

## Intro to Python Solutions

(the solutions are in the 'your code here' sections)

### For Tests

---

```
#!/usr/bin/python3
#
def for_one(n):
    c = 0
    #Given the number variable n
    #Write a for loop to add all numbers from 0
    #to n. Store the result in c.
    #
    #IE: n = 10
    #c = 0 + 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10
    #
    #WWW YOUR CODE HERE WWW
    for i in range(n+1):
        c += i
    #WWW YOUR CODE HERE WWW
    return c
def for_two(list_one):
    list_two = []
    #Given the list variable list_one
    #Write a for loop to add every item
    #to the list variable list_two
    #
    #WWW YOUR CODE HERE WWW
    list_two = []
```

---

Brought to you by:

**CYBRARY** | FOR BUSINESS

Develop your team with the **fastest growing catalog** in the cybersecurity industry. Enterprise-grade workforce development management, advanced training features and detailed skill gap and competency analytics.

# CYBRARY

---

```
for i in list_one:
list_two.append(i)
##### YOUR CODE HERE #####
return list_two
def for_three(list_one, list_two):
list_three = []
#Given the list variables list_one and list_two
#Write a for loop to add the values together
#and store the result in list_three
#
#Note: This requires a bit of research. A look at
#the zip function in the Python documentation
#should help. If you get stuck, have a look at for_solved.py
#
##### YOUR CODE HERE #####
for i, j in zip(list_one, list_two):
list_three.append(i + j)
##### YOUR CODE HERE #####
return list_three
```

---

## Functions Test

---

```
#!/usr/bin/python3
#
#For this exercise, you'll be
#implementing three functions.
#
#The first function will be called "multiply"
#It will take two arguments, a and b
#and will return the result of the multiplication.
#
```

---

Brought to you by:

**CYBRARY** | FOR BUSINESS

Develop your team with the **fastest growing catalog** in the cybersecurity industry. Enterprise-grade workforce development management, advanced training features and detailed skill gap and competency analytics.

# CYBRARY

---

```
#The second function will be called "append"
#It will take two arguments, list_one and list_two
#and will return a list made of both inputs combined.
#
#The third function will be called "say"
#It will take two string arguments, name and phrase
#and will return "<name> says <phrase>"
#EX: say("joe", "Python is easy!")
#OUTPUT: "joe says Python is easy!"
def multiply(a,b):
return a*b
def append(list_one, list_two):
return list_one+list_two
def say(name, phrase):
return "{} says {}".format(name, phrase)
```

---

## If Tests

---

```
#!/usr/bin/python3
#
#Using the variables a and b (provided for you)
#and the Python constants True and False
#https://docs.python.org/3/library/constants.html
#
#Implement the boolean operations listed below.
#Store the results in c.
#
#It is recommended that you review the lesson
#on Boolean logic.
#
#EXAMPLE:
```

---

Brought to you by:

**CYBRARY** | FOR BUSINESS

Develop your team with the **fastest growing catalog** in the cybersecurity industry. Enterprise-grade workforce development management, advanced training features and detailed skill gap and competency analytics.

# CYBRARY

```
def logical_inversion(a):
#implement the logical operation NOT
if a == True:
c = False
elif a == False:
c = True
return c #don't worry about this line of code yet.
def logical_conjunction(a,b):
#implement the logical operation AND
#WWW YOUR CODE HERE WWW
if a == True and b == True:
c = True
else:
c = False
#WWW YOUR CODE HERE WWW
return c #don't worry about this line of code yet.
def logical_disjunction(a,b):
c = ""
#implement the logical operation OR
#WWW YOUR CODE HERE WWW
if a == False and b == False:
c = False
else:
c = True
#WWW YOUR CODE HERE WWW
return c #don't worry about this line of code yet.
def logical_exclusion(a,b):
c = ""
#implement the logical operation XOR
#WWW YOUR CODE HERE WWW
if a != b:
c = True
else:
c = False
```

Brought to you by:

**CYBRARY** | FOR BUSINESS

Develop your team with the **fastest growing catalog** in the cybersecurity industry. Enterprise-grade workforce development management, advanced training features and detailed skill gap and competency analytics.

# CYBRARY

---

```
#\WWW YOUR CODE HERE \WWW
return c #don't worry about this line of code yet.
def inverted_conjunction(a,b):
c = ""
#implement the logical operation NAND
#\WWW YOUR CODE HERE \WWW
if a == True and b == True:
c = False
else:
c = True
#\WWW YOUR CODE HERE \WWW
return c #don't worry about this line of code yet.
def inverted_disjunction(a,b):
c = ""
#implement the logical operation NOR
#\WWW YOUR CODE HERE \WWW
if a == False and b == False:
c = True
else:
c = False
#\WWW YOUR CODE HERE \WWW
return c #don't worry about this line of code yet.
```

---

## While Tests

---

```
#!/usr/bin/python3
#
def while_one(n):
c = 0
#Given the number variable n
#Write a while loop to add all numbers from 0
```

---

Brought to you by:

**CYBRARY** | FOR BUSINESS

Develop your team with the **fastest growing catalog** in the cybersecurity industry. Enterprise-grade workforce development management, advanced training features and detailed skill gap and competency analytics.

# CYBRARY

```
#to n. Store the result in c.
#
#IE: n = 10
#c = 0 + 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10
#
#\\\\\\\\ YOUR CODE HERE \\\\\\\
i = 0
while i < n + 1:
    c += i
    i += 1
#\\\\\\\\ YOUR CODE HERE \\\\\\\
return c
def while_two(list_one):
    list_two = []
    #Given the list variable list_one
    #Write a while loop to add every item
    #to the list variable list_two
    #
    #Note: This may require a bit of research,
    #spending some time reading the python docs
    #about lists will help.
    #\\\\\\\\ YOUR CODE HERE \\\\\\\
    while i < len(list_one):
        list_two.append(list_one[i])
        i += 1
    #\\\\\\\\ YOUR CODE HERE \\\\\\\
    return list_two
```

Brought to you by:

**CYBRARY** | FOR BUSINESS

Develop your team with the **fastest growing catalog** in the cybersecurity industry. Enterprise-grade workforce development management, advanced training features and detailed skill gap and competency analytics.