SWINBURNE UNIVERSITY OF TECHNOLOGY SARAWAK CAMPUS

Lecture 7

User Defined Functions

KNOWING

User Defined Function

- A user-defined function's declaration begins with the keyword def and followed by the function name.
- The function may take argument(s) as input within the opening and closing parentheses, just after the function name followed by a color.

User Defined Function

- After defining the function name and argument(s) a block of program statement(s) start at the next line and these statement(s) must be indented.

User Defined Function

- Syntax:

```
Def function_name(argument1, argument2, …):
        statement_1
        statement_2
        …
```

Call a function

- Calling a function in Python is similar to other programming languages, using the function name, parentheses (opening and closing) and parameter(s).
- Syntax:

```
function_name(arg1, arg2)
```

Call a function

- Example:

File Edit Format Run Options Window Help

```
def avg_number(x,y):
    average = (x + y)/2
    print("Average of ", x, " and ", y, " is ", average)

avg_number(13,14)
```

Python 3.5.3 Shell

File Edit Shell Debug Options Window Help

```
Python 3.5.3 (v3.5.3:1880cb95a742, Jan 16 2017, 15:51:26) [MSC v.1900 32 bit (In
tel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:/Users/HP/AppData/Local/Programs/Python/Python35-32/Image.py ==
Average of  13  and  14  is  13.5
>>>
```

Ln: 6 Col: 4

Function without arguments

- Syntax:

```
def function_name():
        statement_1
        statement_2
        …
```

Function without arguments

- Example:

File Edit Format Run Options Window Help

```
def printmessage():
    print("This is Lecture 6")
    print("Topic: User Defined Function")
    print("Python Programming")

printmessage()
```

Python 3.5.3 Shell

File Edit Shell Debug Options Window Help

```
Python 3.5.3 (v3.5.3:1880cb95a742, Jan 16 2017, 15:51:26) [MSC v.1900 32 bit (In
tel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:/Users/HP/AppData/Local/Programs/Python/Python35-32/Image.py ==
This is Lecture 6
Topic: User Defined Function
Python Programming
>>>
```

Ln: 8 Col: 4

Return statement in function

- The return statement (the word return followed by an expression) is used to return a value from a function, return statement without an expression argument returns none.

Return statement in function

- Syntax:

```
def function_name(argument1, argument2, …):
        statement_1
        statement_2
        …
        return expression
function_name(arg1, arg2)
```

Return statement in function

- Example:

```
def nsquare(x,y):
    ans = (x * x+ 2 * x * y + y * y)
    return ans
print("The square of the sum of 2 and 3 is: ", nsquare(2,3))
```

Python 3.5.3 Shell

```
Python 3.5.3 (v3.5.3:1880cb95a742, Jan 16 2017, 15:51:26) [MSC v.1900 32 bit (In
tel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:/Users/HP/AppData/Local/Programs/Python/Python35-32/Image.py ==
The square of the sum of 2 and 3 is:  25
>>>
```

Ln: 6 Col: 4

Default Argument Values

- In function's parameters list we can specify a default value(s) for one or more arguments.
- A default value can be written in the format "argument1 = value", therefore we will have the option to declare or not declare a value for those arguments.

KNOW ING

Default Argument Values

- Example:

```
def nsquare(x,y=3):
    ans = (x * x+ 2 * x * y + y * y)
    return ans
print("The square of the sum of 2 and 3 is: ", nsquare(2))
print("The square of the sum of 2 and 3 is: ", nsquare(2,4))
```

Python 3.5.3 Shell

```
Python 3.5.3 (v3.5.3:1880cb95a742, Jan 16 2017, 15:51:26) [MSC v.1900 32 bit (In
tel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:/Users/HP/AppData/Local/Programs/Python/Python35-32/Image.py ==
The square of the sum of 2 and 3 is:  25
The square of the sum of 2 and 3 is:  36
>>>
```

Ln: 7 Col: 4

Keyword Arguments

- Functions can also be called using keyword arguments.
- Arguments which are preceded with a variable name followed by a ‘=’ sign (e.g. var_name=“”) are called keyword arguments.

Keyword Arguments

- All keyword arguments passed must match one of the arguments accepted by the function.
- You may change the order of appearance of the keyword

KNOW ING

Keyword Arguments

- Example:

```
def stud_marks(IT, programming = 65, maths = 60):
    print("Marks in: IT: ", IT, "Programming:", programming, "Mathematics:", maths)
stud_marks(80,68) #re-assign the marks for IT and programming
stud_marks(79, maths = 68) #re-assign the marks for IT and maths
stud_marks(80,68,70) #re-assign the marks for all subjects
```

Python 3.5.3 Shell

```
Python 3.5.3 (v3.5.3:1880cb95a742, Jan 16 2017, 15:51:26) [MSC v.1900 32 bit (Intel)] on win3
2
Type "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:/Users/HP/AppData/Local/Programs/Python/Python35-32/Image.py ==
Marks in: IT:  80 Programming: 68 Mathematics: 60
Marks in: IT:  79 Programming: 65 Mathematics: 68
Marks in: IT:  80 Programming: 68 Mathematics: 70
```

Arbitrary Argument Lists

- The arbitrary argument is an another way to pass arguments to a function.
- In the function body, these arguments will be wrapped in a tuple and it can be defined with *args constructs.
- Before this variable, you can define a number of arguments or no argument.

