The pass command in Python

In the previous lesson, you learned how to use the `continue` and `break` commands in Python to modify loops. In this section, we will continue with another command, the `pass` command in Python, used as a placeholder for the execution of functions, loops, etc., in the future.

In the previous lesson, you learned how to use commands `continue`and loops `break`in Python to modify loops. In this section, we will continue with another command – the `python` `pass`in Python – used as a placeholder for the execution of functions, loops, etc., in the future.

In Python, `pass`` is an empty statement. Simply put, the statement `pass`does nothing; it only holds place for functions and loops that you've added but aren't currently using, reserving them for future expansion.

So, are commands `pass`with comments the same? Absolutely not. The code compiler will completely ignore comments, but commands with comments `pass`will not be ignored.

The pass command in Python

The structure of the pass command

```
pass
```

We often use this command as a placeholder. For example, you have a loop or a function, but you're unsure how to build it or how to code it optimally, and you want to leave it for later. However, that function or command cannot have an empty block of code; the compiler will report an error. Therefore, you can simply use this command `pass`to build an empty block of code, and the compiler will understand and won't complain.

Example of the pass command

```
# pass ch? gi? ch? các kh?i l?
nh trong for: sequence = {'p', 'a', 's', 's'} for val in sequence: pass
```

In the code above, `pass`an empty block of code is created for the for loop . When you run this block of code, you'll see that nothing happens; a code `pass`like that is considered successful.

You can do the same for any functions or classes you want to save:

```
def function(args): pass
```

or:

```
class vidu: pass
```

Example: Check two numbers a and b with corresponding values ??a=50 and b=100. If a>b, process the result later; otherwise, print the message "b is greater than a". The specific code would be as follows:

```
a = 50 b = 100 if b  a: pass else: print("b l?n h?n a")
```

In the example above, the output would be the text "" `b l?n h?n a`. If we swap the values ??of the two numbers a and b: a=100 and b=50, the output will be nothing (because the command `pass`is executed without error).

In the next lesson, you will learn about iteration techniques in Python with many illustrative examples, so don't miss it.