```
########################################
### EXAMPLE: Buggy code to reverse a list
### Try to
debug it! (fixes needed are explained below)
########################################
##def
rev_list_buggy(L):
##    """
##    input: L, a list
##    Modifies L such that
its elements are in reverse order
##    returns: nothing
##    """
##    for i
in range(len(L)):
##        j = len(L) - i
##        L[i] = temp
##        L[i] = L[j]
##
 L[j] = L[i]
#
## FIXES: --------------------------
## temp unknown
## list index out of range
-> sub 1 to j
## get same list back -> iterate only over half
##
--------------------------
def rev_list(L):
    """
    input: L, a list

Modifies L such that its elements are in reverse order
    returns: nothing

"""
    for i in range(len(L)//2):
        j = len(L) - i - 1
        temp =
L[i]
        L[i] = L[j]
        L[j] = temp

L =
[1,2,3,4]
rev_list(L)
print(L)
#
#
########################################
### EXAMPLE: Buggy
code to get a list of primes
### Try to debug it! (fixes needed are explained
below)
########################################
##def primes_list_buggy(n):
##
"""
##    input: n an integer > 1
##    returns: list of all the primes up to
and including n
##    """
##    # initialize primes list
##    if i == 2:
##
   primes.append(2)
##    # go through each elem of primes list
##    for i in
range(len(primes)):
##        # go through each of 2...n
##        for j in range(len(n)):
##
         # check if not divisible by elem of list
##            if i%j != 0:
```

```
##
primes.append(i)
#
#
## FIXES: -------------------------
## = invalid syntax, variable i
unknown, variable primes unknown
## can't apply 'len' to an int
## division by zero ->
iterate through elems not indices
##                   -> iterate from 2 not 0
## forgot to
return
## primes is empty list for n > 2
## n = 3 goes through loop once -> range to n+1
not n
## infinite loop -> append j not i
##                -> list is getting modified as
iterating over it!
##                -> switch loops around
## n = 4 adds 4 -> need way to
stop going once found a divisible num
##               -> use a flag
##
-------------------------
def primes_list(n):
    """
    input: n an integer
> 1
    returns: list of all the primes up to and including n
    """
    #
initialize primes list
    primes = [2]
    # go through each of 3...n
    for j in
range(3,n+1):
        is_div = False
        # go through each elem of primes list
        for
p in primes:
            if j%p == 0:
                is_div = True
        if not is_div:

       primes.append(j)
    return primes

print(primes_list(2) )

print(primes_list(15)  )


######################################
# EXAMPLE:
Exceptions and input
######################################
#a = int(input("Tell me one
number: "))
#b = int(input("Tell me another number: "))
#print("a/b =
", a/b)
#print("a+b = ", a+b)

try:
    a = int(input("Tell me one number:
"))
    b = int(input("Tell me another number: "))
    print("a/b = ",
a/b)
except:
    print("Bug in user input.")


try:
    a = int(input("Tell me
one number: "))
    b = int(input("Tell me another number: "))

print("a/b = ", a/b)
    print("a+b = ", a+b)
except ValueError:

print("Could not convert to a number.")
except ZeroDivisionError:

print("Can't divide by zero")
except:
    print("Something went very
wrong.")



#######################################
# EXAMPLE: Raising your own
exceptions
#######################################
def get_ratios(L1, L2):

""" Assumes: L1 and L2 are lists of equal length of numbers
        Returns: a
list containing L1[i]/L2[i] """
    ratios = []
    for index in
range(len(L1)):
        try:
            ratios.append(L1[index]/L2[index])
        except
ZeroDivisionError:
            ratios.append(float('nan')) #nan = Not a Number
        except:

           raise ValueError('get_ratios called with bad arg')
        else:

print("success")
        finally:
            print("executed no matter
what!")
    return ratios

print(get_ratios([1, 4], [2,
4]))


########################################
## EXAMPLE: Exceptions and
lists
########################################
def get_stats(class_list):
        new_stats = []
        for
person in class_list:
                new_stats.append([person[0], person[1], avg(person[1])])
        return
new_stats

# avg function: version without an exception
#def avg(grades):
#    return
(sum(grades))/len(grades)

# avg function: version with an exception
def avg(grades):

try:
        return sum(grades)/len(grades)
    except ZeroDivisionError:

print('warning: no grades data')
```

```
        return 0.0


# avg function: version with assert
def
avg(grades):
    assert len(grades) != 0, 'warning: no grades data'
    return
sum(grades)/len(grades)


test_grades = [[['peter', 'parker'], [80.0, 70.0, 85.0]],

       [['bruce', 'wayne'], [100.0, 80.0, 74.0]],
              [['captain', 'america'], [80.0,
70.0, 96.0]],
              [['deadpool'], []]]

print(get_stats(test_grades))
```