Homework 3

● Graded

Student

Giancarlos Marte

Total Points

92 / 108 pts

Question 1

SequentialSearchST

4 / 5 pts

- **– 0 pts** Correct
- **– 4.5 pts** not correct
- ✔ **– 1 pt** not complete
- **– 2 pts** not complete
- **– 4 pts** not complete
- **– 5 pts** CHEATING!

Question 2

Understanding inner classes

18 / 20 pts

- **– 0 pts** Correct
- **– 20 pts** Not answered
- **– 5 pts** part a not answered
- **– 5 pts** part b not answered
- **– 5 pts** part c not answered
- **– 5 pts** part d not answered
- ✔ **– 2 pts** part a incomplete/wrong
- **– 2 pts** part b incomplete/wrong
- **– 2 pts** part c incomplete/wrong
- **– 2 pts** part d incomplete/wrong
- **– 20 pts** CHEATING!

Question 3

Wrapping PriorityQueue

14 / 15 pts

- **– 0 pts** Correct
- **– 15 pts** Not answered
- **– 3 pts** Constructor implemented incorrectly
- **– 1 pt** Comparable is not dropped
- **– 10 pts** MinPQ not implemented
- **– 5 pts** TestMinPQ not implemented
- **– 3 pts** Test Implemented incorrectly
- ✔ **– 1 pt** Minor syntax issues
- **– 15 pts** CHEATING!

Question 4

Practice on Hash Tables

10 / 10 pts

- ✔ **– 0 pts** Correct
- **– 10 pts** Not answered
- **– 2 pts** a. 2 items not correct
- **– 1 pt** a. 1 item not correct
- **– 2 pts** b. 2 items not correct
- **– 1 pt** b. 1 item not correct
- **– 5 pts** b not answered
- **– 5 pts** a not answered
- **– 10 pts** CHEATING!

Question 5

Set App: Resume Scorer

9 / 13 pts

- **– 0 pts** Correct
- **– 2 pts** part a not complete
- **– 5 pts** part a
- **– 8 pts** part b
- ✔ **– 4 pts** part b not complete
- **– 2 pts** part b not complete
- **– 4 pts** part a not complete
- **– 13 pts** not answered
- **– 3 pts** part b not complete
- **– 13 pts** CHEATING!

Question 6

JDK Sorting

17 / 21 pts

- **– 0 pts** Correct
- **– 3 pts** part a
- **– 3 pts** part b
- **– 5 pts** part c
- **– 5 pts** part d
- **– 5 pts** part e
- **– 1 pt** part c not complete
- **– 2 pts** part c not complete
- ✔ **– 4 pts** part c not complete
- **– 2 pts** part d not complete
- **– 2 pts** part e not complete
- **– 4 pts** part e not complete
- **– 21 pts** not answered
- **– 21 pts** CHEATING!

