CS65: Introduction to Computer Science

Random number
Loop: the while loop

DRAKE UNIVERSITY

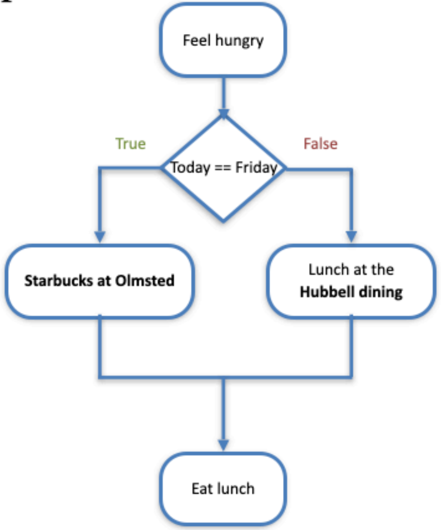
Md Alimoor Reza
Assistant Professor of Computer Science

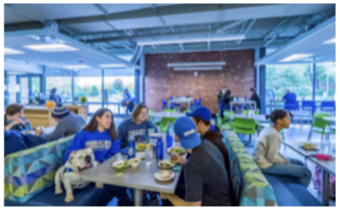
Recap

- **Boolean** datatype, which has **true** or **false** values

- Selection statements are useful for branching inside your program
  - if
  - if-else
  - if-elif-…-else

- Lunch on campus





Recap: ’**bool**’ Data Type

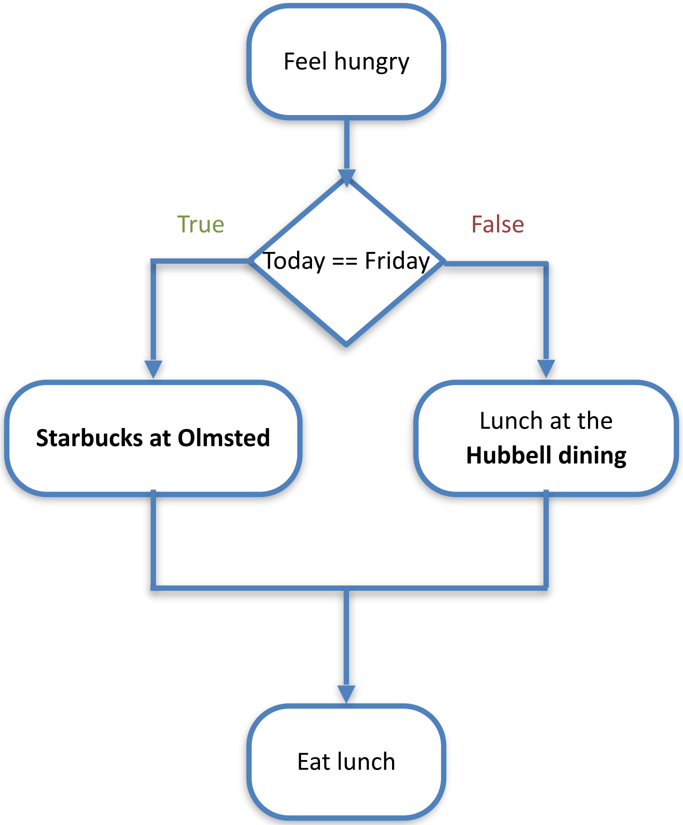
- Notion of something being true and being false — represented with two ‘**bool**’ data types:
  - True
  - False

- Allows us to evaluate true or false questions — in real life, we always encounter question with Yes or No answer

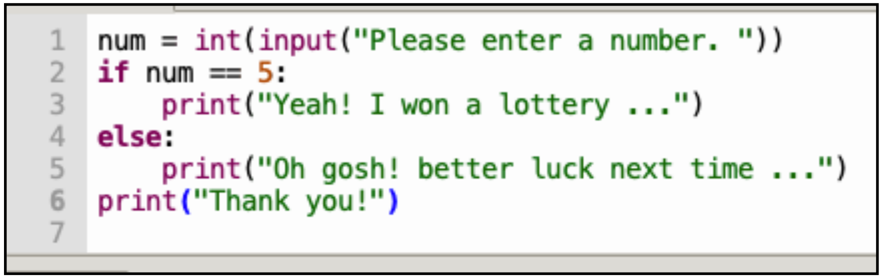
- Logical and comparison operators:
  - Boolean expression with logical operator (and, or, not)
  - Boolean expression with comparison operator (<, <=, >, ==, etc)

