Function Basics

Practice: tutorial Q1

Let's write code to calculate discounted prices for different items. Each item has a different discount rate.

Item	Original Price	Discount Rate
Item 1	120.00	15%
Item 2	59.90	10%
Item 3	249.50	20%
Item 4	19.99	5%

```
Item 1: Original price = $120.00, Discount = $18.00, Final price = $102.00
Item 2: Original price = $59.90, Discount = $5.99, Final price = $53.91
Item 3: Original price = $249.50, Discount = $49.90, Final price = $199.60
Item 4: Original price = $19.99, Discount = $1.00, Final price = $18.99
```

Reflection

- Do you notice anything repetitive in your code?
- What parts of your code are the same? What parts are different?
- If you made a mistake in the logic, how many places do you need to fix it?
- What if you had 100 items to calculate?

Abstraction in Different Aspects

Abstraction in Data: **Data Structures**

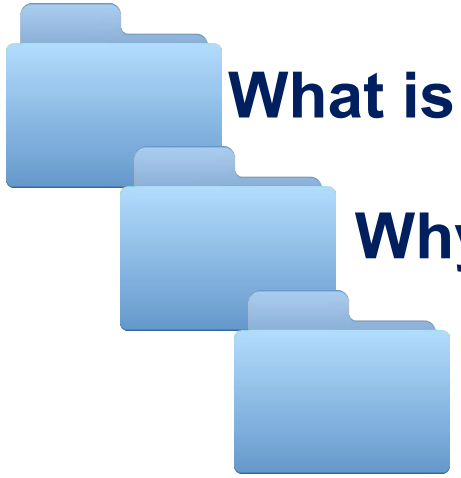
Programs = Algorithms + Data Structures

(1976, Niklaus Wirth)

Abstraction in Algorithms:

Functions

Topic Outline



What is a Function?

Why do we Need Functions?

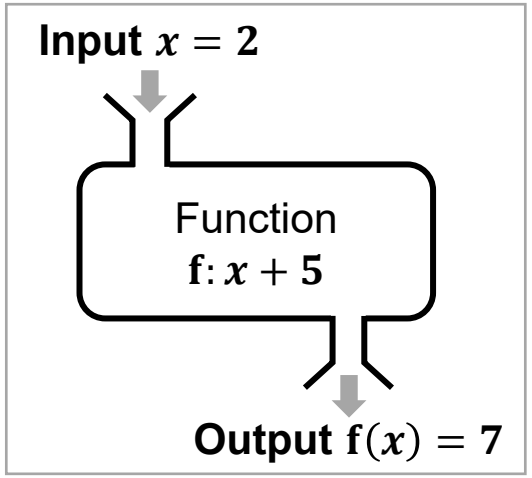
How to Define a Function in Python?

What is a Function?

FUNCTION

In Mathematics

performs some operation and returns **one** value/ thing

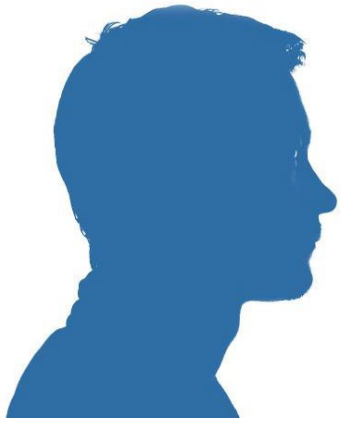


In Python

- represents a single operation to be performed
- takes zero or more arguments as input
- returns one value/ object as output

Python functions "**encapsulate**" the performance of its particular operation, so they can be used by others.

Importance of Functions



WHY Functions ?

- Abstraction
- Divide-and-conquer problem solving
- Reuse
- Sharing
- Security
- Simplification and Readability

Function Definition in Python

Function name must follow variable naming rules.

List of parameters being passed: in parentheses, comma-separated.

```
def functionName (parameter1, parameter2) :
```

Keyword indicating function is defined.

```
    statement1
    statement2

    return valueToReturn
```

tab

Return statement: indicates the value returned when the function finishes.

Function suite:

- contains code to perform some action
- indented

Suite of the function follows the **colon**.

Passing Arguments to Functions

- Argument: piece of data that is sent into a function
  - Function can use argument in calculations and processing
  - When calling the function, the argument is placed in parentheses following the function name

Figure 5-13 The `value` variable is passed as an argument

```
def main():
    value = 5
    show_double(value)
```

```
def show_double(number):
    result = number * 2
    print(result)
```

What is the output of the code in previous slide?

