Module 3 Graded Assessment
Graded Quiz • 50 min

Due Aug 17, 12:59 PM +06

Congratulations! You passed!
TO PASS 80% or higher

Keep Learning

GRADE
90%

Module 3 Graded Assessment

LATEST SUBMISSION GRADE
90%

1. Fill in the blanks of this code to print out the numbers 1 through 7.

1 / 1 point

```
number = 1
while number <= 7:
    print(number, end=" ")
    number += 1
```

Run
Reset

Correct

Nice job! You're really getting the hang of what goes into the while loops!

2. The show_letters function should print out each letter of a word on a separate line. Fill in the blanks to make that happen.

1 / 1 point

```
def show_letters(word):
    for i in word:
        print(i)

show_letters("Hello")
# Should print one line per letter
```

Run
Reset

Correct

Great job! You're working the "for" loops the way they're supposed to be done!

3. Complete the function digits(n) that returns how many digits the number has. For example: 25 has 2 digits and 144 has 3 digits. **Tip:** you can figure out the digits of a number by dividing it by 10 once per digit until there are no digits left.

1 / 1 point

```
def digits(n):
    count = 0
    if n == 0:
      return 1
    while (n> 0 ):
        count += 1
        n= n//10
    return count

print(digits(25))   # Should print 2
print(digits(144))  # Should print 3
print(digits(1000)) # Should print 4
print(digits(0))    # Should print 1
```

Run
Reset

Correct

Woohoo! You've cracked the code of writing code!

4. This function prints out a multiplication table (where each number is the result of multiplying the first number of its row by the number at the top of its column). Fill in the blanks so that calling multiplication_table(1, 3) will print out:

1 / 1 point

```
1 2 3

2 4 6

3 6 9
```

```
def multiplication_table(start, stop):
    for x in range (start ,stop+1):
        for y in range (start,stop+1):
            print(str(x*y), end=" ")
        print()

multiplication_table(1, 3)
# Should print the multiplication table shown above
```

Run
Reset

Correct

Awesome! You've stepped up to the challenge of one of the more complex coding practices, nested loops!

5. The counter function counts down from start to stop when start is bigger than stop, and counts up from start to stop otherwise. Fill in the blanks to make this work correctly.

0 / 1 point

```
def counter(start, stop):
    x = start
    if start> stop :
        return_string = "Counting down: "
        while x >= stop:
            return_string += str(x)
            if x!=stop:
                return_string += ","
            x=-1
    else:
        return_string = "Counting up: "
        while x <= stop:
            return_string += str(x)
            if x!=stop:
                return_string += ","
            x+= 1
    return return_string

print(counter(1, 10)) # Should be "Counting up: 1,2,3,4,5,6,7,8,9,10"
print(counter(2, 1)) # Should be "Counting down: 2,1"
```

Run
Reset

```
print(counter(5, 5)) # Should be "Counting up: 5"
```

! **Incorrect**

Not quite, counter(2, 1) returned Counting down: 2,, should be Counting down: 2,1.

6. The even_numbers function returns a space-separated string of all positive numbers that are divisible by 2, up to and including the maximum that's passed into the function. For example, even_numbers(6) returns "2 4 6". Fill in the blank to make this work.

1 / 1 point

```
def even_numbers(maximum):
    return_string = ""
    for x in range (1,maximum+1 ):
        if (x%2 ==0):
            return_string += str(x) + " "
    return return_string.strip()

print(even_numbers(6))  # Should be 2 4 6
print(even_numbers(10)) # Should be 2 4 6 8 10
print(even_numbers(1))  # No numbers displayed
print(even_numbers(3))  # Should be 2
print(even_numbers(0))  # No numbers displayed
```

Run

Reset

✓ **Correct**

Woohoo! You remembered all of the elements of the range of the for-loop, well done!

7. The following code raises an error when executed. What's the reason for the error?

1 / 1 point

```
def decade_counter():
    while year < 50:
        year += 10
    return year
```

- ○ Incrementing by 10 instead of 1
- [✓ Failure to initialize variables]
- ○ Nothing is happening inside the while loop
- ○ Wrong comparison operator

✓ **Correct**

Well done! The variable year needs to be initialized prior to being used in the while loop.

8. What is the value of x at the end of the following code?

1 / 1 point

```
for x in range(1, 10, 3):
    print(x)
```

7

✓ **Correct**

You got it! The upper limit of a range isn't included, which means that the loop stops before reaching it. The increment is 3, so the loop stops when x reaches 7.

9. What is the value of y at the end of the following code?

1 / 1 point

```
for x in range(10):
    for y in range(x):
        print(y)
```

8

✓ **Correct**

Great job! The upper limit of a range isn't included, which means that the outer loop goes up to 9, so the highest upper limit for the inner loop is 9, which is also not included.

10. How does this function need to be called to print yes, no, and maybe as possible options to vote for?

1 / 1 point

```
def votes(params):
    for vote in params:
        print("Possible option:" + vote)

```

- ○ votes("yes", "no", "maybe")
- ○ votes(yes, no, maybe)
- ○ votes([yes, no, maybe])
- [✓ votes(['yes', 'no', 'maybe'])]

✓ **Correct**

Excellent! This function is looking for one argument, and the list of strings is just one argument.