Recap: Selection Statements

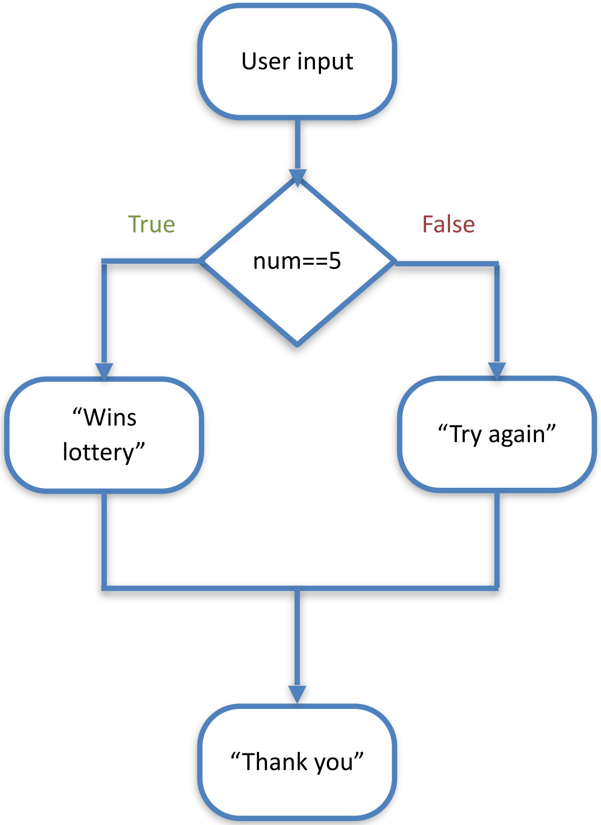
- Program taking one *path* or *branch* of the code instead of taking another, based on the **boolean expression**'s value
- This feature allows to ask true/false questions in the code. Depending on the boolean answer (True or False), the program will execute a specific branch



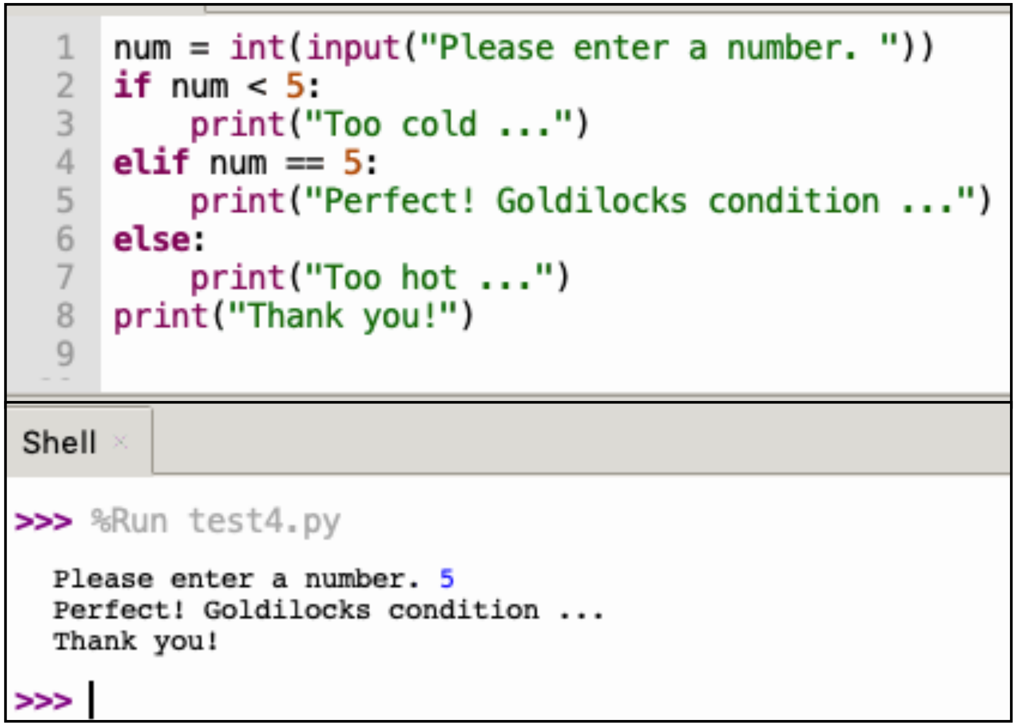
Recap: ’if … else’ Statement



```python
num = int(input("Please enter a number. "))
if num == 5:
    print("Yeah! I won a lottery ...")
else:
    print("Oh gosh! better luck next time ...")
print("Thank you!")

