**LAB # 10**

**LIST, TUPLE, DICTIONARY, CLASS**

**OBJECT AND IMPLEMENT PRIORITY QUEUE**

**OBJECTIVE:**

Familiarization with python language using list, tuple, dictionary, class and object and Manage a set of records with priority queue using queue and heapq module in python

**Lab Tasks:**

1. Store the names of a few of your friends in a list called names. Print each person’s name by accessing each element in the list, one at a time.

* **CODE:**



* **OUTPUT:**

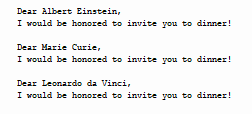


1. If you could invite anyone, living or deceased, to dinner, who would you invite? Make a list that includes at least three people you’d like to invite to dinner. Then use your list to print a message to each person, inviting them to dinner.

* **CODE:**



* **OUTPUT:**

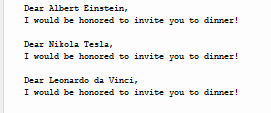


1. **Changing Guest List:** You just heard that one of your guests can’t make the dinner, so you need to send out a new set of invitations. You’ll have to think of someone else to invite.

* Modify your list, replacing the name of the guest who can’t make it with the name of the new person you are inviting.
* Print a second set of invitation messages, one for each person who is still in your list.
* **CODE:**

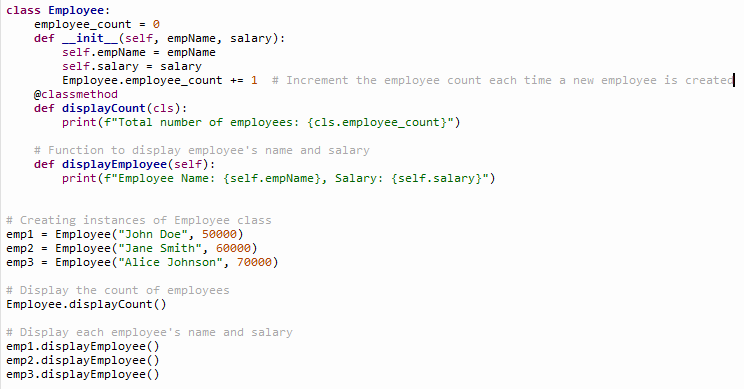


* **OUTPUT:**



1. Create a Class “Employee”, it’s a common base class for all the employee. Then initialize employee’s parameter like empName and salary and create function like displayCount() contain total number of employee in your knowledge base and displayEmployee() contain empName and their salary.

* **CODE:**

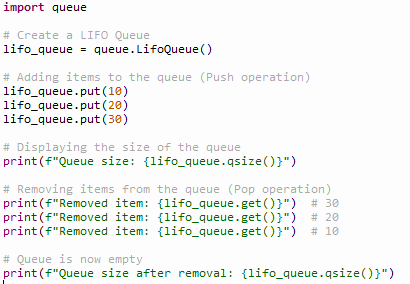
****

* **OUTPUT:**

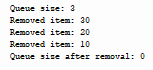
****

1. In contrast to the standard FIFO implementation of Queue, the LifoQueue uses last-in, first- out ordering (normally associated with a stack data structure). Implement LIFO queue using queue module.

