IST 1025

Introduction to Programming

Iteration with the **`for`** Loop

Control Operations

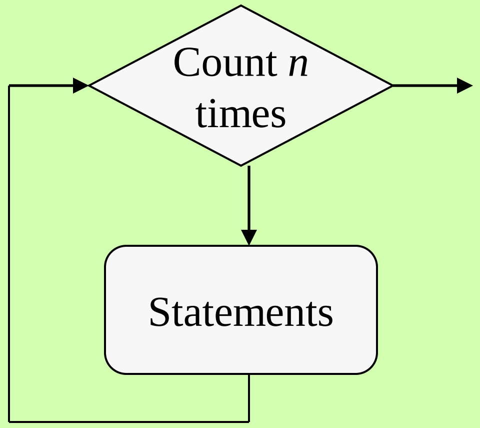
- *Basic operations* are input, output, arithmetic, etc.

- *Control operations* allow for sequencing other operations
  - Choose between several alternative sequences (selection)
  - Repeat a single sequence (iteration)

Doing the Same Thing Many Times

- It’s possible to do something repeatedly by just writing it all out

- Print ‘hello’ 5 times



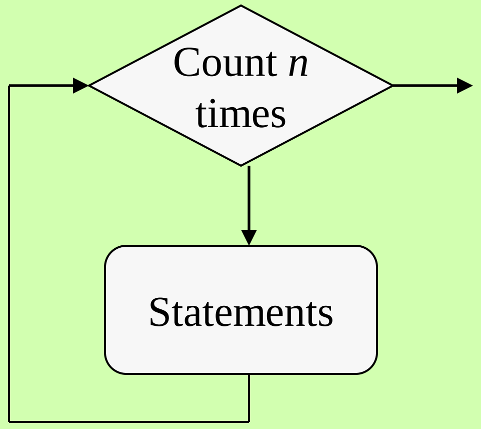
```
print('Hello!')
print('Hello!')
print('Hello!')
print('Hello!')
print('Hello!')
```

```
Hello
Hello
Hello
Hello
Hello
```

Iteration and Loops

- A *loop* repeats a sequence of statements

- A *definite loop* repeats a sequence of statements a predictable number of times



```
for count in range(5): print('Hello!')
```

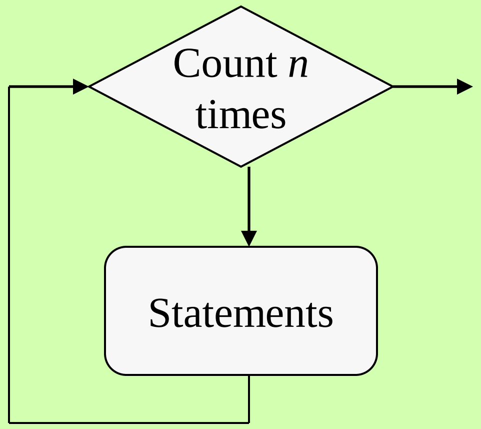
```
Hello
Hello
Hello
Hello
Hello
```

The `for` Loop

Python's **`for`** loop can be used to iterate a definite number of times

```
for <variable> in range(<number of times>): <statement>
```

Use this syntax when you have only one statement to repeat



```
for count in range(5): print('Hello!')
```

```
Hello
Hello
Hello
Hello
Hello
```

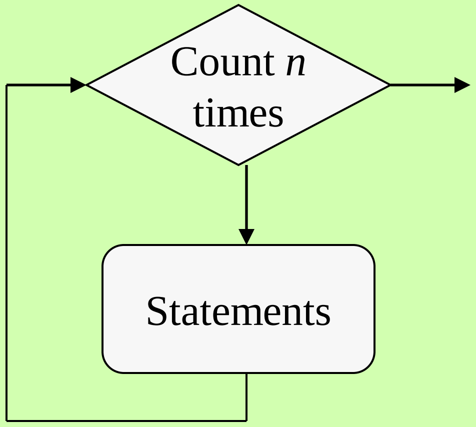
The `for` Loop

```
for <variable> in range(<number of times>):
    <statement-1>
    <statement-2>
    …
    <statement-n>
```

loop header

loop body

Use *indentation* to format two or more statements below the *loop header*



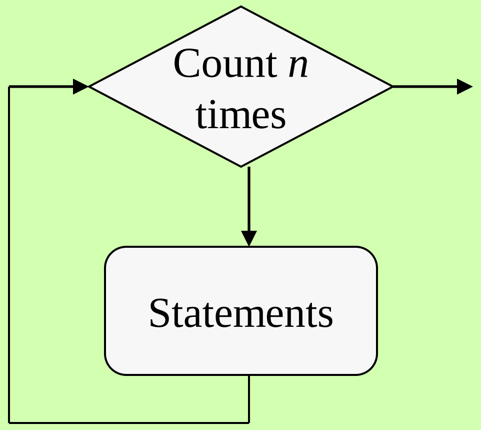
```
for count in range(3):
    print('Hello!')
    print('goodbye')
```

