Congratulations! You passed!

Grade received 100% **To pass** 80% or higher

Go to next item

1. How are while loops and for loops different in Python? **1 / 1 point**

- While loops can be used with all data types, for loops can only be used with numbers.
- For loops can be nested, but while loops can't.
- (selected) While loops iterate while a condition is true, for loops iterate through a sequence of elements.
- While loops can be interrupted using break, for loops using continue.

Correct
You got it! We can use while loops when we want our code to execute repeatedly while a condition is true, and for loops when we want to execute a block of code for each element of a sequence.

2. Fill in the blanks to make the factorial function return the factorial of n. Then, print the first 10 factorials (from 0 to 9) with the corresponding number. Remember that the factorial of a number is defined as the product of an integer and all integers before it. For example, the factorial of five (5!) is equal to 1*2*3*4*5=120. Also recall that the factorial of zero (0!) is equal to 1. **1 / 1 point**

```
def factorial(n):
    result = 1
    for x in range(1,n):
        result = result * x
    return result

for n in range(0,10):
    print(n, factorial(n + 1))
```

Run
Reset

```
0 1
1 1
2 2
3 6
4 24
5 120
6 720
7 5040
8 40320
9 362880
```

Correct
Great work! The pieces of code you're tackling keep getting more complex, you're doing a great job!

3. Write a script that prints the first 10 cube numbers (x**3), starting with x=1 and ending with x=10. **1 / 1 point**

```
for num in range(1,11):
  script = num ** 3
  print(script)
```

Run
Reset

```
1
8
27
64
125
216
343
512
729
1000
```

Correct
You nailed it! You got the code to print the first 10 cubes.

4. Write a script that prints the multiples of 7 between 0 and 100. Print one multiple per line and avoid printing any numbers that aren't multiples of 7. Remember that 0 is also a multiple of 7. **1 / 1 point**

```
for i in range(0, 100, 7):
    print(i)
```

Run
Reset

```
0
7
14
21
28
35
42
49
56
63
70
77
84
91
98
```

Correct
Awesome! You're getting Python to do all the work for you.

5. The retry function tries to execute an operation that might fail, it retries the operation for a number of attempts. Currently the code will keep executing the function even if it succeeds. Fill in the blank so the code stops trying after the operation succeeded. **1 / 1 point**

```
def retry(operation, attempts):
  for n in range(attempts):
    if operation():
      print("Attempt " + str(n) + " succeeded")
      break

    else:
      print("Attempt " + str(n) + " failed")

retry(create_user, 3)
retry(stop_service, 5)
```

Run

Reset

```
Attempt 0 failed
Attempt 1 failed
Attempt 2 succeeded
Attempt 0 succeeded
Attempt 0 failed
Attempt 1 failed
Attempt 2 failed
Attempt 3 succeeded
None
```

Correct

Well done, you! You've fixed the code to stop executing once the function is successful.