CSCE 221
Problem Set 11

Jacob Purcell, Texas A&M, Student

I.

Set class is a list of unique values, an AVL tree will be used to store the data.

Algorithm 1 Void Add: X

```
Require: $AVL\ Tree\ Data\ Member$
  $temp = Contains\ X$
  if $temp$ then
    if $temp\ is\ greater\ than\ X$ then
      $Assign\ X\ to\ temp\ right\ child$
    end if
    if $temp\ is\ less\ than\ X$ then
      $Assign\ X\ to\ temp\ left\ child$
    end if
  end if
  $Increment\ size$
  $Rebalance\ tree$
```

Algorithm 2 Void Clear

```
Require: $AVL\ Tree\ Data\ Member$
  for $iterator = 0$ to $Size$ do
    $Delete\ Root\ Node\ of\ AVL\ tree$
    $Rebalance\ tree$
  end for
  $set\ size\ to\ zero$
```

II.

A map binds a value set to a key set. A single set using the AVL ADT can do both jobs by holding the information in both in a single node. Done this way, the map can just be an extension of the set class to include multiple data storage inside nodes.

Algorithm 3 Node pointer Contains: X

```
Require: $AVL\ Tree\ Data\ Member$
  $Increment\ size$
  $Pointer\ C\ to\ root$
  while $not\ null$ do
    if $X\ is\ greater\ than\ current\ node$ then
      if $right\ child\ of\ C\ is\ the\ null\ pointer$ then
        return $C$
        $Break\ while\ loop$
      end if
      $assign\ C\ to\ point\ to\ its\ right\ child$
    end if
    if $X\ is\ less\ than\ current\ node$ then
      if $left\ child\ of\ C\ is\ the\ null\ pointer$ then
        return $C$
        $Break\ while\ loop$
      end if
      $assign\ C\ to\ point\ to\ its\ left\ child$
    end if
    if $X = dereferenced\ C$ then
      return $nullptr$
    end if
  end while
  return $C$
```

Algorithm 4 Node pointer Find: X

```
Require: AVL Tree Data Member
  Increment size
  Pointer C to root
  while not null do
    if X is greater than current node then
      if right child of C is the null pointer then
        return nullprt
        Break while loop
      end if
      assign C to point to its right child
    end if
    if X is less than current node then
      if left child of C is the null pointer then
        return nullptr
        Break while loop
      end if
      assign C to point to its left child
    end if
    if $X = dereferenced\ C$ then
      return C
    end if
  end while
  return C
```

Algorithm 5 Void Delete: X

```
Require: AVL Tree Data Member
  Pointer C = Find X
  if both of C children are nullptr then
    DeleteC
  end if
  Pointer B = C left child
  while B right child is not nullptr do
    Assign B to be its right child
  end while
  set value at C to be B
  C takes over right subtree of B if needed
  delete B
  Rebalance tree
```

Algorithm 6 boolean IsEmpty

```
Require: size data member
  return size
```

Algorithm 7 unsigned number Size

```
Require: size data member
  return size
```

Algorithm 8 Void Add: key, value

```
Require: AVL tree Set
  Add(key, value) to set
```

Algorithm 9 Void Remove: key

```
Require: AVL tree Set
  Delete(key) from set
```

Algorithm 10 Node Object Find: key

```
Require: AVL tree Set
  return Find(key) in set
```