1. (50 pts) You have been given a set of related files that need to be put together in a project, properly imported, modified, and used:

* Entry
* HeapPQ
* Queue
* Graphs

You need to complete the following tasks:

* 1. Build a project.
  2. Add the files above to the project.
  3. Add the necessary import statements to the files to make them work together properly.
  4. Write another file MyGraph.py.
  5. In MyGraph.py, add the necessary import statements.
  6. In MyGraph.py, define a function MyGraph which accepts three optional named parameters:
  + GraphType One of the following:
    - AdjacencySetGraph
    - UndirectedGraph
    - Digraph
    - WeightedGraph – the default value
  + V(ertices) – a Set containing the vertices with a default of an empty Set
  + E(dges) – a Set with tuples defining each edge with a default of an empty Set. If the graph type is digraph or weighted, the definition will include the weight.
  1. The function MyGraph will return a populated graph ADT based on the type chosen.
  2. Add an appropriate method(s) within the collection of code to be able to print the graph in the format introduced in the text:
  + For example: print(G)
  + Returns something like: ({1, 2, 3},{(1, 2), (1, 3), (2, 3)})  
    For a digraph or weighted graph, weights are not printed. **Under this line, describe what you did, in what file, and where in the code to achieve this**.

Initialize MyGraph checking for the type of graph that the data is giving us. This allows to categorize how to initialize that data set and tell us if it even has edges. The third value of edges tells us the weight, so I checked for length of this. This leverages the string representation.

* 1. Write a function neighbors(G) which will accept a graph and prints out the neighbors of each vertex in the following format. Note again that the weights do not need to be provided for a digraph or weighted graph:
  + Vertex 1: 2, 3
  + Vertex 2: 3 (1, 3 if an undirected graph or weighted graph)
  + Vertex 3: None (1, 2 if an undirected graph or weighted graph)

**Submission:**

In a single posting on HuskyCT (you do get multiple attempts to submit to get it correct):

* Entry.py (potentially modified)
* HeapPQ.py (potentially modified)
* Queue.py (potentially modified)
* Graphs.py (potentially modified)
* MyGraph.py