```

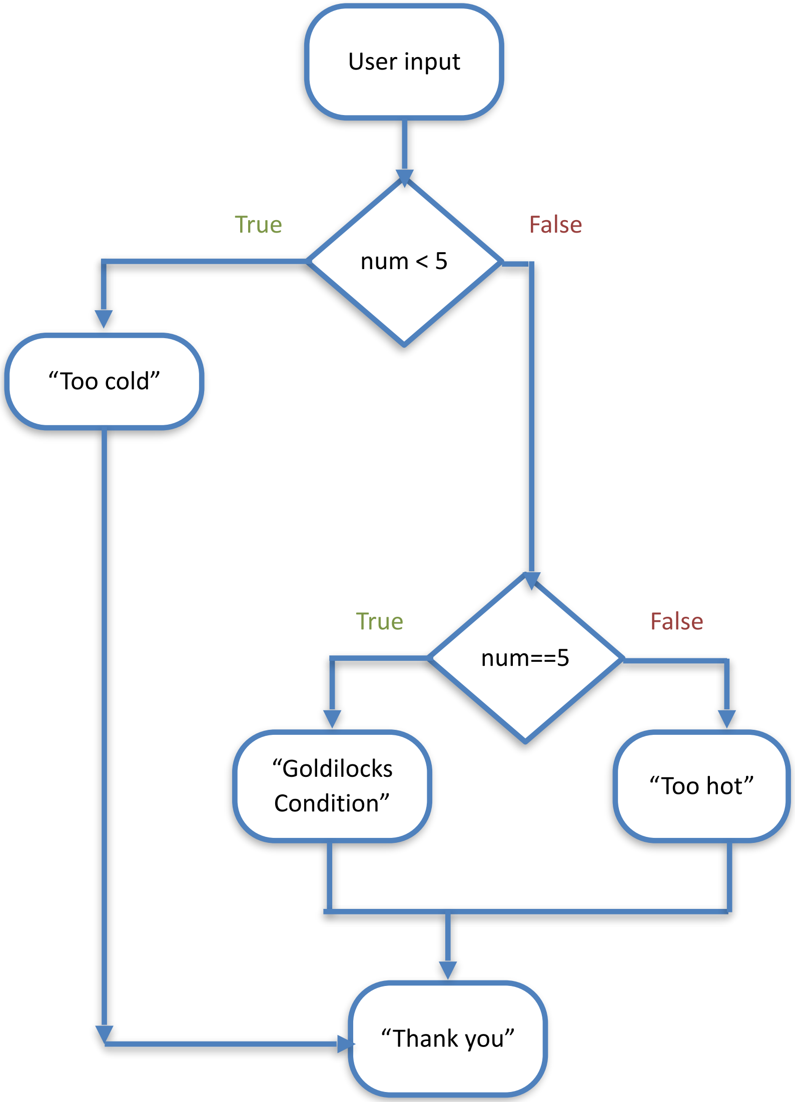


Recap: Multiple Selections



```
num = int(input("Please enter a number. "))
if num < 5:
    print("Too cold ...")
elif num == 5:
    print("Perfect! Goldilocks condition ...")
else:
    print("Too hot ...")
print("Thank you!")
```

```
>>> %Run test4.py

  Please enter a number. 5
  Perfect! Goldilocks condition ...
  Thank you!

>>> 
```



Topics

- Random number
    - Useful for your Assignment 1

- The while loop

- Assignment 1 has been released
    - Due: March, 03 (Thursday)

