BELOW ARE SAMPLE OUTPUT OF WHAT IS HAPPENING AFTER VARIOUS ADD, POP, CHANGE PRIORITY, ETC ACTIVITIES. These tests were made to test the basic functionalities of add, pop, length, changePriority functions.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*SAMPLE CALLS THAT WERE MADE\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Data\_list = [(10,77),(11,77),()]

storeQ.add(10,77);

storeQ.add(11,77);

storeQ.add(12,77);

storeQ.add(40,2);

storeQ.add(200,10);

storeQ.add(50,9);

storeQ.add(41,3);

storeQ.add(42,3);

storeQ.add(43,3);

storeQ.add(40,3);

storeQ.add(20,1);

storeQ.add(30,2);

storeQ.length(); //This is checking if it's retuning the correct count value

storeQ.pop();

storeQ.changePriority(40,10);

storeQ.add(30,3);

storeQ.add(60,2);

storeQ.changePriority(20,4);

storeQ.add(30,2);

storeQ.pop();

storeQ.pop();//this pop is to test what happens when the last value of the highest priority is popped. Should it stay as empty item with a key?

console.log("all priorities are: "+storeQ.get\_all\_priorities());

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*THE OUTPUT LOOKS LIKE\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

After adding value:10 and key:77 the queue is: {"77":[10]}

After adding value:11 and key:77 the queue is: {"77":[10,11]}

After adding value:12 and key:77 the queue is: {"77":[10,11,12]}

After adding value:40 and key:2 the queue is: {"2":[40],"77":[10,11,12]}

After adding value:200 and key:10 the queue is: {"2":[40],"10":[200],"77":[10,11,12]}

After adding value:50 and key:9 the queue is: {"2":[40],"9":[50],"10":[200],"77":[10,11,12]}

After adding value:41 and key:3 the queue is: {"2":[40],"3":[41],"9":[50],"10":[200],"77":[10,11,12]}

After adding value:42 and key:3 the queue is: {"2":[40],"3":[41,42],"9":[50],"10":[200],"77":[10,11,12]}

After adding value:43 and key:3 the queue is: {"2":[40],"3":[41,42,43],"9":[50],"10":[200],"77":[10,11,12]}

After adding value:40 and key:3 the queue is: {"2":[40],"3":[41,42,43,40],"9":[50],"10":[200],"77":[10,11,12]}

After adding value:20 and key:1 the queue is: {"1":[20],"2":[40],"3":[41,42,43,40],"9":[50],"10":[200],"77":[10,11,12]}

After adding value:30 and key:2 the queue is: {"1":[20],"2":[40,30],"3":[41,42,43,40],"9":[50],"10":[200],"77":[10,11,12]}

For pop the maxKey value is: 77 and the delKey is: 10

After pop the queue is: {"1":[20],"2":[40,30],"3":[41,42,43,40],"9":[50],"10":[200],"77":[11,12]}

After pop the count is: 11

Bingo!!! found the value 40 to change priority to 10

found key is 2 and found index is: 0

Deleted key is: 40

After adding value:40 and key:10 the queue is: {"1":[20],"2":[30],"3":[41,42,43,40],"9":[50],"10":[200,40],"77":[11,12]}

the element is after changePriority: {"1":[20],"2":[30],"3":[41,42,43,40],"9":[50],"10":[200,40],"77":[11,12]}

After adding value:30 and key:3 the queue is: {"1":[20],"2":[30],"3":[41,42,43,40,30],"9":[50],"10":[200,40],"77":[11,12]}

After adding value:60 and key:2 the queue is: {"1":[20],"2":[30,60],"3":[41,42,43,40,30],"9":[50],"10":[200,40],"77":[11,12]}

Bingo!!! found the value 20 to change priority to 4

found key is 1 and found index is: 0

Deleted key is: 20

After adding value:20 and key:4 the queue is: {"1":[],"2":[30,60],"3":[41,42,43,40,30],"4":[20],"9":[50],"10":[200,40],"77":[11,12]}

the element is after changePriority: {"1":[],"2":[30,60],"3":[41,42,43,40,30],"4":[20],"9":[50],"10":[200,40],"77":[11,12]}

After adding value:30 and key:2 the queue is: {"1":[],"2":[30,60,30],"3":[41,42,43,40,30],"4":[20],"9":[50],"10":[200,40],"77":[11,12]}

For pop the maxKey value is: 77 and the delKey is: 11

After pop the queue is: {"1":[],"2":[30,60,30],"3":[41,42,43,40,30],"4":[20],"9":[50],"10":[200,40],"77":[12]}

7":[12]}

After pop the count is: 15

For pop the maxKey value is: 77 and the delKey is: 12 7":[]}

After pop the queue is: {"1":[],"2":[30,60,30],"3":[41,42,43,40,30],"4":[20],"9":[50],"10":[200,40],"7

