



Python Programming

COMP 8347

Usama Mir

Usama.Mir@uwindsor.ca

Python Basics

- ▶ Topics:
 - ▶ If statements
 - ▶ Loops
 - ▶ Exception handling
 - ▶ Functions
 - ▶ File Input/Output
 - ▶ Modules

Control Flow

- ▶ Conditional branching
 - ▶ if statements
- ▶ Looping
 - ▶ while
 - ▶ for ... in
- ▶ Exception handling
- ▶ Function or method call

if Statements

- ▶ **suite**: a block of code, i.e. a sequence of one or more statements

- ▶ Syntax:

```
if bool_expression1:
    suite1
elif bool_expression2:
    suite2
...
elif bool_expressionN:
    suiteN
else:
    else_suite
```

- No parenthesis or braces
- Use indentation for block structure

```
if a < 10:
    print("few")
elif a < 25:
    print("some")
else:
    print("many")
```

Loops = What is it?

- ▶ Clear starting condition
- ▶ Clear finishing condition
- ▶ Some statement that leads the loops from its start to the end

Loops: Python vs. Java

- ▶ Python:
 - ▶ while loops
 - ▶ for loops `for i in range (5)`
- ▶ Java:
 - ▶ while loops
 - ▶ do .. while
 - ▶ for loops (`int i = 0....`)

while Statements

- ▶ Used to execute a suite 0 or more times
 - ▶ number of times depends on while loop's Boolean expression.
 - ▶ Syntax:

```
while bool_expression:  
    suite
```

- ▶ Example:

```
x, sum = 0,0
```

```
while x < 10:
```

```
    sum += x
```

```
    x += 2
```

```
print(sum, x) #What is final value of sum and x
```

Answer: sum=**20**, x=**10**



for ... in Statements

- ▶ Syntax:

for variable in iterable:
suite

- ▶ Example: `fruits = ['apple', 'pear', 'plum', 'peach']`

`for item in fruits:`

`print(item)`

- ▶ Alternatively

`for i in range(len(fruits)):`

`print(fruits[i])`

Break and Continue

```
ex2.py - C:\Users\Arunita\OneDrive - University of Windsor\8347\slidesF20\ex2.py (3.8.3)
File Edit Format Run Options Window Help
1 # Example using break and continue
2 # Continue: control returns to top of current loop
3 # Break: exit from current loop
4
5 for num in [11, 8, 3, 25, 9, 16]:
6     if num > 20:
7         print('exiting loop')
8         break # exit the loop completely
9     elif num%2 == 0:
10        continue # immediately start next iteration
11    print(num)
12
```

Ln: 7 Col: 28

Iter#	num	num>20?	Num%2==0?	print(num)
0	11	n	n	11
1	8	n	y	
2	3	n	n	3
3	25	y		

Exception Handling

- ▶ Functions or methods indicate errors or other important events by ***raising exceptions***.
- ▶ Syntax (simplified):

```
try:  
    try_suite  
except exception1 as variable1:  
    exception_suite1  
  
...  
except exceptionN as variableN  
finally:  
    # cleanup
```
- ▶ variable part is optional


Exception Handling - Example 1

Example:

```
s = input('Enter number: ')
try:
    n = float(s)
    print(n, ' is valid. ')
except ValueError as err:
    print(err)
```


- If user enters '8.6' output is: 8.6 is valid
- If user enters 'abc', output is:
ValueError: could not convert string to float:
'abc'

Exception Handling - Example 2

 exceptionhandling.py - C:/Users/manch/OneDrive/Desktop/U of Windsor/Comp 8347/New/All slides/Usama Slides/2/exceptionhandling.py (3.10.1)

File Edit Format Run Options Window Help

```
mylist = [5, 10, 15, 20, 9.9]
for i in range (5):
    item = mylist[i**2]
    print("index=", i**2, "item", item)
```

 IDLE Shell 3.10.1

File Edit Shell Debug Options Window Help

```
Python 3.10.1 (tags/v3.10.1:2cd268a, Dec 6 2021, 19:10:37) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/manch/OneDrive/Desktop/U of Windsor/Comp 8347/New/All slides/Usama Slides/2/exceptionhandling.py
index= 0 item 5
index= 1 item 10
index= 4 item 9.9
Traceback (most recent call last):
  File "C:/Users/manch/OneDrive/Desktop/U of Windsor/Comp 8347/New/All slides/Usama Slides/2/exceptionhandling.py", line 3, in <module>
    item = mylist[i**2]
IndexError: list index out of range
>>>
```

Exception Handling - Example 2

 exceptionhandling.py - C:/Users/manch/OneDrive/Desktop/U of Windsor/Comp 8347/New/All slides/Usama Slides/2/exceptionhandling.py (3.10.1)

File Edit Format Run Options Window Help

```
mylist = [5, 10, 15, 20, 9.9]
```

```
for i in range (5):
```

```
    try:
```

```
        item = mylist[i**2]
```

```
        print("index=", i**2, "item", item)
```

```
    except IndexError as err:
```

```
        print(err)
```

```
        print(i**2, "is out of range")
```

```
>>>
```


```
=====
index= 0 item 5
index= 1 item 10
index= 4 item 9.9
list index out of range
9 is out of range
list index out of range
16 is out of range
```

Functions

► Function definition:

A block of reusable code that is used to perform a single action

Ex. A code without a function →

 function example.py - C:\Users\manch\OneDrive\Desktop\function example.py (3.10.1)

File Edit Format Run Options Window Help

```
#other code here
```

```
name = input("enter your name:")  
time = input("enter the preferred time")  
print("Good" + time + "," + name)
```

```
#other code here
```

```
name = input("enter your name:")  
time = input("enter the preferred time")  
print("Good" + time + "," + name)
```

Functions

- Same code implemented with a function

```
#function starts here
```

```
def greet ():  
    name = input("enter your name:")  
    time = input("enter the preferred time")  
    print("Good" + time + "," + name)
```

```
greet()  
#other code here
```

```
greet()  
#other code here
```



File Input/Output

- ▶ Open a file
 - ▶ `f = open(filename, mode)`
 - ▶ mode is optional; possible values:
 - ▶ 'w' = write
 - ▶ 'r' = read (default)
 - ▶ 'a' = append
 - ▶ 'rb'('wb') = read (write) in binary
 - ▶ Example:
 - ▶ `f = open("text.txt")`

File Methods

- ▶ Read data from a file
`print(f.read())`
- ▶ Writing data on a file
 - ▶ `f = open("test.txt", 'w')`
 - ▶ `f.write("Hello Python")`
- ▶ `f.read(n)`: reads at most `n` bytes from `f`
- ▶ `f.readline()`: reads only one line
- ▶ `f.readlines()`: reads all the lines to the end of file and return them as a list
- ▶ `f.close()`: closes a file and free up the resources

Modules

- ▶ **Modules**: Contain additional functions and custom data types.
 - ▶ Ex. import calendar
- ▶ Other examples:
 - ▶ import os
 - ▶ import math
 - ▶ import datetime

References

- ▶ Slides from Dr. Arunita and Dr. Saja
- ▶ <https://www.softwaretestinghelp.com/python/input-output-python-files/>
- ▶ <https://www.tutorialspoint.com/difference-between-for-loop-and-while-loop-in-python>
- ▶ Programming in Python 3 A complete introduction to the python language (2nd Ed) by Mark Summerfield. Addison Wesley 2010.
- ▶ <https://www.w3schools.com/python/>
- ▶ <https://www.tutorialsteacher.com/python/error-types-in-python>