1.

① Functions are Objects

Python functions are first class objects. In the example below, we are assigning function to a variable. The assignment doesn't call the function. It takes the function object referenced by shout and creates a second name pointing to it.

```
def shout(text):
    return text.upper()
print(shout('Hello'))
yell = shout
print(yell('Hello'))
```

Output

```
HELLO
HELLO
```

First Class functions in Python

- A function is an instance of the Object type
- You can store the function in a variable.
- You can pass the function as a parameter to another function.
- You can return the function from a function
- You can store them in data structures such as hash tables, lists, ...

(2) Functions can be passed as arguments to other functions.

```
def shout(text):
    return text.upper()
def whisper(text):
    return text.lower()
def greet(func):
    greeting = func("Hi, I am created by a function passed as an argument.")
    print(greeting)
greet(shout)
greet(whisper)
```

(3) Functions can return another function:

```
def create_adder(x):
    def adder(y):
        return x+y
    return adder
add_15 = create_adder(15)
print(add_15(10))
```

Recursive Functions

Recursion is the process of defining something in terms of itself. A physical world example would be to place two parallel mirrors facing each other. Any object in between them would be reflected recursively.

Advantages of Recursion

1. Recursive functions make the code look clean and elegant.
2. A complex task can be broken down into simpler sub-problems using recursion.
3. Sequence generation is easier with recursion than using some nested iteration.

Disadvantages of Recursion

1. Sometimes the logic behind recursion is hard to follow through.
2. Recursive calls are expensive (inefficient) as they take up a lot of memory and time.
3. Recursive functions are hard to debug.