In [44]:
```
# 1. Write a function that will accept two numbers and return the sum of those two numbers
def sumOfTwo():
    float1 = float(input("Please enter first number: "))
    float2 = float(input("Please enter second number: "))
    sum = float1 + float2
    print("The sum of both numbers is:", sum)
sumOfTwo()
```

```
Please enter first number: 12
Please enter second number: 25
The sum of both numbers is: 37.0
```

In [45]:
```
# 2. Write a function that will accept any amount of numbers provided as individual parameters (not a list)
# and return the sum of those numbers. Use a for loop in the function, not a list comprehension.
def sumOfAny(first, second, *therest):
    bigSum = first + second + sum(therest)
    print("The sum of", first, "+", second, "+", therest, "=", bigSum)

sumOfAny(1, 2, 3, 8, 100)
sumOfAny(12, 13)
```

```
The sum of 1 + 2 + (3, 8, 100) = 114
The sum of 12 + 13 + () = 25
```

In [46]:
```
def foo(first, second, third, *therest): # accept parameters, one to many
    print("First: %s" % first)
    print("Second: %s" % second)
    print("Third: %s" % third)
    print("And all the rest: %s" % list(therest))
foo(2, 4, 6, 8, 100)
```

```
First: 2
Second: 4
Third: 6
And all the rest: [8, 100]
```

In [47]:
```
# 3. Write a function that will return True if a provided string is a palindro
me, and False if it is not.
# first function to reverse word
def reverse(word):
    return word[::-1]

# second function, calling first, checking the palindrome
def isPalindromeOrNot(word):
    revWord = reverse(word)
    if (revWord == word):
        return True
    return False

#isPalindromeOrNot("anna")
isPalindromeOrNot("oscar")
```

Out[47]:
```
False
```

In [48]:
```
# 4. Write a function that will return a list of all of the odd numbers from a
given list.
# Use a for loop, not a list comprehension.
def returnOdds(list):
    oddsList = [] # IMPORTANT> NEW LIST to fill with results from operation mu
st be declared BEFORE for loop that performs!!
    for i in list:
        if (i % 2 == 0):
            oddsList.append(list[list.index(i)])
            print("value", i)
            print("at index", list.index(i))
    print(oddsList)

listToTry = [2, 3, 4, 5, 6, 7, 8, 9]
returnOdds(listToTry)
```

```
value 2
at index 0
value 4
at index 2
value 6
at index 4
value 8
at index 6
[2, 4, 6, 8]
```

In [49]:
```
# 5. Amend your answer to question 4 to include a try…except clause which will
print an informative error message if a list
# element which is not a number is encountered.
# The error message should be: “Type Error Caught: [the built-in exception mes
sage]”
def returnOddsIntsOnly(list):
    oddsList = [] # IMPORTANT> NEW LIST to fill with results from operation mu
st be declared BEFORE for loop that performs!!
    for i in list:
        try: #****************************************** try
            i = int(i) #******************************* check if i is int, con
tinue if it is int
            if (i % 2 == 0):
                oddsList.append(list[list.index(i)])
                print("value", i)
                print("at index", list.index(i))
        except ValueError: #*************************** for every "non-int" th
row exception
            print(i, "is not an int") #**************** explain what happened
    print(oddsList)

listToTry = [2, 3, "f", 4, 5, 6, 7, 8, 9]
returnOddsIntsOnly(listToTry)
```

```
value 2
at index 0
f is not an int
value 4
at index 3
value 6
at index 5
value 8
at index 7
[2, 4, 6, 8]
```

In [50]:
```
# 6. Amend your answer to question 3 to include a try…except clause
# which will print an informative error message if a string is not supplied.
def reverse2(word):
    return word[::-1]

# second function, calling first, checking the palindrome
def isPalindromeOrNotWithStringCheck(word):
    try:
        isinstance(word, str) == True
        revWord = reverse2(word)
        if (revWord == word):
            #return True
            print(word, "is a palindrome.")
            #return False
        else:
            print(word, "is NOT a palindrome.")
    except:
        print(word, "That is not a string, dummy!")

isPalindromeOrNotWithStringCheck(121)
#isPalindromeOrNotWithStringCheck("anna")
#isPalindromeOrNotWithStringCheck("oscar")
```

```
121 That is not a string, dummy!
```

In [51]:
```
# 7. The following code example will not execute without an error.
# Amend the code to catch the specific exception thrown and print the followin
g error message: “Key not found: [name of key]”

def returnValue(key):
    animals = {'cow':'moo','cat':'meow','dog':'bark'}
    try:
        return animals[key]
    except:
        print("Key not found:", key)

#returnValue('cat')
returnValue('sheep')
```

```
Key not found: sheep
```