The Programming Assignment Report
CSCE 221

1. Purpose of the assignment:

   The assignment involves implementing a minimum priority queue based on linked list, vector and binary heap, and applying the structure to schedule computer jobs by their priority.

2. The description of data structures and algorithms used to solve the problem, and instructions to compile the programs.

   For Phase 1, I implemented a template class that has a vector as its data members,and another template of doubly linked list class similarly to PA1. Both of the two classes have functions is_empty() to check if the queue is empty, function insert_ to insert a new object into the queue and some helper functions to sort the queue, then remove the object that is least significant out of the queue and return it.

   For Phase 2, I implemented a class of Binary Heap based on a vector instead of an array so that I don't have to worry about resizing the array to insert new data to the queue. The implement is similar to Phase 1, with the exception of several functions such as walkUp, walkDown, buildHeap to construct and reconstruct the heap.

   For Phase 3, I implemented some functions to compare CPU jobs priority. These are in another separate header files since compared to the implementation of a MPQs type int, constructing the MPQs for scheduling CPU jobs is more complicated.

   (a) Describe algorithms used to solve the problem.

      i. About the algorithms, I used insertion sort to sort the data in vector MPQ and to insert new nodes into linked list MPQ. It's worst case is O(n^2). By testing using the clock, the runtime is actually less than $O(n^2)$,but more like $O(n)$. Details about runtime will be put below.

   (b) Instructions on handling:

      i. The header files and .cpp files for each phase are put in separate files: Phase 1, Phase 2, and Phase 3. Phase 4 are in each .cpp file, comment out for unnecessary part. Please use both header or cpp files to test each type, int or CPUjob, for example `binary.cpp` and `binaryint.cpp`.

      ii. Compiled by: `g++ -std=c++17`

   (c) Provide features of the C++ programming paradigms like Inheritance or Polymorphism in case of object oriented programming, or Templates in the case of generic programming used in your implementation.

   The struct CPUjob was added to each MPQ header type so that the class member can access ID, length and priority of each job.

```
struct CPUjob {
      int ID;
      int length;
      int prior;
      public:      CPUjob(int a =0, int b=0, int c=0) : ID(a), length(b), prior(c) {};
      int getID() {return this->ID;}
      int getLength() {return this->length;}
      int getPrior() {return this->prior;}
      bool operator<(const CPUjob &job) {return (this->prior < job.prior);}
};
```

3. Provide types of exceptions and their purpose in your program.

   (a) logical exceptions (such as deletion of an item from an empty container, etc.).
   EmptyList exception, invalid argument (not able to find a node to insert before/after).

   (b) runtime exception (such as division by 0, etc.)
   Runtime error for not able to open file.

   (c) Operator overload [], output/input operator overload, comparison operator overload.

4. Test your program for correctness using valid, invalid, and random inputs (e.g., insertion of an item at the beginning, at the end, or at a random place into a sorted vector). Include evidence of your testing, such as an output file or screen shots with an input and the corresponding output.

   Analysis:

Functions	Vector	Linked List	Binary Heap
buildMPQ	$O(n)$	$O(n)$ since traversal through the list to find node	$O(n)$
insert	$O(1)$ push_back + $O(n)$ sort $\rightarrow O(n)$	$O(1)$ insert_first, insert_last; $O(n)$ insert to build queue	$O(n\log_2 n)$
sort	$O(n)$ average, insertion sort	$O(n)$	$O(1)$,only comparisons
min	$O(1)$, min at the head	$O(1)$	$O(1)$ the root
removeMin	$O(n)$, use std::erase	$O(1)$ since it is sorted	$O(\log n)$, swap and walkUp
walkUp	n/a	n/a	$O(logn)$ height of the tree
walkDown	n/a	n/a	$O(logn)$ height of the tree
is_empty	$O(1)$	$O(1)$	$O(1)$

   Runtime for insert data and build a MPQ:

input size	Vector MPQ	Linked list MPQ	Binary MPQ
1000	19435 μs = 19.43 ms	3366μs	719μs
10000	1119861 μs = 1120 ms	375565μs=373ms	1375μs
100000	108s	82s (5658μs to insert a new obj)	43364μs

   It can be seen from the table that Binary Heap data structures takes the least amount of time to build a MPQ, since it is $O(n\log_2 n)$ worst case. Linked List has a slighly better performance even though they are both $O(n)$ in the worst case, since each node is connected with the other two nodes. The performance confirms the analysis theory.