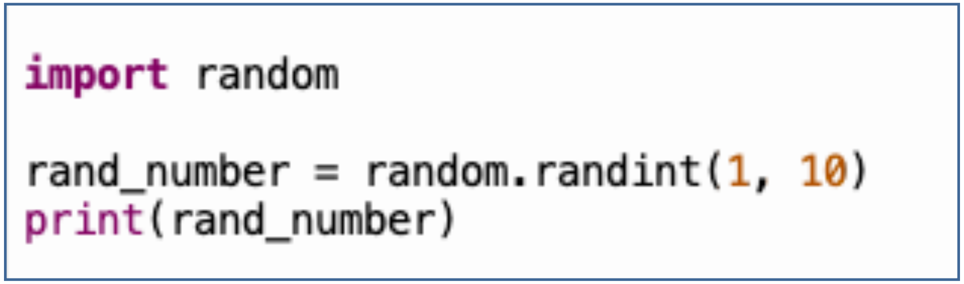
Random Number

- Random numbers are useful several programming tasks:
  - Simulating a coin toss — random flipping of head or tail
  - Simulating a dice roll — random roll of one of six sides
  - Simulating a card shuffling from 52 cards

- Python provides library to generate random numbers
  - Like math module or graphics module, you can import random module to get access to random number generating functions

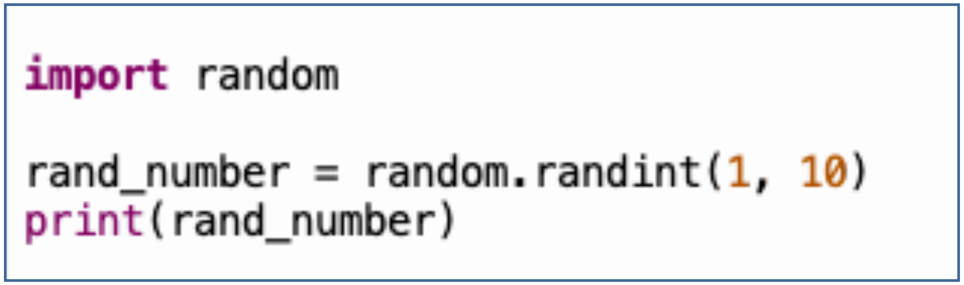
Random Number

- Steps for generating a random number are as follows:
  - Step 1: Import the **random** module
  - Step 2: Generate a random number (eg, an integer number) between a range of values denoted by a lower_range and an upper_range
    - For example, in order to generate a random integer between lower_range of 1 and upper_range of 10, we need to do the following:



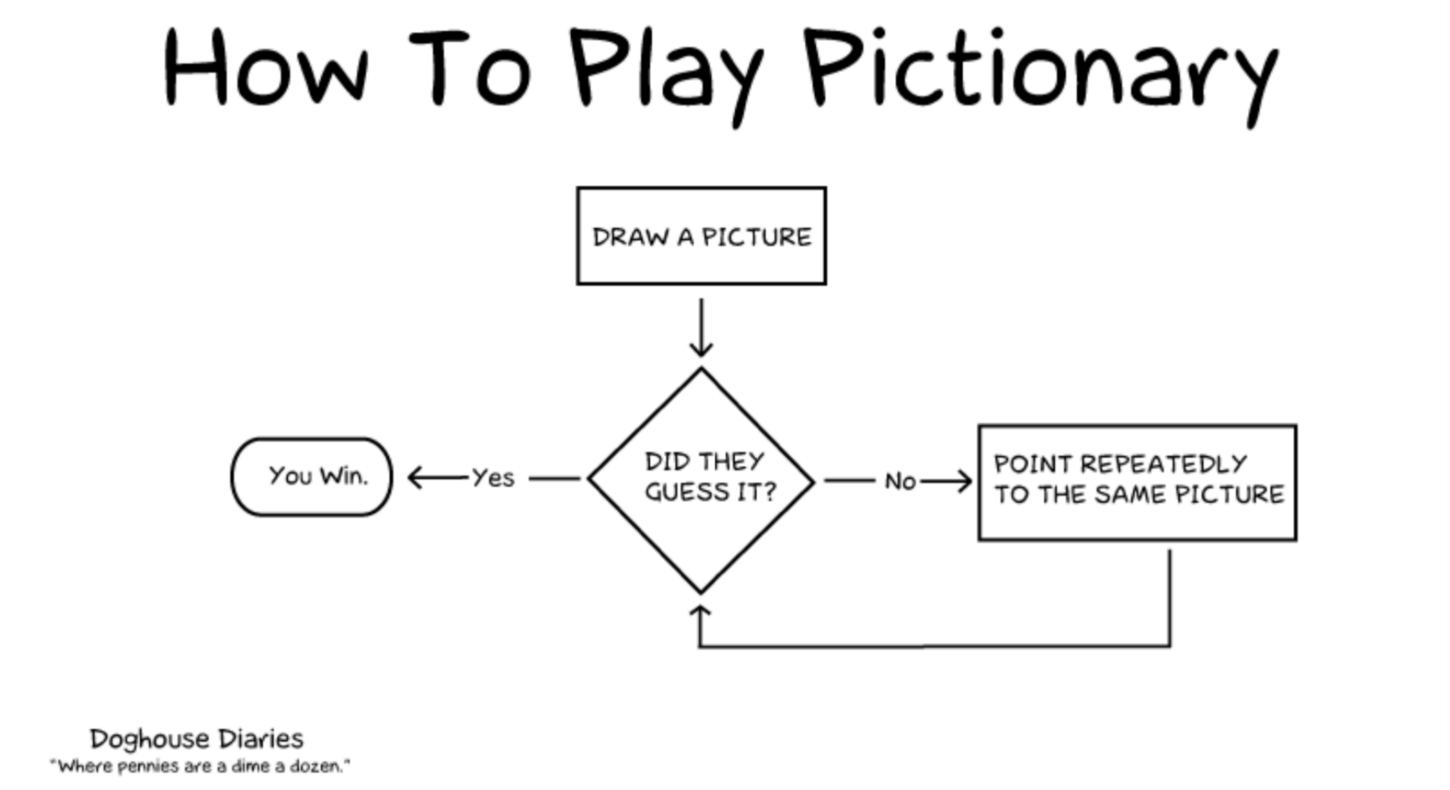
```
import random

rand_number = random.randint(1, 10)
print(rand_number)
```