7":[]}

After pop the count is: 14

all priorities are: 1,2,3,4,9,10,77

\*\*\*\*\*\*\*\*\*\*\*\*\*\*BELOW IS SAMPLE TESTS TO SEE IF THE “forEach” FUNCTION ITERATING THROUGH THE ENTIRE storeQ ITEMS CORRECTLY OR NOT (This was tested without a callback(function) parameter\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

storeQ.add(10,77);

storeQ.add(11,77);

storeQ.add(12,77);

storeQ.add(40,2);

storeQ.add(200,10);

storeQ.add(50,9);

storeQ.add(41,3);

storeQ.add(42,3);

storeQ.add(43,3);

storeQ.add(40,3);

storeQ.add(20,1);

storeQ.add(30,2);

storeQ.length(); //This is checking if it's retuning the correct count value

storeQ.pop();

storeQ.forEach(); //this was tested without a passing a callback (function) parameter to test if all functionalities are working correctly or not

\*\*\*\*\*\*\*\*\*\*\*\*\*\*OUTPUT AS BELOW\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

After adding value:200 and key:10 the queue is: {"2":[40],"10":[200],"77":[10,11,12]}

After adding value:50 and key:9 the queue is: {"2":[40],"9":[50],"10":[200],"77":[10,11,12]}

After adding value:41 and key:3 the queue is: {"2":[40],"3":[41],"9":[50],"10":[200],"77":[10,11,12]}

After adding value:42 and key:3 the queue is: {"2":[40],"3":[41,42],"9":[50],"10":[200],"77":[10,11,12]}

After adding value:43 and key:3 the queue is: {"2":[40],"3":[41,42,43],"9":[50],"10":[200],"77":[10,11,12]}

After adding value:40 and key:3 the queue is: {"2":[40],"3":[41,42,43,40],"9":[50],"10":[200],"77":[10,11,12]}

After adding value:20 and key:1 the queue is: {"1":[20],"2":[40],"3":[41,42,43,40],"9":[50],"10":[200],"77":[10,11,12]}

After adding value:30 and key:2 the queue is: {"1":[20],"2":[40,30],"3":[41,42,43,40],"9":[50],"10":[200],"77":[10,11,12]}

For pop the maxKey value is: 77 and the delKey is: 10

After pop the queue is: {"1":[20],"2":[40,30],"3":[41,42,43,40],"9":[50],"10":[200],"77":[11,12]}

After pop the count is: 11

Key is: 1

Values are 20

Key is: 2

Values are 40

Values are 30

Key is: 4

Values are 41

Values are 42

Values are 43

Values are 40

Key is: 1

Values are 50

Key is: 1

Values are 200

Key is: 2

Values are 11

Values are 12

Key is: 4

Values are 41

Values are 42

Values are 43

Values are 40

Key is: 1

Values are 50

Key is: 1

Values are 200

Key is: 2

Values are 11

Values are 12

Key is: 1

Values are 200

Key is: 2

Values are 11

Values are 12

A SAMPLE EXPLORATORY SESSION BASED TEST CASE SCENARIO

(THERE ARE SOME QUESTION MARKS AS SOME REQUIREMENTS/INFORMATIONS ARE MISSING AND NEEDED TO BE ADDRESSED)

**Test Charter:**

Analyze actions of a priority queue, consists with keys(priority) and values by performing add, pop, and change priority, get length actions.

**Areas:**

* Add items in the priority queue with various data type for values. Check if it handles them well or not (e.g. integer, negative number, decimal, fraction, undefined value, etc.). What happens if an user enter decimal, or negative, or undefined value?
* If keys/priorities are not integer, does it notify the user saying keys have to be an integer.
* Add duplicate keys and test it is handled well
* Add multiple values and priorities in the stack
* Add same priority multiple times to see how it’s handled
* Add item with a key and an empty value
* Pop the priority queue randomly and check if it’s acting correctly
* What happens when the last value of a key is popped, should the key be deleted as well or not?
* Add duplicate priority to check how it is handled
* Add with empty priority and value to check error handling
* Test the change priority. E.g. what if the NewPriority already exists in the queue, does it handle it well? Otherwise, does it add a new priority successfully or not?
* For change priority, what if there is multiple matching value found, is it supposed to delete the first one found or delete all the values found?
* Is the length supposed to return the length of the keys or the total length of values of keys?

**Test environment:** Check if the application run well in different environments, e.g. Windows, MAC os., etc.

**Start:** [ENTER START TIME]

**Tester:** [ENTER TESTER NAME]

**Duration:** 90-120 min [estimated time]

**Test Design and execution:** Estimated time to spent is 60-70% in this area.

**Bug Investigaton and reporting:** Estimated time to spent is 20-30% in this area.

**Session setup:** 5-10% of time

**Testing notes:** [To complete this section, some of the questions needed to be answered and the product needed to have some feature completed (e.g. exception handler, etc.). Also due to time restriction, this part couldn’t be completed.]

Data files: [ENTER IF THERE IS ANY LOG FILES FOR BUGS]

Bugs: [ENTER THE BUGS FOUND]