Module 3: Expanding Logical Structures With Iteration

Quick Reference Guide

Learning Outcomes

* Recognize, interpret, and write programs using while loops and for loops.
* Apply indexes and slices to strings and lists to access individual parts.
* Recognize, interpret, and write programs that iterate through lists and strings with for loops.
* Evaluate provided test sets and write new test sets to verify that code works as expected.

While Loops

def sum\_to\_n(n):

counter = 1

total = 0

while(counter <= n):

total += counter

counter += 1

return total

print(sum\_to\_n(4))

def sum\_from\_m\_to\_n(m, n):

index = m

result = 0

while(index <= n):

result += index

index += 1

return result

print(sum\_from\_m\_to\_n(3, 5))

def sum\_of\_odds\_v1(m, n):

result = 0

index = m

while (index <= n):

if (index % 2 == 1):

result += index

index += 1

return result

print(sum\_of\_odds\_v1(2, 7))

def sum\_of\_odds\_v2(m, n):

if (m % 2 == 0):

m += 1

result = 0

index = m

while (index <= n):

result += index

index += 2

return result

print(sum\_of\_odds\_v2(2, 7))

For Loops

def sum\_to\_n(n):

total = 0

for i in range(n+1):

total += i

return total

print(sum\_to\_n(4))

def sum\_from\_m\_to\_n(m, n):

total = 0

for i in range(m, n+1):

total += i

return total

print(sum\_from\_m\_to\_n(3, 5))

def sum\_of\_odds\_v1(m, n):

total = 0

for i in range(m, n+1):

if (i % 2 == 1):

total += i

return total

print(sum\_of\_odds\_v1(2, 7))

def sum\_of\_odds\_v2(m, n):

total = 0

if (m % 2 == 0):

m += 1

for i in range(m, n+1, 2):

total += i

return total

print(sum\_of\_odds\_v2(2, 7))