

Python Commands and Operations

- The following slides come from:

Marty Stepp (stepp@cs.washington.edu)

Lecturer, Computer Science & Engineering

University of Washington

The for loop

- **for loop**: Repeats a set of statements over a group of values.

- Syntax:

```
for variableName in groupOfValues:  
    statements
```

- We indent the statements to be repeated with tabs or spaces.
- **variableName** gives a name to each value, so you can refer to it in the **statements**.
- **groupOfValues** can be a range of integers, specified with the `range` function.

- Example:

```
for x in range(1, 6):  
    print (x, "squared is", x * x)
```

Output:

```
1 squared is 1  
2 squared is 4  
3 squared is 9  
4 squared is 16  
5 squared is 25
```

range

- The `range` function specifies a range of integers:
 - `range(start, stop)` - the integers between **start** (inclusive) and **stop** (exclusive)
 - It can also accept a third value specifying the change between values.
 - `range(start, stop, step)` - the integers between **start** (inclusive) and **stop** (exclusive) by **step**

- Example:

```
for x in range(5, 0, -1):  
    print x  
print ("Blastoff!")
```

Output:

```
5  
4  
3  
2  
1  
Blastoff!
```

Cumulative loops

- Some loops incrementally compute a value that is initialized outside the loop. This is sometimes called a *cumulative sum*.

```
sum = 0
for i in range(1, 11):
    sum = sum + (i * i)
print ("sum of first 10 squares is", sum)
```

Output:

```
sum of first 10 squares is 385
```

if

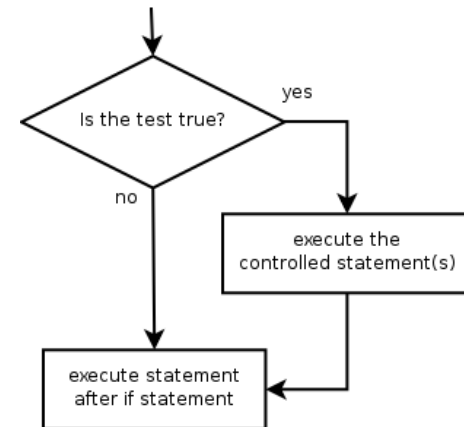
- **if statement:** Executes a group of statements only if a certain condition is true. Otherwise, the statements are skipped.

- Syntax:

```
if condition:  
    statements
```

- Example:

```
gpa = 3.4  
if gpa > 2.0:  
    print ("Your application is accepted.")
```



if/else

- **if/else statement:** Executes one block of statements if a certain condition is True, and a second block of statements if it is False.

- Syntax:

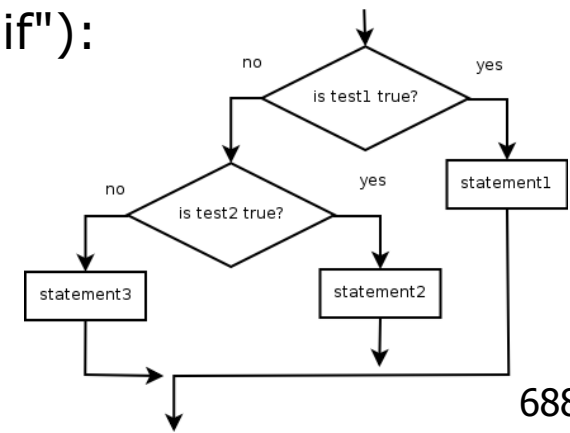
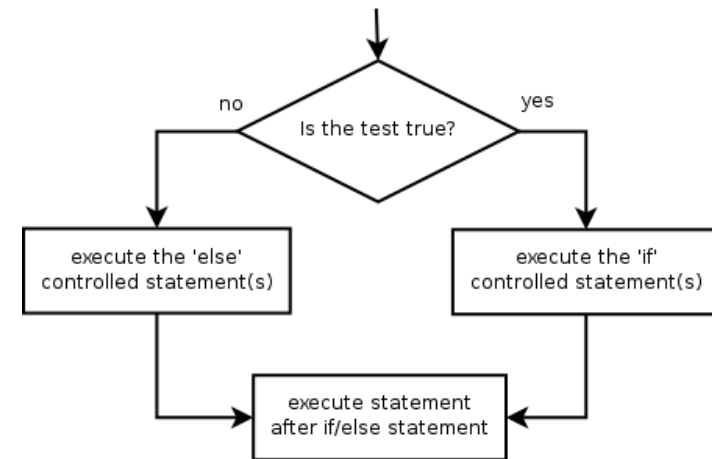
```
if condition:  
    statements  
else:  
    statements
```

- Example:

```
gpa = 1.4  
if gpa > 2.0:  
    print ("Welcome to Mars University!")  
else:  
    print ("Your application is denied.")
```

- Multiple conditions can be chained with `elif` ("else if"):

```
if condition:  
    statements  
elif condition:  
    statements  
else:  
    statements
```



while

- **while loop:** Executes a group of statements as long as a condition is True.
 - good for *indefinite loops* (repeat an unknown number of times)

- **Syntax:**

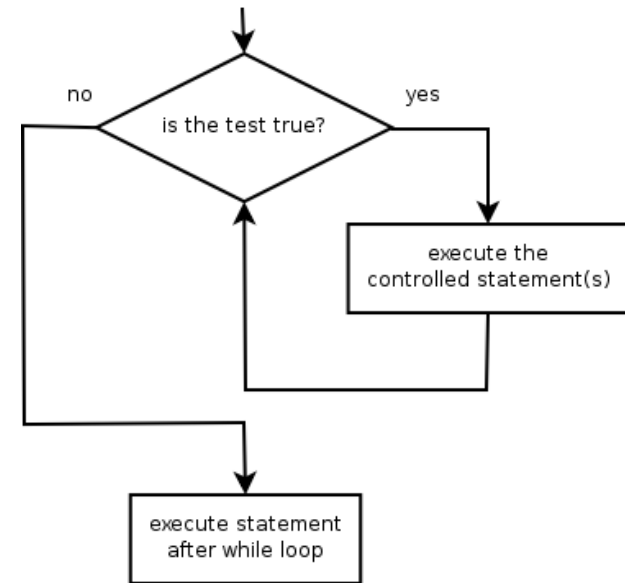
```
while condition:  
    statements
```

- **Example:**

```
number = 1  
while number < 200:  
    print (number)  
    number = number * 2
```

- **Output:**

1 2 4 8 16 32 64 128



Logic

- Many logical expressions use *relational operators*:

Operator	Meaning	Example	Result
==	equals	<code>1 + 1 == 2</code>	True
!=	does not equal	<code>3.2 != 2.5</code>	True
<	less than	<code>10 < 5</code>	False
>	greater than	<code>10 > 5</code>	True
<=	less than or equal to	<code>126 <= 100</code>	False
>=	greater than or equal to	<code>5.0 >= 5.0</code>	True

- Logical expressions can be combined with *logical operators*:

Operator	Example	Result
and	<code>9 != 6 and 2 < 3</code>	True
or	<code>2 == 3 or -1 < 5</code>	True
not	<code>not 7 > 0</code>	False