CSE 2040 Programming IV

Lecture #14

What will we learn today

- Introduction to function in Python
- Difference between argument and parameter
- Understanding positional and keyword parameters

What we need to know about functions in Python

- Defining a function
- Arguments to a function
- Function Parameters vs Arguments
- Types of parameters and arguments
- Understanding `print` function
- Order of specifying parameters
- Scope of variables and name resolution
- Function annotations
- Function decorator
- Generator functions
- Anonymous functions
- Using functions in other modules

Defining a function

```
# def is the keyword to indicate a function
   '''Documentation about the function'''
def functionName():
   statement1
   statement2
   …
   statementn
```

Sample functions

```
def functionName(numb):
    statement1
    statement2
    …
    return a + b
def functionName(numb):
    statement1
    statement2
    …
    return
def functionName(numb):
    statement1
    statement2
    # No return statement
```

- No type need be specified
- Python deciphers it from the argument passed to the function when the function is called

```
functionName(121)
functionName(203)
```

Returning values from a function

- Return statement ends the execution of the function call
  1. Return the value of the expression following the `return` keyword
  2. Return `None` when return statement is without an expression
  3. Return `None` when no explicit return statement
  4. Return more than one value using sequences

Parameters vs Arguments

What	Description	Types
Parameter	• Named entity in a function or a method • Defines types of arguments a function can accept • Can specify optional or mandatory	1. Positional or Keyword 2. Positional-only 3. Keyword-only 4. var-positional 5. var-keyword
Argument	• Valued passed to a function or a method when calling the function or method	1. Positional 2. Keyword 3. Packed positional 4. Packed keyword

Types of parameters

Type	Description	Example
Positional or Keyword	• Normal parameters in a function definition – with or without default values • Each parameter has a name and an index • Can accept a positional argument with the same index • Can accept a keyword argument with the same name • Can accept nothing if it has a default value	`def fn(x, y = 10):` [**Note**: Default values are evaluated only once]