Demo: Random Number



```python
import random

rand_number = random.randint(1, 10)
print(rand_number)
```

Motivation: Loop



http://www.thedoghousediaries.com/2659

Topic: Solving Repetitive Task

- Designed to solve a repetitive task — runs a block of code based on a Boolean expression:
  - Summing all the numbers from 0 to 100
  - Taking user inputs until a special number is provided

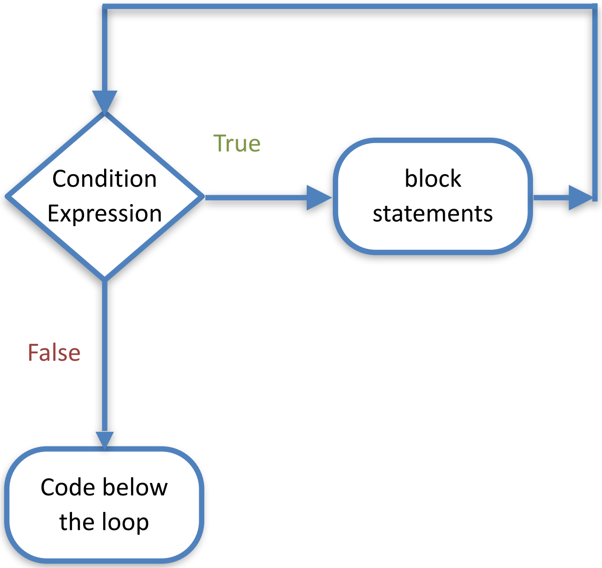
- Two different ways to solve a repetitive task in Python
  - The **while** loop
  - The **for** loop

