Daniel Frey
CS 3160
Assignment 5
11/7/18

1. Does exception have to be part of the type system of a language?
   b. No

2. For a language to support exception, it must support:
   a. An operation to raise an exception
   b. An operation to handle an exception

3. Write what the output would be for each of the code fragments.
   i.
```
count = 0
for letter in "Snow!":
    print("Letter #", count, "is", letter)
    count+=1
```

   output:
```
Letter # 0 is S
Letter # 1 is n
Letter # 2 is o
Letter # 3 is w
Letter # 4 is !
```

   ii.
```
num = 10
while True:
    if num < 7:
        break;
    print(num)
    num-=1
```

   output:
```
10
9
8
7
```

4. Write a program to generate a dictionary that contains (n: n*n*n), where key is n and value is n*n*n for all values from 1 to n. Display dictionary.
   Suppose if input given is n = 5 then output should be {1: 1, 2: 8, 3: 27, 4: 64, 5: 125}

```
n = 5
cube_dict = dict()
for i in range(1,n+1):
  cube_dict[i] = i**3
print(cube_dict)
```

   output: {1: 1, 2: 8, 3: 27, 4: 64, 5: 125}

5. Add static method **is_workingday()** to Employee class. Accepts date (year, month, date) as argument and returns True (if working day) or False (Saturday/Sunday). User provides date. Include entire class **Employee**. Create 3 employee objects and display all results using all methods.

```
import datetime
class Employee:
  num_of_emp = 0

  def __init__(self, fname, lname, eid):
    self.fname, self.lname, self.eid = fname, lname, eid
    Employee.num_of_emp+=1

  def displayname(self):
    print(self.fname, self.lname)

  @classmethod
  def disp_num_of_emps(cls):
    print(Employee.num_of_emp)

  @staticmethod
  def is_workingday(year, month, day):
    year, month, day = int(year), int(month), int(day)
    if datetime.date(year, month, day).weekday() < 5 :
      return True
    else:
      return False

emp1 = Employee("John", "Doe", 1234)
emp1.displayname()
emp1.disp_num_of_emps()
print("2018/11/5 working day?", emp1.is_workingday(2018, 11, 5), '\n')

emp2 = Employee("Jane", "Doe", 4321)
emp2.displayname()
emp2.disp_num_of_emps()
print("2018/11/11 working day?", emp2.is_workingday(2018, 11, 11), '\n')
```

```
emp3 = Employee("Ronald", "McDonald", 1)
emp3.displayname()
emp3.disp_num_of_emps()
print("2018/11/12 working day?", emp3.is_workingday(2018, 11, 12), '\n')
```

output:

John Doe
1
2018/11/5 working day? True

Jane Doe
2
2018/11/11 working day? False

Ronald McDonald
3
2018/11/12 working day? True

6. Create **point** class. Overload > operator. Check if one point greater than the other. For example. **point**(1,1) > **point**(-2,-3) displays False.
(Hint: To compare, find magnitude of each point using formula $x^2 + y^2$)

```
class point:
    def __init__(self, x, y):
        self.x, self.y = x, y

    def __gt__(self, other):
        self.mag = self.x**2 + self.y**2
        other.mag = other.x**2 + other.y**2
        return self.mag > other.mag

point(1, 1) > point(-2, -3)
```

Output: False