KNOW ING

Arbitrary Argument Lists

- Example:

File Edit Format Run Options Window Help

```
def sum(*numbers):
    x = 0
    for n in numbers:
        print("N in numbers:",x)
        x += n
    return x
print("Total", sum(1,2,3,4,5))
```

Python 3.5.3 Shell

File Edit Shell Debug Options Window Help

```
(Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:/Users/HP/AppData/Local/Programs/Python/Python35-32/Image.py =
=
N in numbers: 0
N in numbers: 1
N in numbers: 3
N in numbers: 6
N in numbers: 10
Total 15
>>>
```

Ln: 11 Col: 4

KNOW ING

Passed by value

- Parameters to functions are reference to objects, which are passed by value.
- When you pass a variable to a function, python passes the reference to the object to which the variable refers (the value). Not the variable itself.
- If the value is immutable, the function does not modify the caller's variable.

Passed by value

- If the value is immutable, the function does not modify the caller's variable.
- If the value is mutable, the function may modify the caller's variable in-place.

Passed by value

- Example:

File Edit Format Run Options Window Help

```
def number(x, y, z):
    x = 12
    y.append(50)
    z = [22] #new reference
    print("x", x)
    print("y", y)
    print("z", z)

a = 36     #immutable variable
b = [46]   #mutable variable
c = [56]

number(a,b,c)
print("a", a)
print("b", b)
print("c", c)
```

Python 3.5.3 Shell

File Edit Shell Debug Options Window Help

```
== RESTART: C:/Users/HP/AppData/Local/P
=
x 12
y [46, 50]
z [22]
a 36
b [46, 50]
c [56]
>>>
```

KNOW ING

Global Variables

- Variables declared outside the function can be reference within the function.
- Global variables cannot be modified within the function, unless declared global in the function.

Global Variables

- Example:

File Edit Format Run Options Window Help

```
x = 567

def number():
    print(x)
    z = 5 #local variable ar number() function
    print(z)

number()

print(z)
```

Python 3.5.3 Shell

File Edit Shell Debug Options Window Help

```
=
567
5
Traceback (most recent call last):
  File "C:/Users/HP/AppData/Local/Programs/Python/Python35-32/Image.py", lin
e 10, in <module>
    print(z)
NameError: name 'z' is not defined
>>>
```

Ln: 109 Col: 4

KNOW ING

Global Variables

- Example:

File Edit Format Run Options Window Help

```
x = 567

def number():
    print(x)
    global z #set to global variable
    z = 5
    print(z)

number()

print(z)
```

Python 3.5.3 Shell

File Edit Shell Debug Options Window Help

```
Python 3.5.3 (v3.5.3:1880cb95a742, Jan 16 2017, 15:51:26) [MSC v.
tel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:/Users/HP/AppData/Local/Programs/Python/Python35-32
567
5
5
>>>
```

KNOW ING

Global Variables

- Example:

File Edit Format Run Options Window Help

```
x = 8

def number():
    print(x)
    print(x+5)
    x += 6

number()
```

Python 3.5.3 Shell

File Edit Shell Debug Options Window Help

```
Python 3.5.3 (v3.5.3:1880cb95a742, Jan 16 2017, 15:51:26) [MSC v.1900 32 bit
(Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:/Users/HP/AppData/Local/Programs/Python/Python35-32/Image.py =
=
Traceback (most recent call last):
  File "C:/Users/HP/AppData/Local/Programs/Python/Python35-32/Image.py", lin
e 8, in <module>
    number()
  File "C:/Users/HP/AppData/Local/Programs/Python/Python35-32/Image.py", lin
e 4, in number
    print(x)
UnboundLocalError: local variable 'x' referenced before assignment
>>>
```

KNOW ING

Global Variables

- Example:

File Edit Format Run Options Window Help

```
x = 8

def number():
    global x
    print(x)
    print(x+5)
    x += 6

number()
```

Python 3.5.3 Shell

File Edit Shell Debug Options Window Help

```
t (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:/Users/HP/AppData/Local/Programs/Python/Python35-32/Image.py
==
8
13
>>>
```

Ln: 7 Col: 4

KNOW ING