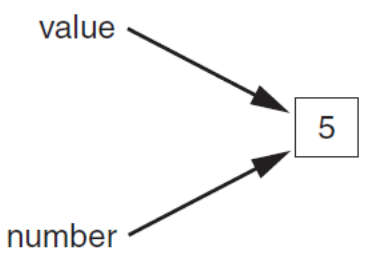
Parameter variable in Functions

- Parameter variable: variable that is assigned the value of an argument when the function is called
  - The parameter and the argument reference the same value
  - General format:
  - `def function_name(parameter):`
  - Scope of a parameter: the function in which the parameter is used

Figure 5-14 The `value` variable and the `number` parameter reference the same value

```
def main():
    value = 5
    show_double(value)


def show_double(number):
    result = number * 2
    print(result)
```



Passing Multiple Arguments

Python allows writing a function that accepts multiple arguments

- Parameter list replaces single parameter
  - Parameter list items separated by comma

Arguments are passed *by position* to corresponding parameters

- First parameter receives value of first argument, second parameter receives value of second argument, etc.

Passing Multiple Arguments (cont'd.)

Figure 5-16 Two arguments passed to two parameters

```
def main():
    print('The sum of 12 and 45 is')
    show_sum(12, 45)
```

```
def show_sum(num1, num2):
    result = num1 + num2
    print(result)
```

What is the output of the previous slide?

Default parameters in Python functions

- In Python, when defining a function, you can assign a **default value** to one or more parameters.
- If the caller does not provide a value for that parameter, Python will use the default.
- If the caller **does provide** a value, it will **override** the default.

What is the output of following code?

```python
def add(a, b=5):
    return a + b

print(add(3))
```

8

What will the following print?

```
def show(a=1, b=2, c=3):
    return a + b + c

print(show(5, c=10))
```

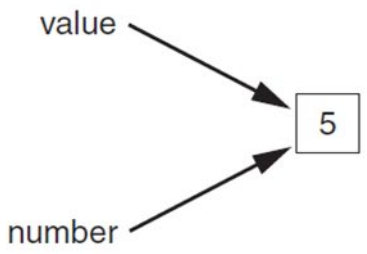
17

Return statement

Figure 5-14 The `value` variable and the `number` parameter reference the same value

```
def main():
    value = 5
    show_double(value)


def show_double(number):
    result = number * 2
    print(result)
```

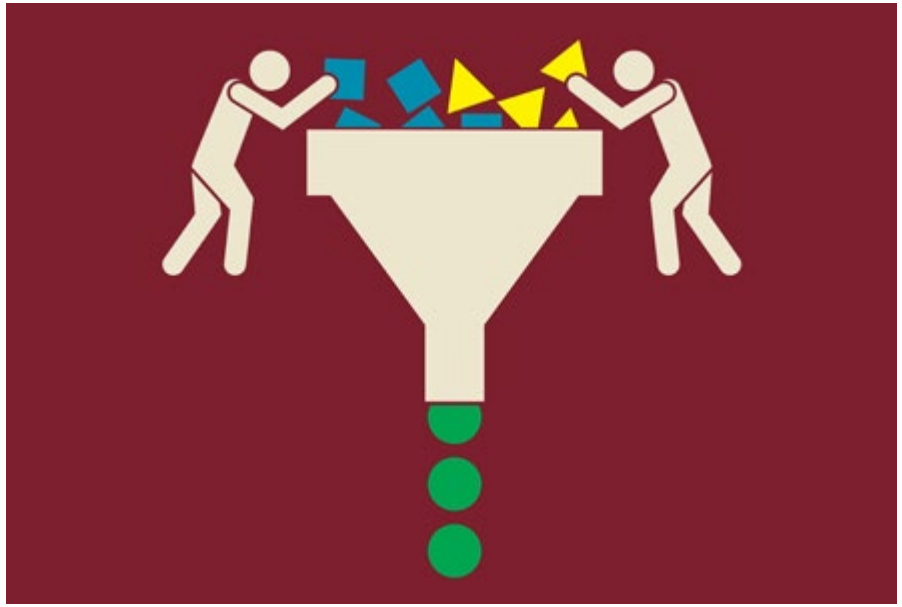



```
def show(a=1, b=2, c=3):
    return a + b + c

print(show(5, c=10))
```

`return` Statement

- The `return` statement indicates the value that is returned by the function.
- The statement is optional (the function can return nothing).
- If there is no return, the function is often called a procedure.