   Output for each cases:

5. Vector:

```
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$ g++ -std=c++17 pqvector.cpp
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$ ./a.out
Input size : 1000
Time it takes to input and sort in a vector:19435
Time it takes to insert in a sorted vector: 8206
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$ g++ -std=c++17 pqvector.cpp
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$ ./a.out
Input size : 10000
Time it takes to input and sort in a vector:1119861
Time it takes to insert in a sorted vector: 720802
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$ g++ -std=c++17 pqvector.cpp
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$ ./a.out
Input size : 10000
Time it takes to input and sort in a vector:1seconds
Time it takes to insert in a sorted vector: 0seconds
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$ g++ -std=c++17 pqvector.cpp
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$ ./a.out
Input size : 100000
Time it takes to input and sort in a vector:108seconds
Time it takes to insert in a sorted vector: 70seconds
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$
```

```
 1tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$ g++ -std=c++17 pqvector.cpp
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$ ./a.out
---------Test with type int----------
Test insert values 30,15,23,-8,1,3,5 (unsorted)
30, 15, 23, -8, 1, 3, 5,
Test insert into sorted vector with value -3:
-8, -3, 1, 3, 5, 15, 23, 30,
Test remove min:
min removed: -8
vector now:
1, 3, 5, 15, 23, 30,
Clear vector and test is_empty:
 1
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$
```

6. Linked List:

```
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$ g++ -std=c++17 pqlinkedlist.cpp
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$ ./a.out
Input size 1000:
------------Before insert-------------
Time it takes to input and sort:3366
------------After insert-------------
Time it takes to input new obj into sorted linked list:15
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$ g++ -std=c++17 pqlinkedlist.cpp
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$ ./a.out
Input size 10000:
------------Before insert-------------
Time it takes to input and sort:375565
------------After insert-------------
Time it takes to input new obj into sorted linked list:267
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$ g++ -std=c++17 pqlinkedlist.cpp
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$ ./a.out
Input size 100000:
------------Before insert-------------
Time it takes to input and sort:82
------------After insert-------------
Time it takes to input new obj into sorted linked list:0
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$ g++ -std=c++17 pqlinkedlist.cpp
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$ ./a.out
Input size 100000:
------------Before insert-------------
Time it takes to input and sort:80
------------After insert-------------
Time it takes to input new obj into sorted linked list:5658
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$
```

```
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$ g++ -std=c++17 pqlinkedlist.cpp
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$ ./a.out
--------Test PQ linked list type int-----------
Check is empty:
1
Check insert input: 8 2 20 0 -15 3 7 8 and make min p queue:
-15 -8 0 2 3 7 8 20
Test insert 100:
-15 -8 0 2 3 7 8 20 100
Test remove min:
min is: -15
-8 0 2 3 7 8 20 100
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$
```

7. Heap

```
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$ g++ -std=c++17 binaryint.cpp
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$ ./a.out
----------Test Binary Heap type int----------
Build heap with values 30,15,23,-8,1,3,5
before delete min: (should be -8 1 3 15 30 23 5)
-8 1 3 15 30 23 5

after delete min: (should now be 1 5 3 15 30 23)
1 5 3 15 30 23
Insert (-3): (should be -3 5 1 15 30 23 3)
-3 5 1 15 30 23 3
 check IsEmpty:
1
terminate called after throwing an instance of 'EmptyHeap'
  what():  Empty Heap
Aborted (core dumped)
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$
```

```
Input size 10:
---------------Build Heap-------------
2 10 -16 9 9 -3 4 7 -16 10 9 -3 1 2 11 8 3 -3 6 2 15 3 8 3 7 7 8 5 9 11
Time it takes to build heap: 0
---------------Insert New-------------
Time it takes to insert new obj into heap: 0
2 10 -16 9 9 -3 4 7 -16 10 9 -3 1 2 11 8 3 -3 6 2 15 3 8 3 7 7 8 5 9 11 112 0 19
---------------Delete Min-------------
Time it takes to delete minimum obj: 0
4 7 -16 9 9 -3 8 3 -3 10 9 -3 1 2 11 112 0 19 6 2 15 3 8 3 7 7 8 5 9 11
---------------Delete Min(2)-------------
8 3 -3 9 9 -3 5 9 11 10 9 -3 1 2 11 112 0 19 6 2 15 3 8 3 7 7 8
---------------Delete Min(3)-------------
9 9 -3 10 9 -3 5 9 11 3 8 3 1 2 11 112 0 19 6 2 15 7 7 8
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$
```

```
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$ g++ -std=c++17 binary.cpp
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$ ./a.out
Input size 1000:
---------------Build Heap-------------

Time it takes to build heap: 719
---------------Insert New-------------
Time it takes to insert new obj into heap: 1

---------------Delete Min-------------
Time it takes to delete minimum obj: 2
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$
```

```
Input size 10000:
---------------Build Heap-------------

Time it takes to build heap: 1375
---------------Insert New-------------
Time it takes to insert new obj into heap: 1

---------------Delete Min-------------
Time it takes to delete minimum obj: 3
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$
```

```
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$ g++ -std=c++17 binary.cpp
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$ ./a.out
Input size 100000:
---------------Build Heap-------------

Time it takes to build heap: 43364
---------------Insert New-------------
Time it takes to insert new obj into heap: 3

---------------Delete Min-------------
Time it takes to delete minimum obj: 10
tqn14@LAPTOP-PEC7250E:/mnt/c/Users/quynh/Documents/csce221/pa2$
```