Question 7

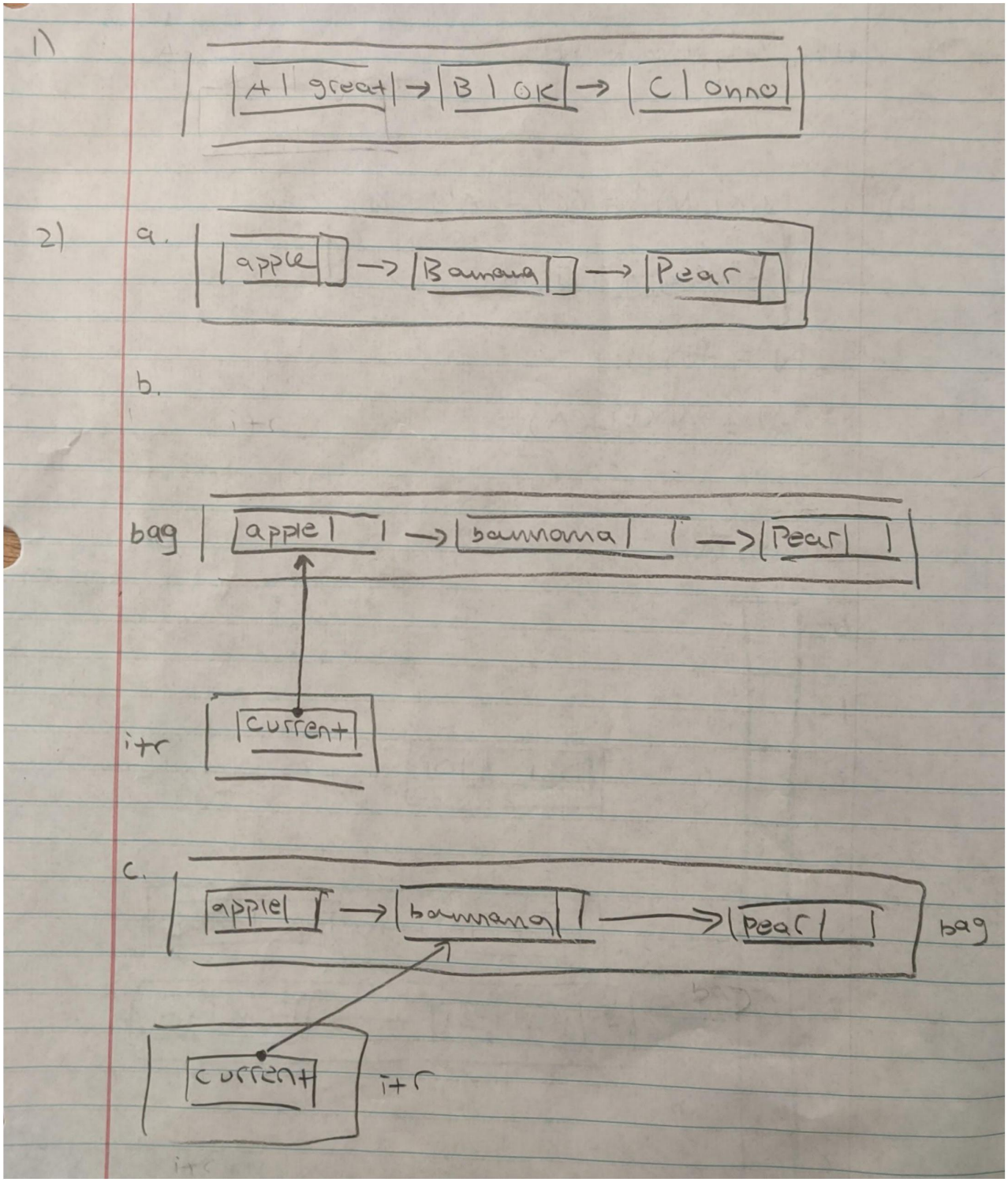
Become an expert on interfaces

20 / 24 pts

- **– 0 pts** Correct
- **– 24 pts** Not answered
- **– 2 pts** 1 item wrong
- ✔ **– 4 pts** 2 items wrong
- **– 6 pts** 3 items wrong
- **– 8 pts** 4 items wrong
- **– 10 pts** 5 items wrong
- **– 24 pts** CHEATING!

Questions assigned to the following page: 1 and 2

Giancarlos Marte
3/21/21
Homework 3



Question assigned to the following page: 2

2d. Yes they can have two iterators existing at the same time. Each iterator can have references to the bag elements without interference with one another since all three objects are completely separate from one another.

Question assigned to the following page: 3

3)

```
import java.util.Comparator;
import java.util.PriorityQueue;

public class MinPQ<Key> {
   private PriorityQueue<Key> pq;

   public MinPQ() {
        pq = new PriorityQueue<>();
   }

   public MinPQ(int capacity) {
        pq = new PriorityQueue<>(capacity);
   }

   public MinPQ(int capacity, Comparator<Key>
comparator) {
        pq = new PriorityQueue<>(capacity,
comparator);
   }

   public void insert(Key key) {
        pq.add(key);
   }

   public Key min() {
        Key[] keys = (Key[]) pq.toArray();
        return keys[keys.length - 1];
   }
```

Question assigned to the following page: 3

```
    public Key delMin() {
        Key[] keys = (Key[]) pq.toArray();
        Key k = keys[keys.length - 1];
        pq.remove(k);
        return k;
    }

    public boolean isEmpty() {
        return pq.size() == 0;
    }

    public int size() {
        return pq.size();
    }
}
```

```
public class TestMinPQ {
    public static void main( String [ ] args ) {
        MinPQ<String> minPQ = new MinPQ<>();
        minPQ.insert("hello");
        minPQ.insert("apple");
        minPQ.insert("zebra");
        String first = minPQ.delMin();
        String sec = minPQ.delMin();
        String third = minPQ.delMin();
        System.out.printf("%s %s %s", first, sec,
third);
    }
}
```

Question assigned to the following page: 4

4)

