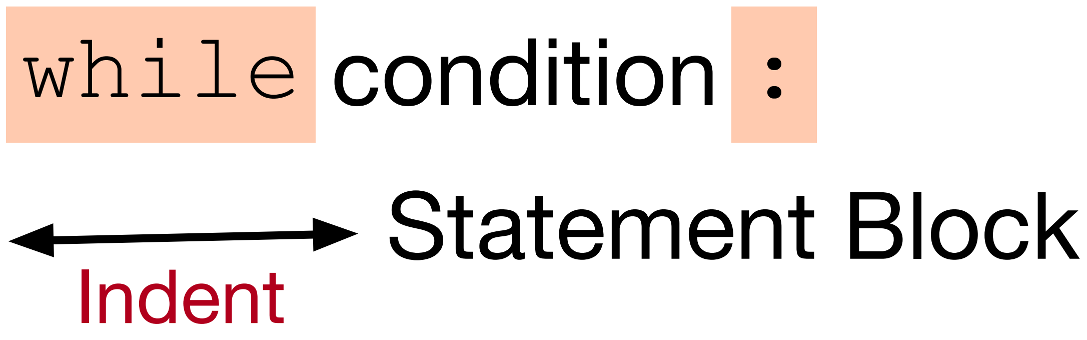
While Loops

Python

While Loops

- Form of the while loop:



`while` condition `:`

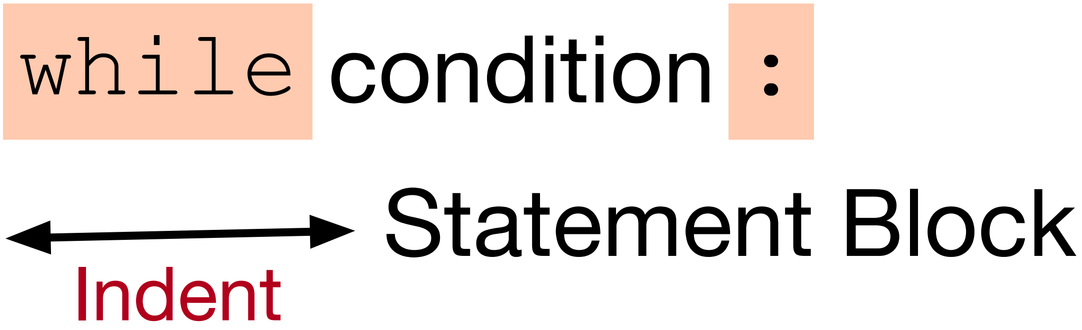
    Statement Block

- Keyword is while
- Condition needs to evaluate to either True or False
    - Condition is a <u>boolean</u>

While Loop Conditions

- Statement block is executed as long as condition is valid.
    - Allows the possibility of infinite loops

Apple Inc.
One Infinite Loop
Cupertino, CA 95014
(408) 606-5775



`while` condition `:`

Statement Block

An Infinite Loop

```
while True:
  print(“Hello World”)
```

If this happens to you, you might have to kill Idle process.

While Loops can emulate for loops

- Find an equivalent while loop for the following for-loop
    - (which calculates $\sum_{\nu=1}^{n} \frac{1}{\nu}$)

```
n = int(input("Enter n: "))
suma = 0
for i in range(1,n+1):
    suma += 1/i
print("The", n, "th harmonic number is", sum)
```

While loops can emulate for loops

- Solution: the loop-variable *i* has to start out as 1 and then needs to be incremented for every loop iteration
- We stop the loop when *i* reaches *n*+1, i.e. we continue as long as *i* <= *n.*

```
n = int(input("Enter n: "))
sum = 0
i = 1
while i<= n:
    sum += 1/i
    i += 1
print("The", n, "th harmonic number is", sum)
```

Harmonic Numbers

- The *n*th harmonic number is $$h_n = \sum_{\nu=1}^{n} \frac{1}{\nu}$$
  - It is known that this series diverges.
- Given a positive number *x*, we want to determine *n* such that the *n*th harmonic number is just above x $$\min(\{n \,|\, h_n > x\})$$
- Solution: add $\frac{1}{\nu}$ while you have not reached *x*

Harmonic Numbers

```
x = float(input("Enter x: "))
nu = 1
sum = 0
while sum <= x:
    sum += 1/nu
    nu += 1
print("The number you are looking for is ", nu-1,
      "and incidentally, h_n =“, sum)
```

- When we stop, we need to undo the last increment of nu, but not for sum.

Breaking out of a while loop

- You break out of a while loop, if the condition in the while loop is False
- Or by using a statement
  - `break` breaks out of the current loop
  - Can be used in for loops as well
- A related statement is the continue statement
  - `continue` breaks out of the current iteration of the loop and goes to the next
- We'll learn them in the course of the classes.

Example

- Find a number that fulfills the following congruences

$$\begin{aligned} x &\equiv 2 \pmod{3} \\ x &\equiv 3 \pmod{5} \\ x &\equiv 2 \pmod{7} \end{aligned}$$

  - This is Sun-Tsu's problem and the Chinese Remaindering Theorem in Mathematics helps with solving these problems.

Example

- We try out all numbers between 1 and $3 \times 5 \times 7$
  - We check each number whether they fulfill the congruences
  - If we find one, we print it out and break out of the while loop.

```
x = 1
while x < 3*5*7:
    if x%3==2 and x%5==3 and x%7==2:
        print(x)
        break
    x += 1
```