**Practice Quiz 1: While Loops**

**1.What are while loops in Python?**

While loops let the computer execute a set of instructions while a condition is true.

**2.Fill in the blanks to make the print\_prime\_factors function print all the prime factors of a number. A prime factor is a number that is prime and divides another without a remainder.**

def print\_prime\_factors(number):

# Start with two, which is the first prime

factor = 2

# Keep going until the factor is larger than the number

while factor <= number:

# Check if factor is a divisor of number

if number % factor == 0:

# If it is, print it and divide the original number

print(factor)

number = number / factor

else:

# If it's not, increment the factor by one

factor = factor + 1

return "Done"

print\_prime\_factors(100)

# Should print 2,2,5,5

# DO NOT DELETE THIS COMMENT

**3.The following code can lead to an infinite loop. Fix the code so that it can finish successfully for all numbers. Note: Try running your function with the number 0 as the input, and see what you get!**

def is\_power\_of\_two(n):

if n == 1:

return True

# Check if the number can be divided by two without a remainder

while n != 0 and n % 2 == 0:

n = n / 2

# If after dividing by two the number is 1, it's a power of two

if n == 1:

return True

return False

print(is\_power\_of\_two(0)) # Should be False

print(is\_power\_of\_two(1)) # Should be True

print(is\_power\_of\_two(8)) # Should be True

print(is\_power\_of\_two(9)) # Should be False

print(is\_power\_of\_two(64)) # Should be True

**4.Fill in the empty function so that it returns the sum of all the divisors of a number, without including it. A divisor is a number that divides into another without a remainder.**

def sum\_divisors(n):

sum = 0

for i in range(1,n):

if n % i == 0:

sum = sum + i

# Return the sum of all divisors of n, not including n

return sum

print(sum\_divisors(0))

# 0

print(sum\_divisors(3)) # Should sum of 1

# 1

print(sum\_divisors(36)) # Should sum of 1+2+3+4+6+9+12+18

# 55

print(sum\_divisors(102)) # Should be sum of 2+3+6+17+34+51

# 114

**5.The multiplication\_table function prints the results of a number passed to it multiplied by 1 through 5. An additional requirement is that the result is not to exceed 25, which is done with the break statement. Fill in the blanks to complete the function to satisfy these conditions.**

def multiplication\_table(number):

# Initialize the starting point of the multiplication table

multiplier = 1

# Only want to loop through 5

while multiplier <= 5:

result = number \* multiplier

# What is the additional condition to exit out of the loop?

if result > 25 :

break

print(str(number) + "x" + str(multiplier) + "=" + str(result))

# Increment the variable for the loop

multiplier += 1

multiplication\_table(3)

# Should print: 3x1=3 3x2=6 3x3=9 3x4=12 3x5=15

multiplication\_table(5)

# Should print: 5x1=5 5x2=10 5x3=15 5x4=20 5x5=25

multiplication\_table(8)

# Should print: 8x1=8 8x2=16 8x3=24