Introduction to High Performance Scientific Computing

Autumn, 2017

Python lecture 2

Imperial College
London

Other variable types

- *dictionaries* and *classes* are also important variable types
- We will (probably) not cover them in this class
- See supplementary material section on course webpage

Loops and if statements

- Loops and if statements are building blocks of most codes
- Python requires colon, “:” to indicate **start** of block
- Indentation sets the **size** of the block

```
if < Boolean expression 1>:
        <block 1>
# code outside of block
```

Loops and if statements

- **Loops and if statements are building blocks of most codes**
- **Python requires colon, “:” to indicate start of block**
- **Indentation sets the size of the block**

```
if < Boolean expression 1>:
        <block 1>
elif < Boolean expression 2> :
        <block 2>
else:
        <block 3>
```

Loops and if statements

- **Loops and if statements are building blocks of most codes**
- **Python requires colon, “:” to indicate start of block**
- **Indentation sets the size of the block**

```
if < Boolean expression 1>:
        <block 1>
elif < Boolean expression 2> :
        <block 2>
else:
        <block 3>
```

```
x=rand(1)[0]
if x<0.5:
    print(“left: x=%s" %(x))
elif x>=0.5:
    print("right: x=%s” %(x))
else:
    print "error, x is not numeric”

right: x=[ 0.97813575]
```

if block

Code generates random variable between zero and one and determines if it is greater than or less than 0.5

while loops

- **Structure of while loops is similar to if statements**

```
while < Boolean expression 1>:
        <block 1>
```

Imperial College
London

While loops

- **Structure of while loops is similar to if statements**

```
while < Boolean expression 1>:
        <block 1>
```

```
x = rand(1)
while x<1:
    print("x=%s" %(x))
    x=x+0.1

x=[ 0.17099121]
x=[ 0.27099121]
x=[ 0.37099121]
x=[ 0.47099121]
x=[ 0.57099121]
x=[ 0.67099121]
x=[ 0.77099121]
x=[ 0.87099121]
x=[ 0.97099121]
```

Generate random variable, x, and add increments of 0.1 until x>1

Imperial College
London

for loops

- **for loops iterate through items in a list:**

```
for x in list:
        <block>
```

Imperial College
London

for loops

- **for loops iterate through items in a container:**

```
for x in list:
        <block>
```

```
In [37]: y=0

In [38]: for x in [1,2,3]:
        ....:     y = y + x #can also write y += x
        ....:     print("y = %s" %(y))
        ....:
y = 1
y = 3
y = 6
```

for loops

- **for loops iterate through items in a container:**

```
for x in list:
        <block>
```

```
In [37]: y=0

In [38]: for x in [1,2,3]:
       ....:     y = y + x #can also write y += x
       ....:     print("y = %s" %(y))
       ....:
y = 1
y = 3
y = 6
```

- **Can also iterate through:**
  - **Items in a tuple**
  - **Characters in a string**

for loops

range **function is useful for generating lists:**

```
In [23]: range(4)
Out[23]: [0, 1, 2, 3]

In [24]: range(2,6)
Out[24]: [2, 3, 4, 5]

In [25]: range(2,6,2)
Out[25]: [2, 4]
```

for loops

range **function is useful for generating lists:**

```
In [23]: range(4)
Out[23]: [0, 1, 2, 3]

In [24]: range(2,6)
Out[24]: [2, 3, 4, 5]

In [25]: range(2,6,2)
Out[25]: [2, 4]
```

```
In [45]: L = ["a","few","words"]

In [46]: for i in range(3):
    ....:     print(i,L[i])
    ....:
0 a
1 few
2 words
```

for loops

range **function is useful for generating lists:**

```
In [23]: range(4)
Out[23]: [0, 1, 2, 3]

In [24]: range(2,6)
Out[24]: [2, 3, 4, 5]

In [25]: range(2,6,2)
Out[25]: [2, 4]
```

```
In [45]: L = ["a","few","words"]

In [46]: for i,j in enumerate(L):
   ....:     print(i,j)
   ....:
0 a
1 few
2 words
```

- ***enumerate*** **function can also be used for this example**
- **(also see *zip*)**

Block controls: *break* and *continue*

continue **allows you to skip remaining steps in block**

```
In [65]: words = ["yes","yes","no","yes"]

In [66]: for w in words:
    ....:     if w == "yes":
    ....:         continue
    ....:     print(w)
    ....:
    no
```

print statement is only executed if w is not equal to “yes”

Block controls: *break* and *continue*

break **statement allows premature ending of loop:**

```
In [54]: words = ["yes","yes","no","yes"]

In [55]: for w in words:
     ....:     if w == "no":
     ....:         print("breaking for loop”)
     ....:         break
     ....:     else:
     ....:         print(w)
     ....:
yes
yes
breaking for loop
```