Intro to Parameters

Dragons vs Goblins

A computer game company is working on some back-end code for a new game, **Bad Goblins**

A player encounters a wild **goblin** **5 times** in the forest.

- The player **attacks**!
- The **goblin** **defends**!
- The winner is announced.

Open your IDE, copy/paste the starter code and run the file.

Run the code.

Does the program work as expected?

Code Analysis

Answer the following questions:

Why is *`name`* a global variable?	
What range of numbers does `roll_attack()` and `roll_defense()` produce?	
Which function calls `roll_attack()` and `roll_defence()`?	
Which function calls `encounter()`?	

Program Development #1

The designers want to make the program more interesting!

For example, when calling `roll_attack()` there should be a different range for each call.

For example if `roll_attack()` is called like this:

```
roll_attack(10)
```

then it will generate a random number between 1 and 10. If it is called like this:

```
roll_attack(101)
```

then it will generate a random number between 1 and 101.

How can we make this possible?

One strategy is to redesign the `roll_attack()` function and make use of **parameters**:

```
def roll_attack(num): #num is a parameter!
    global attack
    attack = random.randint(1,num)
```

Cool!

Now we can **call** the function by sending some data. The parameter **num** will grab the data and help the function to process that data!

eg

```
roll_attack(10)  #will generate random number, 1-10
roll_attack(100) #will generate random number, 1-100
roll_defense(50) #will generate random number, 1-50
roll_defense(700)    #will generate random number,
1-700
```

1. Redesign the `roll_attack()` and `roll_defense()` functions so that they process a **parameter, num**.

2. In the `encounter()` function, call `roll_attack()` and `roll_defence()` as follows:

    ```
    roll_attack(50)
    roll_defense(50)
    ```

Does your program still work? **Mess around** with the numbers when calling those functions.

Program Development #2
The designers want the player to **encounter** different enemies in the forest, not just a goblin.

For example, we want to call the `encounter()` function like this:

```
encounter("goblin")
encounter("witch")
encounter("dragon")
```

We will need to redesign the `encounter()` function to process a parameter eg:

```
def encounter(enemy):
    pass
```

3. Redesign the `encounter()` function so that it processes a **parameter** representing the kind of enemy being encountered.

   Does the program work as expected.

Program Development #3
In the encounter function, the following code is included after the 2 dice rolls:

```
if enemy == "witch":
    spellPower = random.randint(5,15)
    cast_spell(spellPower)
```

The `cast_spell()` function analyses `spellPower`
If the value is greater than 10, then 5 is added to the global variable `defense`.

Define the `cast_spell` function as follows:

```
def cast_spell(sp):
    pass #your code
```

Program Development #4

Your ideas!

Can you further develop Bad Goblins using your own ideas?

Think about:

- Variable scope
- Functions with parameters
- Random numbers
- Selection (if..elif..else)

*If we can figure out how to share data between functions by using **parameters**, it reduces our need for global variables.*

Good computer programs don't rely on too many global variables.

I wonder why?