Algorithms and Data Structures

Summer Semester 2025

For discussion on Wednesday, June 18, 2025

1. (GTG Exercise C-9.27) Show how to implement the FIFO queue ADT using only a priority queue and one additional integer instance variable.

2. (GTG Exercise C-9.28) Professor Idle suggests the following solution to the previous problem. Whenever an item is inserted into the queue, it is assigned a key that is equal to the current size of the queue. Does such a strategy result in FIFO semantics? Prove that it is so or provide a counterexample.

3. (GTG Exercise R-10.21) The following implementation of binary search is used by GTG in their implementation of the `SortedTableMap` class:

```
1  def _find_index(self, k, low, high):
2    """Return index of the leftmost item with key greater than or equal
                                          to k.
3
4    Return high + 1 if no such item qualifies.
5
6    That is, j will be returned such that:
7       all items of slice table[low:j] have key < k
8       all items of slice table[j:high+1] have key >= k
9    """
10   if high < low:
11     return high + 1                                  # no element
                                        qualifies
12   else:
13     mid = (low + high) // 2
14     if k == self._table[mid]._key:
15       return mid                                     # found exact match
16     elif k < self._table[mid]._key:
17       return self._find_index(k, low, mid - 1)     # Note: may return
                                          mid
18     else:
19       return self._find_index(k, mid + 1, high)    # answer is right
                                          of mid
```

Consider the following variant of the find index method:

```
def _find_index(self, k, low, high):
  if high < low:
    return high + 1
  else:
    mid = (low + high) // 2
    if self._table[mid].key < k:
      return self._find_index(k, mid + 1, high)
    else:
      return self._find_index(k, low, mid - 1)
```

Does this always produce the same result as the original version? Justify your answer.

4. (GTG Exercise R-10.6) Which of the hash table collision-handling schemes could tolerate a load factor above 1 and which could not?

5. (GTG Exercise R-10.18) Explain why a hash table is not suited to implement a sorted map.

6. (GTG Exercise C-10.45) Describe how to modify a skip-list representation so that index-based operations, such as retrieving the item at index j, can be performed in $O(\log n)$ expected time.