The **while** Loop

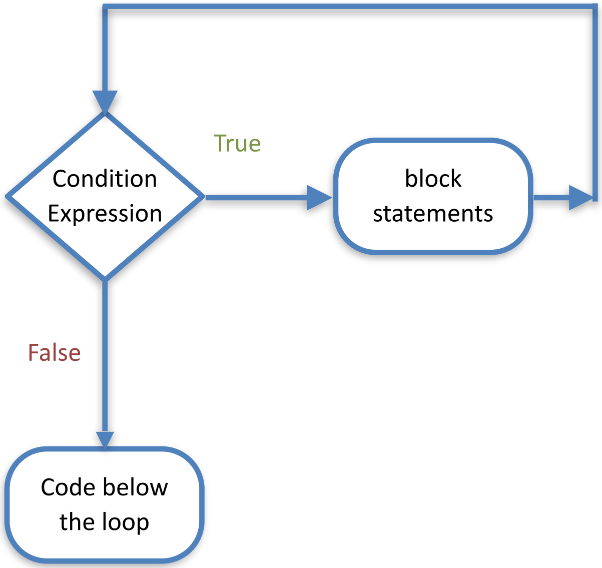
- while **<condition expression> :**

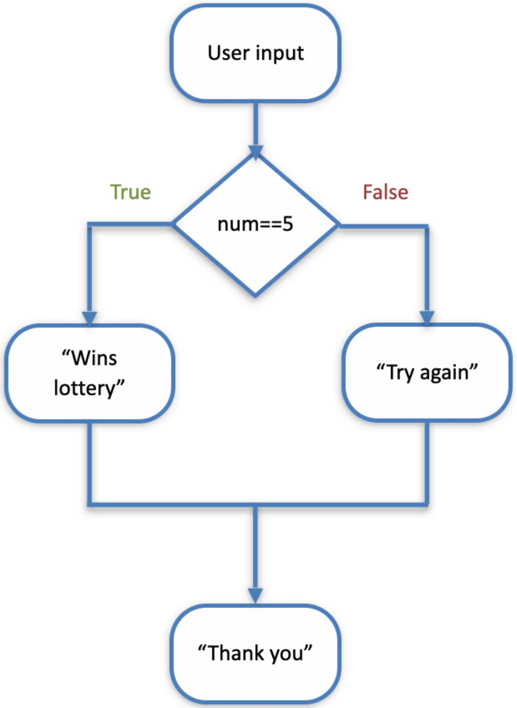
  <block statements>

- **condition expression**: a boolean expression
- **block statements**: statements to be executed if result of the condition expression is **True**
- Unlike if statement, the **<block statements>** will repeatedly be executed until the **<condition expression>** becomes False



while loop
vs
if/else selection statements



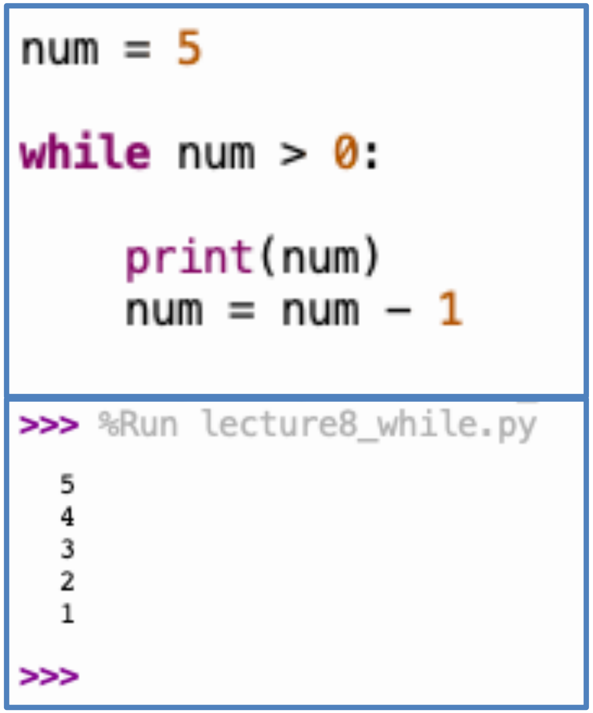


While loop

If/else blocks

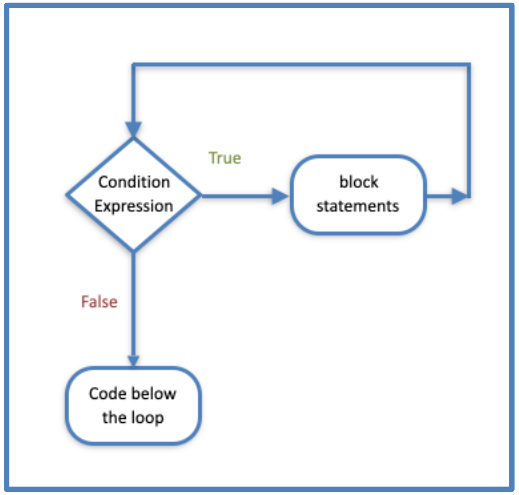
The while Loop

- Section of code that repeats — designed to solve a repetitive task
  - decrease the value of a variable by 1 until it becomes negative



