**Congratulations! You passed!**

**TO PASS**80% or higher

Keep Learning

**GRADE**

80%

**Module 3 Graded Assessment**

**LATEST SUBMISSION GRADE**

80%

1.Question 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

RunReset

1 2 3 4 5 6 7

**Correct**

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

the while loops!

2.Question 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

RunReset

H

e

l

l

o

**Correct**

Great job! You're working the "for" loops the way they're

supposed to be done!

3.Question 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>=1:

        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

RunReset

2

3

4

1

**Correct**

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

4.Question 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 2 3

2 4 6

3 6 9

**1 / 1 point**

def multiplication\_table(start, stop):

    for x in range(1,stop+1):

        for y in range(1,stop+1):

            print(str(x\*y), end=" ")

        print()

multiplication\_table(1, 3)

# Should print the multiplication table shown above

1 2 3

2 4 6

3 6 9

**Correct**

Awesome! You've stepped up to the challenge of one of the

more complex coding practices, nested loops!

5.Question 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.

**1 / 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"

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

RunReset

Counting up: 1,2,3,4,5,6,7,8,9,10

Counting down: 2,1

Counting up: 5

**Correct**

You nailed it! You've figured out all of the situations that

need to be considered!

6.Question 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(2,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

RunReset

2 4 6

2 4 6 8 10

2

**Correct**

Woohoo! You remembered all of the elements of the range of

the for-loop, well done!

7.Question 7

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

def decade\_counter():

    while year < 50:

        year += 10

    return year

**1 / 1 point**



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.Question 8

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

1

2

for x in range(1, 10, 3):

    print(x)

**0 / 1 point**

1 4 7

**Incorrect**

The answer you gave is not a number.

9.Question 9

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

1

2

3

for x in range(10):

    for y in range(x):

        print(y)

**0 / 1 point**

0010120123012340123450123456012345670123456780123456789

**Incorrect**

Not quite. You may want to review the videos about the "for" loops, and how the range counters work.

10.Question 10

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

def votes(params):

    for vote in params:

        print("Possible option:" + vote)

**1 / 1 point**



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.