Chapter 7: Exceptions

Last Update: March 27, 2023

**Program 1**:

07 11-why.py

```
 1 # unvalidated input
 2 # 0 <= n <= 100
 3 # x.xxx is not a valid int
 4 #
 5 def next():
 6     input("\nNext> ")
 7
 8 print("\t*** Case 1: Trust the user")
 9 x = int(input("Enter a number (0-100): "))
10 print(x)
11 next()
12
13 print("\t*** Case 2: Repeat input")
14 while True:
15     n = int(input("Enter a number (0-100): "))
16     if 0<=n<=100:
17         break
18 print(n, "is in the range.")
19 next()
20
21 print("\t*** Case 3: Repeat input")
22 n = int(input("Enter a number (0-100): "))
23 while 0<=n<=100:
24     print (n, "is not in the range. Try again.")
25     n = int(input("Enter a number (0-100): "))
26 else:
27     print(n, "is positive.  Thank you.")
```

**Program 2**:

07 12-solution.py

```
# Multiple exceptions
# Assert some condition
# What can go wrong?

while True:
    try:
        x = int(input("Please enter a number: "))
        assert(0<=x<=100)
        break
    except ValueError:
        print("Oops! Not an integer number.  ")
    except AssertionError:
        print("Oops! Not between 0 & 100. ")

print('Wonderful...')
```

**Program 3**:

07 dd-divide.py

```
# A simple try-except

try:
    i = int(input('Enter an integer: '))
    i = i/i
    print(i)
except Exception as ex:
    print(ex)
```

**Program 4**:

07 21-all.py

```
try:
    x = 3 + 3
except:
    x *= 2
else:
    x += 1
finally:
    print(f'The value of x is {x}')
```

**Program 5**:

07 22-all.py

```
# This example shows the use of all four clauses
# the except can be repeated multiple times
# else and finally are optional
#
def read_file(path):
    try:
        f = open(path)
        data = f.read()
    except FileNotFoundError:
        print(f'File {path} not found.')
    else:
        f.close()
        return data
    finally:
        print("-------------------------")
    return None

name = input("Please enter a file name: ")
print(read_file(name))
print("-------------------------")
```

**Program 6**:

07 24-just do it.py

```
# using the exception, we can "skip" cases that we are unable to
handle
# In some sense, try-except can replace an if-else

list = [10, 20, 'Just Try It', 9.99, 'zero', 'Just do it',
        [1, 2, 399999], 100]

sum = 0
for element in list:
    try:  # use try as an if # if element is a number
        sum = sum + element
    except:  # with Exception or not  # else
        print(element)
        pass
print(sum)
```

**Program 7**:

07 25-nested.py

```
 1 # Like almost everything else,try can be nested
 2 # Indent properly.
 3 #
 4 try:
 5     f = open("demofile.txt")
 6     try:
 7         f.write("Looks good")
 8     except:
 9         print("Something went wrong when writing to the file")
10     finally:
11         f.close()
12 except:
13     print("Something went wrong when opening the file")
```

**Program 8**:

07 31-assert.py

```
def next():
    input("Next> ")

# Assert without handling
print("\t*** Case 1: assert only")
x = int(input("Please enter a number: "))
assert(0<=x<=100)
print("Wonderful!")
next()

# Assert with handling
# It does not crash
#
print("\t*** Case 2: assert in try")
try:
    x = int(input("Please enter a number: "))
    assert(0<=x<=100)
    print("Wonderful!")
except AssertionError:
    print(f'X = {x} is outside the range. Give up.')
next()

print("\t*** Case 3: assert in try + loop")
while True:
    try:
        x = int(input("Please enter a number: "))
        assert(0<=x<=100)
        break
    except AssertionError:
        print(f'X = {x} is outside the range.')
print(f'Wonderful.  x = {x}')
next()

# This is a very defensive program
# It won't give up until a right number is entered
#
def get_x():
    while True:
        try:
            x = int(input("Please enter a number: "))
            assert(0<=x<=100)
            return x
        except AssertionError:
            print(f'X = {x} is outside the range.')

print("\t*** Case 4: function-while-try")
x = get_x()
print(f'Wonderful.  x = {x}')
```

**Program 9**:

07 32-raise.py

```
1  temp = 105
2  try:
3      if temp > 100:
4          raise Exception(f'temperature (={temp}) should not exceed 100.' )
5  except Exception as e:
6      print(e)
```