it, returns a number, from 0 to the first parameter entered in the function minus one (that is, n-1).

```
>>> for x in range(4):
        print(x)
```

Output

```
0
1
2
3
```

FOR LOOP

Example: molecular weight of a protein

The following code calculates the molecular weight of a protein based on its individual amino acids.

```
prot_seq = input('Enter your protein sequence: ')
prot_weight = {'A':89, 'V':117, 'L':131, 'I':131, 'P':115,
               'F':165, 'W':204, 'M':149, 'G':75, 'S':105,
               'C':121, 'T':119, 'Y':181, 'N':132, 'Q':146,
               'D':133, 'E':147, 'K':146, 'R':174, 'H':155}
total_weight = 0
for aa in prot_seq:
        total_weight = total_weight + prot_weight.get(
                aa.upper(), 0)
total_weight = total_weight - (18 * (len(prot_seq) - 1))
print('The net weight is: {0}'.format(total_weight))
```

WHILE LOOP

- it also executes a code portion in a repeated way
- here there is no iteration over an object, so this loop doesn't end when the iteration object is traversed, but when a given condition is not true. Syntax of while loop:

```
while EXPRESSION:
        BLOCK
```

Note: Avoid infinite loop

```
>>> a = 10
>>> while a < 40:
        print(a)
        a += 10


10
20
30
```

BREAK: BREAKING THE LOOP

- break is used to escape from a loop structure

```
>>> a = 10
>>> while True:
        if a < 40:
                print(a)
        else:
                break
        a += 10

10
20
30
```

BREAK: BREAKING THE LOOP

Example: Searching a value in a list of tuples using FOR

```
color_code = [('red',1), ('green',2), ('blue',3), ('black',4)]
name = 'blue'
for color_pair in color_code:
        if name == color_pair[0]:
                code = color_pair[1]
                break
print(code)
3
```

BREAK: BREAKING THE LOOP

Example: Searching a value in a list of tuples using WHILE

```
color_code = [('red',1), ('green',2), ('blue',3), ('black',4)]
name = 'blue'
i = 0
while name != color_code[i][0]:
        i += 1
code = color_code[i][1]
print(code)
3
```

BREAK: BREAKING THE LOOP

Example: Searching a value in a list of tuples using a dictionary

```
color_code = [('red',1), ('green',2), ('blue',3), ('black',4)]
name = 'blue'
color_code_d = dict(color_code)
print(color_code[name])
3
```

Reading - Additional Examples

- Estimate the Net Charge of a Protein
- Search for a Low-Degeneration Zone

Exercises

1. Given a protein sequence in the one-letter code, calculate the percentage of methionine (M) and cysteine (C). For example, from **MFKFASAVILCLVAASSTQA** the result must be 10% (1 M and 1 C over 20 amino acids).
2. Make a program that converts everything you type into Leetspeak, using the following equivalence: 0 for O, 1 for I (or L), 2 for Z (or R), 3 for E, 4 for A, 5 for S, 6 for G (or B), 7 for T (or L), 8 for B, and 9 for P (or G and Q). So "Hello world!" is rendered as "H3770 w02ld!"