Lecture 06: Bubble Sort

March 2, 2023

1 Functions in Python

Ahmed Moustafa

2 Functions

- Functions are reusable blocks of code that perform a specific task
- They can take input parameters and return output values
- Functions are essential in modular programming, as they help organize code and make it more readable

3 Defining a Function

- To define a function in Python, use the keyword “def” followed by the function name and input parameters in parentheses
- The function body is indented below the header line
- Use the keyword “return” to specify the output value(s) of the function

4 Calling a Function

To call a function, use its name followed by input values in parentheses The function returns the output value(s), which can be stored in a variable or used directly

5

5 Default Parameter Values

- Functions can have default values for input parameters, which are used when no value is provided
- Default values are specified in the function header

```
Hello, Alice
```

[5]:

```
greet("Bob", "Hi")
```

```
Hi, Bob
```

6 Variable-Length Arguments

- Variable-length arguments allow a function to accept any number of input arguments
- They are useful when the number of input arguments is unknown or can vary

[6]:

```
def add_numbers(*args):
    result = 0
    for num in args:
        result += num
    return result

print(add_numbers(1, 2, 3))
```

```
6
```

[7]:

```
print(add_numbers(1, 2, 3, 4, 5))
```

```
15
```

7 Lambda Functions

- Lambda functions are anonymous functions that can be defined inline and used immediately
- They are useful for simple tasks that don't require a named function

[8]:

```
double = lambda x: x * 2
print(double(3))
```

```
6
```

8 Recursion

- Recursion is a technique where a function calls itself
- It is useful for solving problems that can be broken down into smaller subproblems

[9]:

```
def factorial(n):
    if n == 0:
        return 1
    else:
        return n * factorial(n-1)

print(factorial(5))
```

```
120
```

9 Global vs Local Variables

- Global variables are defined outside of any function and can be accessed from anywhere in the program
- Local variables are defined inside a function and can only be accessed within that function

[10]:

```
global_var = 10

def my_func():
    local_var = 20
    print(global_var)
    print(local_var)

my_func()
```

```
10
20
```

[11]:

```
print(global_var)
```

```
10
```

[12]:

```
print(local_var)
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[12], line 1
----> 1 print(local_var)

NameError: name 'local_var' is not defined
```

10 Error Handling

- Error handling is the process of detecting and responding to errors in a program

- Python has a try-except block for handling exceptions that might occur in a function

[13]:

```
def divide(x, y):
    try:
        result = x / y
    except ZeroDivisionError:
        print("Error: division by zero")
    else:
        return result

print(divide(10, 5))
```

```
2.0
```

[14]:

```
print(divide(10, 0))
```

```
Error: division by zero
None
```

11 Conclusion

- Functions are a fundamental concept in Python programming
- They help to modularize code, making it more organized and easier to maintain
- There are many types of functions and techniques for working with them
- Understanding these concepts can help to write more efficient, flexible, and readable code