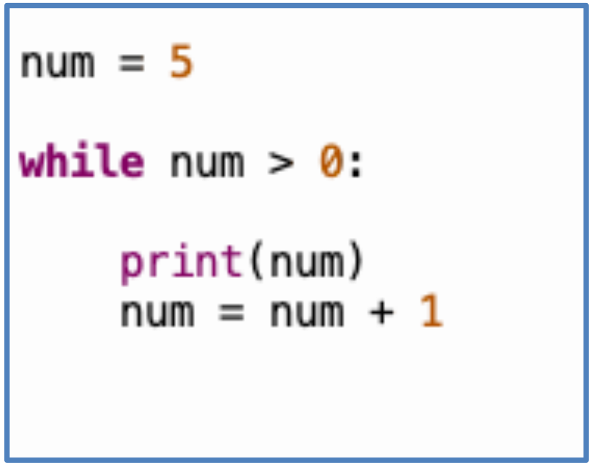
```
num = 5

while num > 0:

    print(num)
    num = num - 1
```

```
>>> %Run lecture8_while.py

  5
  4
  3
  2
  1

>>>
```

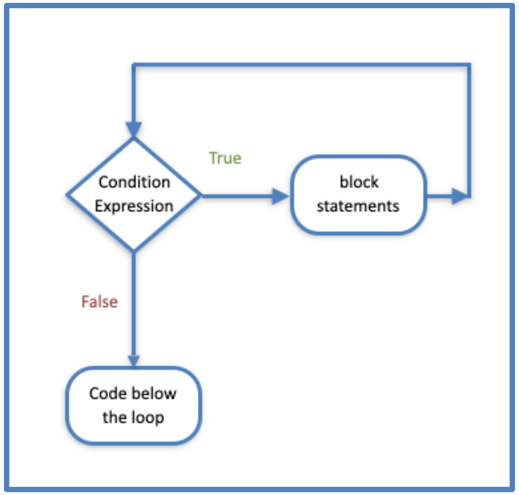


The while Loop

- **<u>Infinite loop:</u>** section of code that repeats <u>forever</u>
  - The condition expression should be designed carefully so that the loop terminates after a certain number of iterations



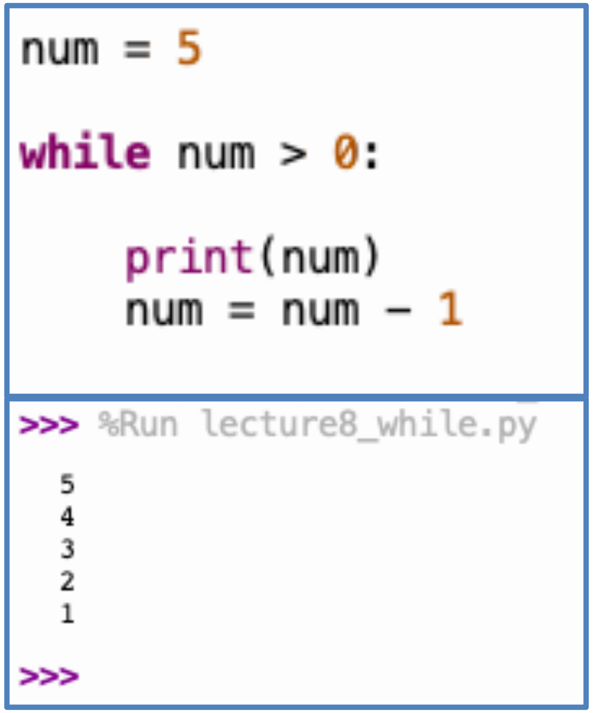
```
num = 5

while num > 0:

    print(num)
    num = num + 1
```



What will happen?

