IG CS Subroutine Rev

Subroutine

- A self-contained piece of code that has an identifier (name), and it can be called from anywhere in the main program.
- Subroutines are useful because they reduce code. You write the subroutine one, and then you can call it as many times as you need to, instead of having to re-write it every time. Each time you re-write it there is a chance of an error, so this reduces the chances of this error.
- Subroutines is divided into **Procedures** and **Functions**.

Procedures and Functions

1. Procedures

- A subroutine that does not return a value to the program that called it.
- Pseudocode Grammar:

```
PROCEDURE <identifier>(<param1> : <datatype>, <para2> : <datapyte>...)
  <statement(s)>
ENDPROCEDURE

CALL <identifier>(Value1, Value2)  // use this to call the fucntion
```

- Pseudocode Example:

```
// An program that outputs number 1 to 10
PROCEDURE Output1To10()
  For Count <- 1 to 10
    OUTPUT Count
  NEXT Count
ENDPROCUDURE

// The main program can then call the procudure with the code
CALL Output1To10
```

- Python Grammar:

```
def identifier(param1, param2...):
    code to run inside the procedure

identifier(Value1, Value2)
```

- Python Example:

```
# The piece of code below will also outputs number 1 to 10 (or to Num if you want)
def Output1ToNum(Num):
    for i in range(1, Num+1):
        print(i)

Output1ToNum(10)
```

2. Functions

- A function returns a value to the program that called it.
- Pseudocode Grammar:

```
FUNCTION <identifier>(<param1> : <datatype>, <para2> : <datapyte>...)
RETURNS <data type>:
  <statement(s)>
  RETURN data
ENDFUNCTION

<identifier>(param1, param2...)
```

- Pseudocode Example:

```
FUNCTION Plus (Num1 : INTERGER, Num2 : INTEGER) returns INTERGER:
        RETURN Num1 + Num2
ENDFUNCTION

DECLARE Sum : INTERGER
Sum <- Plus(5, 15)
```

- Python Grammar:

```
def identifier(param1, param2...):
    code to run inside the procedure
    return data

identifier(Value1, Value2)
```

- Python Example:

```
def Plus(Num1, Num2):
    return Num1 + Num2

Sum = Plus(5, 15)

print(Sum)
```

Scope

- The sections in the code where the variable, or constant, can be accessed.

Global and Local scope

- Global scope
  - The variable or constant can be accessed from any part of the program.
  - You need to use a global statement to globalized the variable in python.
  - Example

    ```
    Sum = 0

    def Plus(Num1, Num2):
        global Sum
        Sum = Num1 + Num2

    Plus(5, 15)
    print(Sum)
    ```

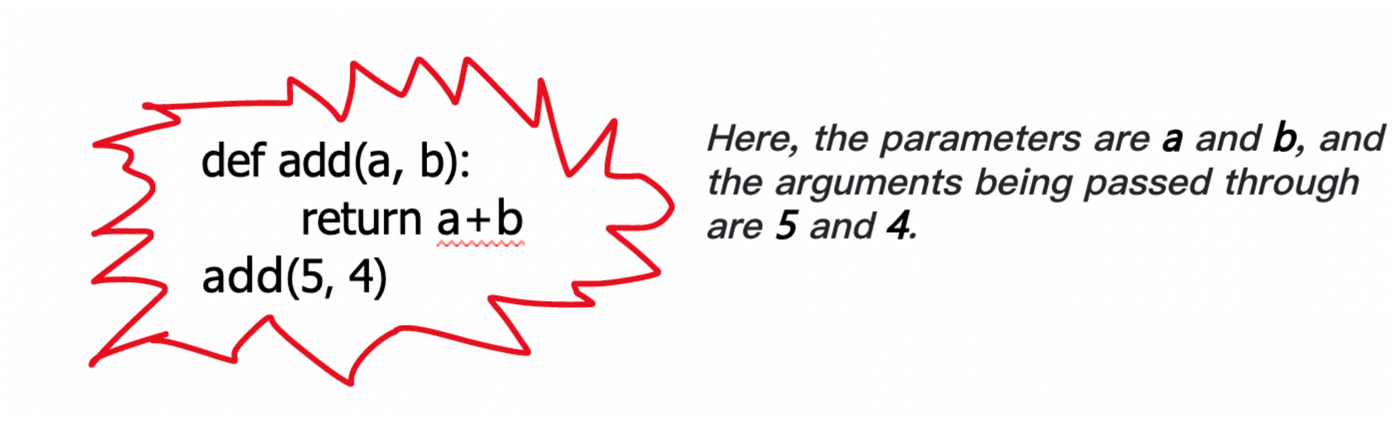
    The global statement here makes the procedure to be able to change the scope "Sum"
- Local scope
  - The variable or constant can only be accessed in the subroutine it is declared within.

Parameters

- A parameter is a value that is sent from the main program to the subroutine (procedure or function). Parameters are declared inside the brackets after the subroutines name.

Parameters and Arguments

- Parameters:
  - What is declared in the function
- Arguments:
  - What is passed through when calling the function



```
def add(a, b):
        return a+b
add(5, 4)
```

*Here, the parameters are **a** and **b**, and the arguments being passed through are **5** and **4**.*

Library Routines

- A program library is a set of subroutines that are pre-written and that can be called within a program
- A pre-writtten subroutine that can be called from within a program
- Library Routine you need to know:
    - round()

      ```
      a = 1.231311
      a = round(a, 1)
      ```

      This piece of code will store the value of a to 1 decimal place
    - random.randint()

      ```
      import random  # remember to include this
      a = random.randint(1, 1000)
      ```

      This piece of code would randomly generate an integer from 1 to 1000

Created by HardyWen