CIS 23: Data Structures and Algorithms

Homework 4
Prof. Sana Vaziri

Cody Vig

Problem 1

Based on the `arrayListType`, `unorderedArrayListType`, and `orderedArrayListType` classes that are defined from the textbook, what happens when the following statement is run?

```
unorderedArrayListType intList(-23);
```

Solution

The constructor for `unorderedArrayListType` is included below:

```
arrayListType::arrayListType(int size)
{
    if (size <= 0)
    {
        cout << "The array size must be positive. Creating "
             << "an array of the size 100." << endl;
        maxSize = 100;
    }
    else
    {
        maxSize = size;
    }
    length = 0;
    list = new int[maxSize];
}
```

According to this constructor, if the integer parameter `size` is nonpositive, we print a warning to the console and set the array size to the default value 100.

Problem 4

Unordered sets are a collection of elements with no repeats and no order. If we were to extend the `unorderedArrayListType` to design a `unorderedSetType`, what methods from the `unorderedArrayListType` would have to be redefined?

Solution

The following is a list of methods found in the `unorderedArrayListType` class:

- `void insertAt(int location, int insertItem);`
- `void insertEnd(int insertItem);`

- `void replaceAt(int location, int repItem);`
- `int seqSearch(int searchItem) const;`
- `void remove(int removeItem);`

If we were to develop an `unorderedSetType` class, there would be no notion of ordering, so we would not be able to implement anything like `insertAt()`. Additionally, there are no repeat elements, so `remove()` would not need to be adapted to remove *all* instances of the element in question. Lastly, `replaceAt()` would have to be modified, since we cannot insert elements into a list at *any* index. Instead, we would have to first remove the old element from the list, then add the new element at the end of the list. Essentially, the implementation would be:

```
void unorderedSetType::replace(int oldItem, int newItem)
{
    remove(oldItem);
    insertEnd(newItem);
}
```