```
"/Volumes/GoogleDrive/My Drive/UMBC Semesters/Spring 2021/Data Struct/Projects/Proj3/cmake-build-debug/Proj3"

queue1 dump after inserting 4 students (Nick, Sam, Liz, Eva):
(((6:Eva)3:Sam)0:Nick(3:Liz))

queue2 dump after inserting 2 students (John, Mia):
((6:John)0:Mia)

Contents of queue1 after merging with queue2:
[0] Student: Nick, priority: 2, year: 0, major: CS, group: Minority
[0] Student: Mia, priority: 3, year: 0, major: CS, group: Minority
[3] Student: Liz, priority: 5, year: 1, major: SCI, group: Minority
[6] Student: John, priority: 6, year: 3, major: SCI, group: Majority
[3] Student: Sam, priority: 1, year: 1, major: ENG, group: Majority
[6] Student: Eva, priority: 4, year: 3, major: SCI, group: Majority

Dump queue1 after merging with queue2:
(((3:Liz)0:Mia(6:John))0:Nick((6:Eva)3:Sam))
Showing queue 2 is empty: Empty skew heap.

Contents of queue1 after changing the priority function to priorityFn2:
[1] Student: Sam, priority: 1, year: 1, major: ENG, group: Majority
[4] Student: Eva, priority: 4, year: 3, major: SCI, group: Majority
[5] Student: Liz, priority: 5, year: 1, major: SCI, group: Minority
[2] Student: Nick, priority: 2, year: 0, major: CS, group: Minority
[6] Student: John, priority: 6, year: 3, major: SCI, group: Majority
[3] Student: Mia, priority: 3, year: 0, major: CS, group: Minority

```

```
Dump queue1 after changing the priority function to priorityFn2:
(((5:Liz)4:Eva)1:Sam((6:John)2:Nick(3:Mia)))

Contents of queue1 after changing the priority function back to priorityFn1:
[0] Student: Nick, priority: 2, year: 0, major: CS, group: Minority
[3] Student: Sam, priority: 1, year: 1, major: ENG, group: Majority
[6] Student: John, priority: 6, year: 3, major: SCI, group: Majority
[6] Student: Eva, priority: 4, year: 3, major: SCI, group: Majority
[0] Student: Mia, priority: 3, year: 0, major: CS, group: Minority
[3] Student: Liz, priority: 5, year: 1, major: SCI, group: Minority

Dump queue1 after changing the priority function back to priorityFn1:
(((6:John)3:Sam(6:Eva))0:Nick((3:Liz)0:Mia))

Final queue1 in priority order:
[0] Student: Nick, priority: 2, year: 0, major: CS, group: Minority
[0] Student: Mia, priority: 3, year: 0, major: CS, group: Minority
[3] Student: Sam, priority: 1, year: 1, major: ENG, group: Majority
[3] Student: Liz, priority: 5, year: 1, major: SCI, group: Minority
[6] Student: John, priority: 6, year: 3, major: SCI, group: Majority
[6] Student: Eva, priority: 4, year: 3, major: SCI, group: Majority
Deletion of an empty queue: test passed

Showing queue 1 is empty: Empty skew heap.
Showing queue 2 is empty: Empty skew heap.

Lets begin copy constructor and assignment override testing:
Show the content of queue1 so far:
((6:Eva)0:Nick((3:Liz)3:Sam))
```

```
Copying into an empty queue, showing that copy tester: Empty skew heap.
Begin assignment override:
Show content of copyTester:
(((6:Eva)3:Liz)0:Nick(3:Sam))
Add an element to copyTester and compare to original queue1:
copyTester content:
(((6:Joud)3:Sam)0:Nick((6:Eva)3:Liz))
O.G. queue1 content:
((6:Eva)0:Nick((3:Liz)3:Sam))

Begin copy constructor testing:
Show the content of copyConTester:
(((6:Joud)6:Eva)0:Nick((3:Sam)3:Liz))
Show the address of copyConTester:
0x7ffeea37e7e0
Show the address of copyTest:
0x7ffeea37e838

Testing a case of an RQueue with a different priroty function being added:
Show the content of priortyCopyTest:
(((6:John)3:Mia)1:Joud)
Show the content of priortyCopyTest after assignment to RQueue with a diffrent priority function:
(((6:Joud)6:Eva)0:Nick((3:Sam)3:Liz))

Now remove an element from priortyCopyTest for fun:
[0] Student: Nick, priority: 2, year: 0, major: CS, group: Minority
(((6:Joud)6:Eva)3:Liz(3:Sam))

```

```
How many elements are in the queue?:
Print the size of the queue: 4

Print the number of students in the queue: 4

Now testing merging of two different priority :
(((5:Liz)2:Joud)1:Sam(4:Eva))
Merge two different priorities queues: test passed

Measuring the efficiency of insertion functionality:
Inserting 1000 members took 275 clock ticks (0.000275 seconds)!
Deletion 1000 members took 298 clock ticks (0.000298 seconds)!
Inserting 2000 members took 547 clock ticks (0.000547 seconds)!
Deletion 2000 members took 612 clock ticks (0.000612 seconds)!
Inserting 4000 members took 1602 clock ticks (0.001602 seconds)!
Deletion 4000 members took 2224 clock ticks (0.002224 seconds)!
exception: The queue is empty...can not delete
exception: You are attempting to merge the queues with different priorities
Inserting 8000 members took 2429 clock ticks (0.002429 seconds)!
Deletion 8000 members took 2698 clock ticks (0.002698 seconds)!
Inserting 16000 members took 4575 clock ticks (0.004575 seconds)!
Deletion 16000 members took 7975 clock ticks (0.007975 seconds)!
Inserting 32000 members took 10277 clock ticks (0.010277 seconds)!
Deletion 32000 members took 16233 clock ticks (0.016233 seconds)!

Process finished with exit code 0

```