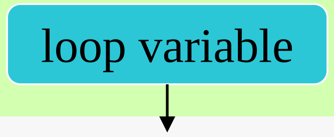
```
Hello!
goodbye
Hello!
goodbye
Hello!
goodbye
```

Using the Loop Variable

The *loop variable* picks up the next value in a *sequence* on each pass through the loop

The expression **`range(n)`** generates a sequence of **`int`**s from **`0`** through **`n - 1`**

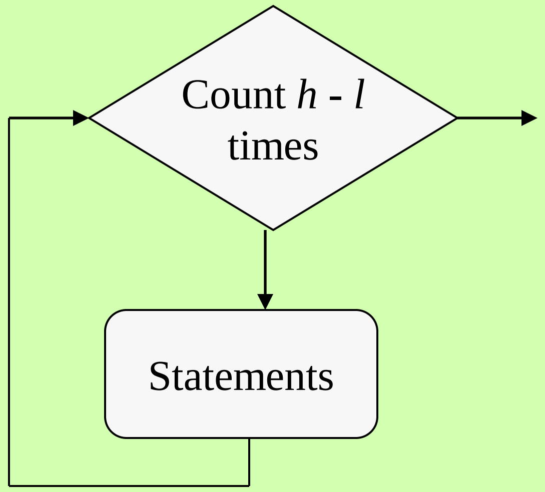




```
>>> for count in range(5):
print(count)
...
0
1
2
3
4
>>> list(range(5)) # Show as a list
[0, 1, 2, 3, 4]
```

Counting from 1 through *n*

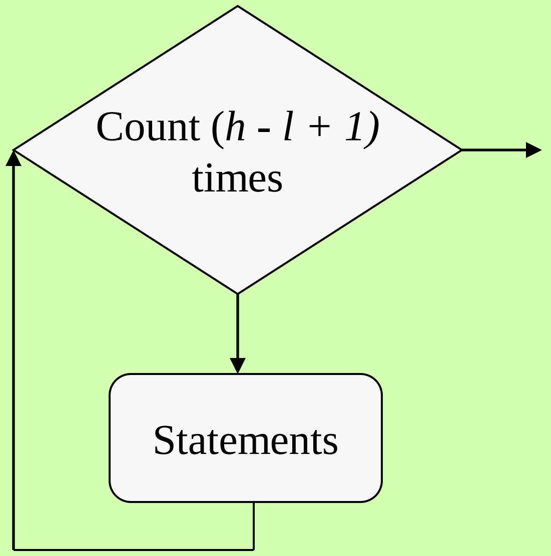
The expression **`range(low, high)`** generates a sequence of **`int`**s from **`low`** through **`high - 1`**



```
>>> for count in range(1, 6): print(count)
...
1
2
3
4
5
>>> list(range(1, 6))   # Show as a list
[1, 2, 3, 4, 5]
```

Skipping Steps in a Sequence

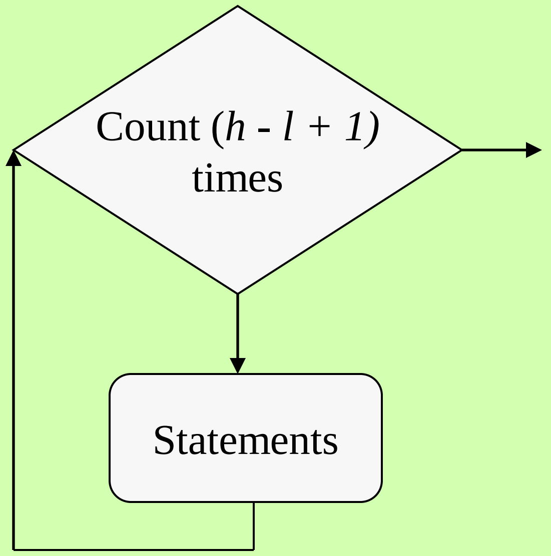
The expression **`range(low, high, step)`** generates a sequence of **`int`**s starting with **`low`** and counting up by **`step`** until **`high - 1`** is reached or exceeded



```
>>> for count in range(1, 6, 2):
        print(count)
...
1
3
5
>>> list(range(1, 6, 2)) # Show as a list
[1, 3, 5]
```

Counting down in a Sequence

The expression **`range(high, low, step)`** generates a sequence of **`int`**s starting with **`high`** and counting down by **`step`** until **`low - 1`** is reached, when **`step`** is negative



```
>>> for count in range(4, 1, -1):
        print(count)
...
4
3
2
>>> list(range(4, 1, -1)) # Show as a list
[4, 3, 2]
```