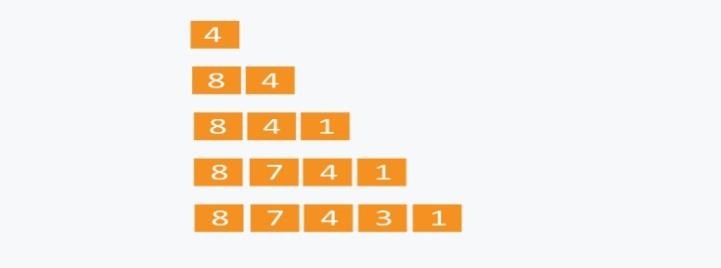
* **CODE:**

****

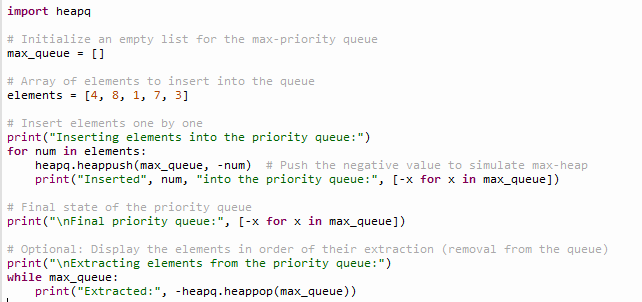
* **OUTPUT:**



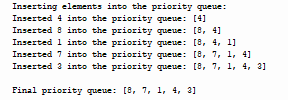
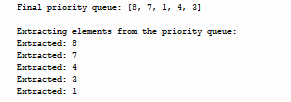
1. We have an array of 5 elements: [4, 8, 1, 7, 3] and we have to insert all the elements in the max-priority queue. First as the priority queue is empty, so 4 will be inserted initially. Now when 8 will be inserted it will move to front as 8 is greater than 4. While inserting 1, as it is the current minimum element in the priority queue, it will remain in the back of priority queue. Now 7 will be inserted between 8 and 4 as 7 is smaller than 8. Now 3 will be inserted before 1 as it is the 2nd minimum element in the priority queue. All the steps are represented in the diagram below:



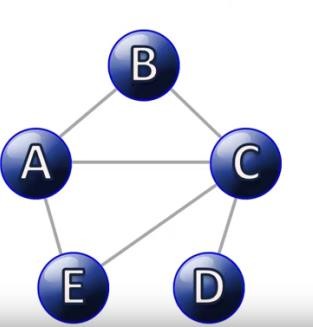
* **CODE:**



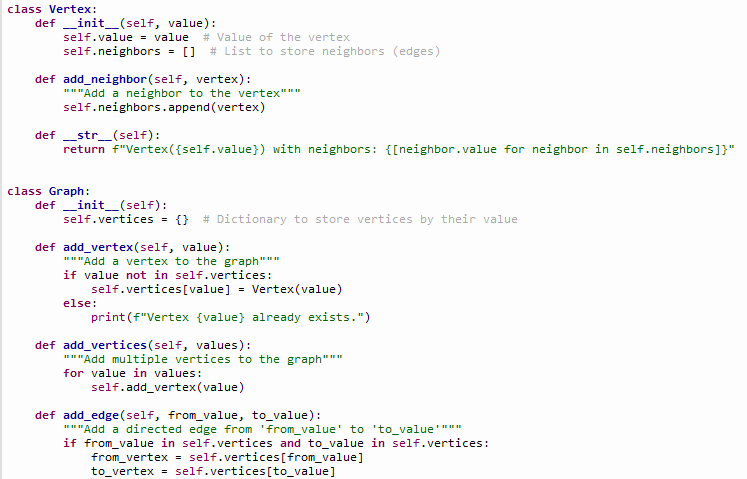
* **OUTPUT:**

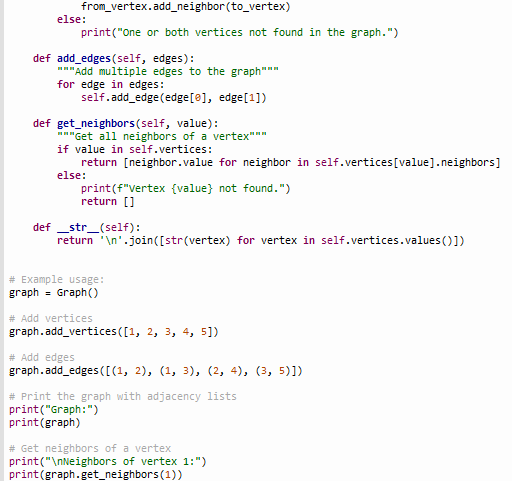


1. Implement graph using adjacency list using list or dictionary, make a class such as Vertex and Graph then make some function such as add\_nodes, add\_edges, add\_neighbors, add\_vertex, add\_vertices and suppose whatever you want to need it.



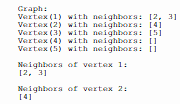
* **CODE:**

****





* **OUTPUT:**



**Files Upload On Github:**