While Loops

Introduction to Computer Programming

Dr. Paul Vrbik

October 16, 2018

Definition (Loop)

A loop is a control structure that repeats code that belongs to it.

Definition (While-Loop)

A while-loop is a control structure that repeats code while some condition is satisfied.

Definition (While-Loop)

```
while <condition>:
    <code>
```

```
>>> x = 0
>>> while x < 10:
...    x += 1

>>> x
10
```

```
>>> x = 0
>>> while True:
...    print(x)
...    x += 1
0
1
2
3 ...
```

There is usually a key-stroke that terminates a loop — it is a good idea to learn what it is in your IDE!

```
>>> x = 0
>>> while False:
...    print(x)
...    x += 1
```

Nothing prints.

Motivation

There are (at least) two scenarios where for-loops are insufficient:

1. prompting the user for valid input, and
2. playing a random game (i.e. stochastic process).

Random number generation is handled by an external library. It is not built-in and therefore must be imported.

Single Command

```
>>> from random import randint
>>> randint(1,6)
2
```

Chosen uniformly over the interval.

Entire library

```
>>> import random
>>> random.randint(1,6)
2
```

A word on random number generation

Generating truly random numbers from software alone is impossible (it is possible with hardware).

Software generates pseudorandom numbers meaning they appear random but are generated by deterministic methods.

Consequently, you can generate the same sequence of random numbers (useful for testing) using `random.seed`.

```
>>> random.seed(1)
>>> random.randint(1, 10**4)
2202
>>> random.randint(1, 10**4)
9326
```

```
>>> random.seed(1)
>>> random.randint(1, 10**4)
2202
>>> random.randint(1, 10**4)
9326
```

Question

Write a function that prints the result of a six-sided dice roll until a six is rolled. Return the number of rolls required.

Answer

```
def game01() -> int:
```

Question

Write a function that accumulates the result of dice rolls until some target sum is achieved. Return the number of rolls required.

Answer

```
def game02(num_dice_sides:int, target:int) -> int:
```

Definition (Break)

The break keyword terminates a loop immediately and continues.

```
>>> while True:
...    print("hello")
...    break
...    print("world")
hello
```

Definition (Do-While)

A do-while or repeat-until are while-loop variants available in other languages.

```
do:
    <code>
while <condition>
```

A do-while executes its code at least once.

Do-While

Although not natively supported in Python, we can simulate them.

```
>>> x = 0
>>> while 1:
...     x += 1
...     if x > 0:
...         break
>>> x
1
```

Alternatively

```
>>> x = 0
>>> while 1:
...    x += 1
...    break if x > 0 else continue
>>> x
1
```

`continue` is basically the opposite of `break`.

Question

Write a function that prints the result of a six-sided dice-rolls and stops once a one is rolled.

Answer

```
def game03() -> NoneType:
```

Converting For to While

For every for-loop there is a while-loop equivalent (but not the converse).

Question

Write the following for-loop as a while-loop.

```
>>> for x in "abcdef":
...     print(x)
```

Answer

```
>>> i = 1
>>> while i <= 10:
...    print("abcdef"[i])
...    i += 1
```

or equivalently

```
>>> i = 0
>>> while i <= 10:
...    i += 1
...    print("abcdef"[i])
```

Question

Write the following for-loop as a while-loop.

```
>>> for x in range(a, b, c):
...    print(x)
```

Answer

```
>>> x = a
>>> while x < b:
...    x += c
...    print(x)
```

Question

Find a while-loop that has no for-loop equivalent.

Answer

`game01`.

Question

Write a function which engages the user in a guessing game.

Repeatedly ask the user to input a number between 1 and 100, until they guess the correct (randomly selected) number.

What is the best guessing strategy?

Answer

```
def game04() -> None:
```

Question

Implement the “string slice” `s[::k]` as the function

```
def slice_k(s:string, k:int) -> string
```

using a while-loop.

Next Time

1. Lists.