Accumulator Loop: Summation

Compute and print the sum of the numbers between 1 and 5, inclusive

```
total = 0
for n in range(1, 6):
    total = total + n
print(total)
```

In this program, the variable **`total`** is called an *accumulator variable*

Extended Assignment

Compute and print the sum of the numbers between 1 and 5, inclusive

```
total = 0
for n in range(1, 6):
    total += n
print(total)
```

The expression

<variable> += <expression>

is shorthand for the expression

<variable> = <variable> + <expression>

Accumulator Loop: Product

Compute and print the product of the numbers between 1 and 5, inclusive

```
product = 1
for n in range(1, 6):
   product = product * n
print(product)
```

The loop *pattern* or *idiom* is the same as that of the sum loop

Vary the initial value of the accumulator variable and the means of increasing it

Using a Loop in a Real Problem

An investor deposits $10,000 with the Get-Rich-Quick agency and receives a statement predicting the earnings on an annual percentage rate (APR) of 6% for a period of 5 years. Write a program that prints the beginning principal and the interest earned for each year of the period. The program also prints the total amount earned and the final principal. The interest is compounded annually.

Design in Pseudocode

An investor deposits $10,000 with the Get-Rich-Quick agency and receives a statement predicting the earnings on an annual percentage rate (APR) of 6% for a period of 5 years. Write a program that prints the beginning principal and the interest earned for each year of the period. The program also prints the total amount earned and the final principal.

```
principal = 10000
rate = .06
term = 5
totalinterest = 0
for each year in term
   print principal
   interest = principal * rate
   print interest
   principal = principal + interest
   totalinterest = totalinterest + interest
print totalinterest
print principal
```

First Coding in Python

An investor deposits $10,000 with the Get-Rich-Quick agency and receives a statement predicting the earnings on an annual percentage rate (APR) of 5% for a period of 5 years. Write a program that prints the beginning principal and the interest earned for each year of the period. The program also prints the total amount earned and the final principal.

```python
principal = 10000
rate = .06
term = 5
totalinterest = 0
for year in range(term):
   print(principal)
   interest = principal * rate
   print(interest)
   principal = principal + interest
   totalinterest = totalinterest + interest
print(totalinterest)
print(principal)
```

Refinement: Clean Up Output

An investor deposits $10,000 with the Get-Rich-Quick agency and receives a statement predicting the earnings on an annual percentage rate (APR) of 5% for a period of 5 years. Write a program that prints the beginning principal and the interest earned for each year of the period. The program also prints the total amount earned and the final principal.

```
principal = 10000
rate = .06
term = 5
totalinterest = 0
for year in range(term):
    interest = principal * rate
    print(principal, interest)
    principal = principal + interest
    totalinterest = totalinterest + interest
print('Total interest:', totalinterest)
print('Total principal:', principal)
```

Print all data on each line

Label outputs

Second Refinement: $d.cc

An investor deposits $10,000 with the Get-Rich-Quick agency and receives a statement predicting the earnings on an annual percentage rate (APR) of 5% for a period of 5 years. Write a program that prints the beginning principal and the interest earned for each year of the period. The program also prints the total amount earned and the final principal.

```
principal = 10000
rate = .06
term = 5
totalinterest = 0
for year in range(term):
   interest = principal * rate
   print(round(principal, 2), round(interest, 2))
   principal = principal + interest
   totalinterest = totalinterest + interest
print('Total interest:', round(totalinterest, 2))
print('Total principal:', round(principal, 2))
```

Round to nearest 100th before each output

Third Refinement: Print the Year

An investor deposits $10,000 with the Get-Rich-Quick agency and receives a statement predicting the earnings on an annual percentage rate (APR) of 5% for a period of 5 years. Write a program that prints the beginning principal and the interest earned for each year of the period. The program also prints the total amount earned and the final principal.

```
principal = 10000
rate = .06
term = 5
totalinterest = 0
for year in range(1, term + 1):
   interest = principal * rate
   print(year, round(principal, 2), round(interest, 2))
   principal = principal + interest
   totalinterest = totalinterest + interest
print('Total interest:', round(totalinterest, 2))
print('Total principal:', round(principal, 2))
```

Generalize to Solve for *Any* Principal, Rate, and Term

An investor deposits a given amount with the Get-Rich-Quick agency and receives a statement predicting the earnings on an annual percentage rate (APR) for a period of years. Write a program that prints the beginning principal and the interest earned for each year of the period. The program also prints the total amount earned and the final principal.

```
principal = ?
rate = ?
term = ?
totalinterest = 0
for year in range(1, term + 1):
    interest = principal * rate
    print(year, round(principal, 2), round(interest, 2))
    principal = principal + interest
    totalinterest = totalinterest + interest
print('Total interest:', round(totalinterest, 2))
print('Total principal:', round(principal, 2))
```