Procedures

- Functions **without return** statements are often called **procedures**.
- **Procedures** are used to perform some duty (print output, store a file, etc.).
- A **return** statement is not always required.

Function Definition in Python

Function name must follow variable naming rules.

List of parameters being passed: in parentheses, comma-separated.

```
def functionName (parameter1, parameter2) :
```

Keyword indicating function is defined.

```
    statement1
    statement2

    return valueToReturn
```

tab

Return statement: indicates the value returned when the function finishes.

Function suite:

- contains code to perform some action
- indented

Suite of the function follows the **colon**.

Dynamics of Function Calls

1. Function call copies argument C to parameter temp.
2. Control transfers to function “celsius2Fahrenheight”.
3. Expression in celsius2Fahrenheight is evaluated .
4. Value of expression is returned to invoker.

C = 10

F = celsius2Fahrenheit(C)

F = 50

temp = 10

def celsius2Fahrenheit(temp):

return temp * 1.8 + 3.20

return 10 * 1.8 + 32.0

return 50

Dynamics of Function Calls

Main Program

```
statement
                          25
fahrenheit = cel2fahr(25)
statement
statement
```

call

return

Function

```
def cel2fahr(celsius):
    77 val = celsius * 1.8 + 32
       return val
```

Multiple `return` Statements

- A function could have multiple **`return`** statements.
- The first executed return statement ends the function.

Caution

Multiple return statements might be confusing to the reader.

USE CAREFULLY!

slido

What is the output of the following code?

ⓘ Start presenting to display the poll results on this slide.

```
def test(a):
  if (a > 3):
    return 3
  elif (a > 2):
    return "fail"
  else:
    return(0,255,123)


tt = test(-81)
print(tt[2])
```

123

```python
def test(a):
  if (a > 3):
    return 3
  elif (a > 2):
    return "fail"
  else:
    return(0,255,255)

print(test(5))
print(test(2.3))
print(test(1))
```

```
3
fail
(0, 255, 255)
```

Q1: What is the output of the following Python program?

```
charList = ['a', 'e', 'i', 'o', 'u']
myStr = "This is a string!"

def funcA(content, target):
    num = 0

    for char in content:
        if char in target:
            num += 1

    return num


result = funcA(myStr, charList)
print(result)
```

A. 0
B. 1
C. 2
D. 3
E. 4
F. 5

What is the output of previous slide?

Q1: What is the output of the following Python program?

```
charList = ['a', 'e', 'i', 'o', 'u']
myStr = "This is a string!"

def funcA(content, target):
    num = 0

    for char in content:
        if char in target:
            num += 1

    return num


result = funcA(myStr, charList)
print(result)
```

A. 0
B. 1
C. 2
D. 3
✓ E. 4
F. 5

What does the function funcA do?

How can the function be modified to count vowels regardless of case (uppercase or lowercase)?

Q2: What is the output of the following Python program?

```
myStr = "Hello"
result = 0

def funcA(content):
    num = 0

    for char in content:
        num += 1

    return num


funcA(myStr)
print(result)
```

A. 0
B. 1
C. 2
D. 3
E. 4
F. 5

What is the output of previous slide?

Q2: What is the output of the following Python program?

```
myStr = "Hello"
result = 0

def funcA(content):
    num = 0

    for char in content:
        num += 1

    return num


funcA(myStr)
print(result)
```

✓ A. 0
B. 1
C. 2
D. 3
E. 4
F. 5

Q2: What is the output of the following Python program?

```
myStr = "Hello"
result = 0

def funcA(content):
    num = 0

    for char in content:
        num += 1

    return num

result = funcA(myStr)
print(result)
```

5

✓ A. 0
B. 1
C. 2
D. 3
E. 4
F. 5

Functions Calling Functions

- Functions are made to solve a problem and can be called from other functions.

- Functions calling functions is the same as users calling functions.
  - There is no limit to the “depth” of multiple function calls.
  - Deep function calls could make following the flow of a program difficult.

What does it mean when we say "a function can be called from another function"?

Functions Calling Functions: Example

```
funcA('abc')
```

positive!

```python
def str_length(a_str):
    count = 0

    for ch in a_str:
        count = count + 1;

    return count
```

```python
def funcA (text):
    length = str_length(text)

    if length > 0:
        return "positive!"

    elif length < 0:
        return "negative!"

    else:
        return "zero!"
```

What will be the output of the following code?

```
def funcA(x):
    return x + 1

def funcB(y):
    return funcA(y) * 2

print(funcB(3))
```

8