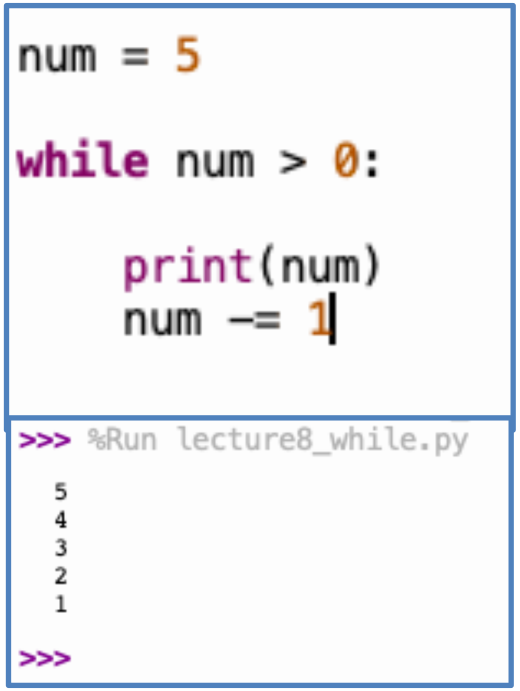
The while Loop

- The index variable can be updated (**decreased**) with a shorthand:



```
num = 5

while num > 0:

    print(num)
    num = num - 1
```

```
>>> %Run lecture8_while.py

  5
  4
  3
  2
  1

>>>
```

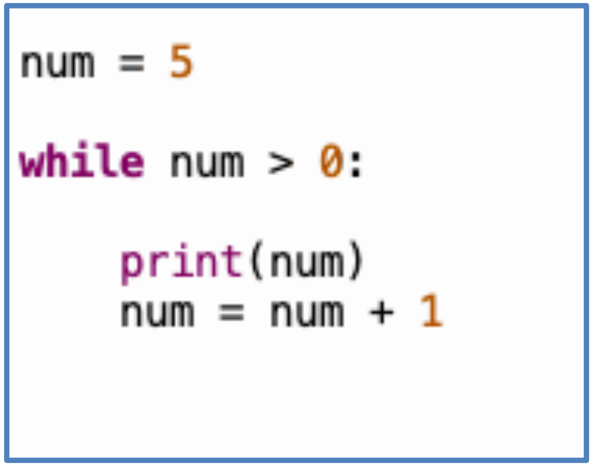


```
num = 5

while num > 0:

    print(num)
    num -= 1
```

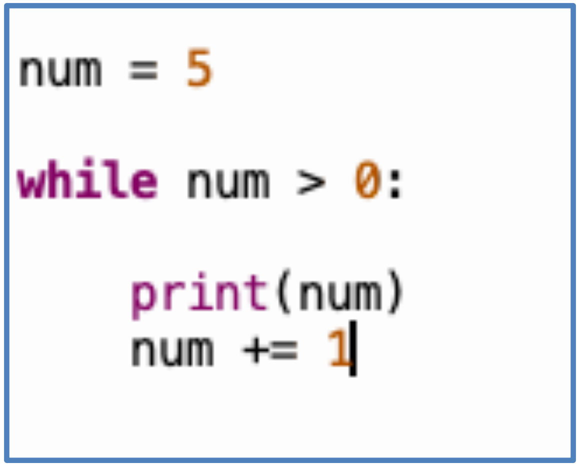
```
>>> %Run lecture8_while.py

  5
  4
  3
  2
  1

>>>
```

The while Loop

- The index variable can be updated (**<u>increased</u>**) with a shorthand:



```
num = 5

while num > 0:

    print(num)
    num = num + 1
```



```
num = 5

while num > 0:

    print(num)
    num += 1
```

Exercise 1

- Write a code that will do the following:
    - prompt the user for an integer (between 1 to 100)
    - then **computes** the sum of all number from 0 to the given number

Exercise 2

- Write a code that will do the following:
    - prompt the user for an integer number (between 1 to 100)
    - then **prints** all the even numbers between 0 and the given number

Exercise 3

- Write a code that will do the following:
    - prompt the user for a state's name from the following:
        - {"NY", "PA", "MD", "VA"},
        - {"IA", "IN", "IL", "MN"},
        - {"TX", "LA", "FL", "AK"},
        - {"CA", "OR", "WA", "NV"}
    - then prints its geographic location from one of the categories:
        - "Eastern", "Midwestern", "Southern", "Western"
    - program will terminate only when the user enters "END"

Announcements

- Assignment 1 has been released. You have two weeks (due by 03/03/22)

- Quiz 2 will take place on next Tuesday
  - *boolean expression* and *selection statements* (Lecture 7)
  - Implicitly, there will be function calls — so review functions again