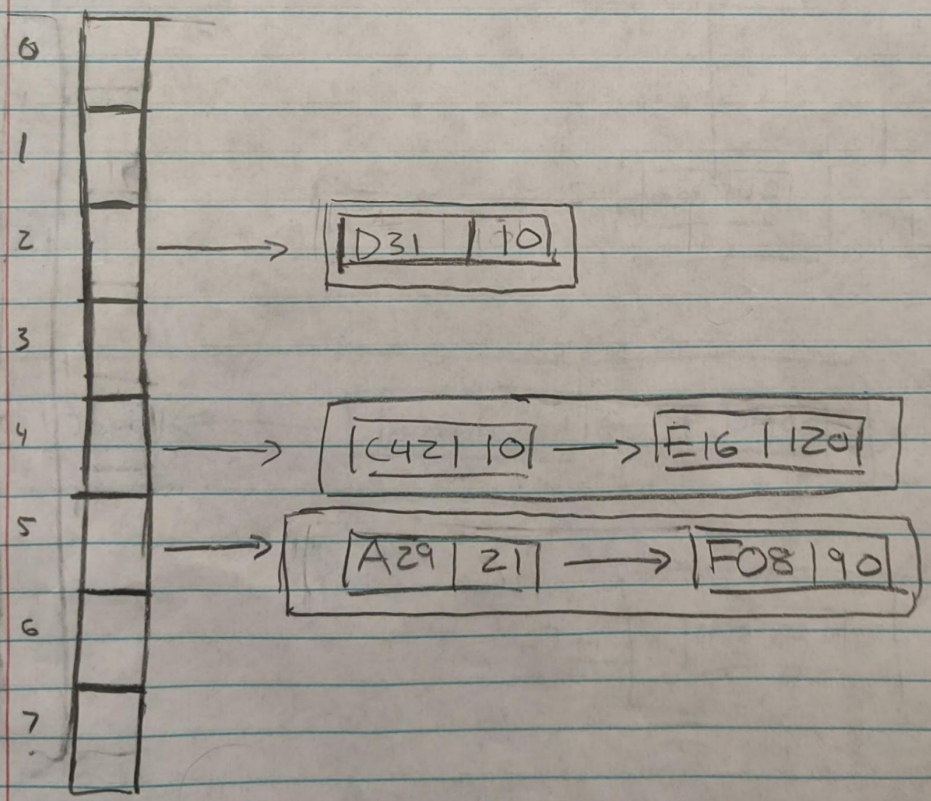
4) a. h(L,N) = ((L - 'A') + N) % 8

L = ASCII (letter)

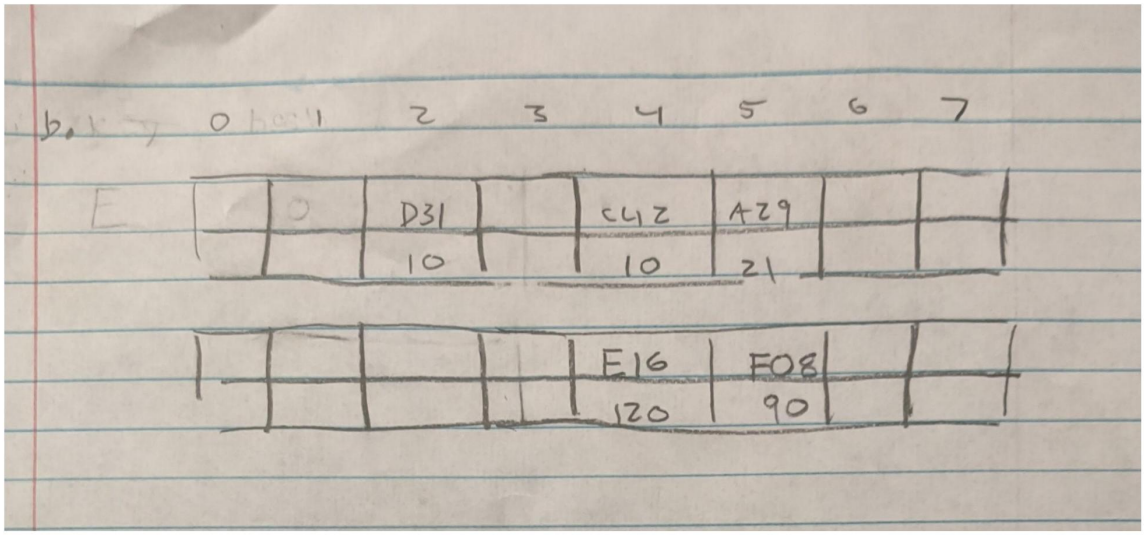
N = #

'A' = ASCII(A)

hash Table



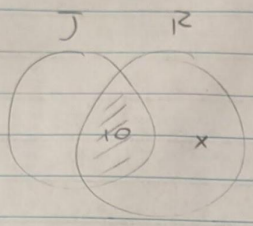
Question assigned to the following page: 4


b. 0 1 2 3 4 5 6 7
D31
10
C42
10
A29
21
E16
120
F08
90

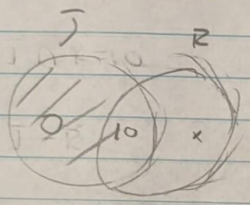
Question assigned to the following page: 5

5)

5) a. $J = 10$

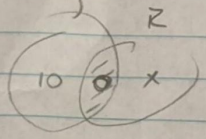


$J = 10 \qquad R = 10 + x$



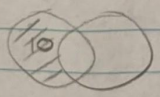
J - R → all in J, but not in R

$J - R = 0$



J intersect R = 0

$J - R = 10$



J intersect R = 10

$10 - 0 = 10$

The maximum score would be 10

$0 - 10 = -10$

The minimum score would be −10

Question assigned to the following page: 5

b.

```
public class ResumeScorer {
    private Set<String> J;
    private Set<String> R;
    private String jFile;
    private String rFile;
    public int score;

    public ResumeScorer(String rFile, String jFile) {
            J = new HashSet<>();
            R = new HashSet<>();
            score = 0;
            readFiles(rFile, jFile);
            calculateScore();
    }

    private void readFiles(String rFile, String jFile) {
            File rf = new File(rFile);
            File jf = new File(jFile);
            Scanner scanR = new Scanner(rf);
            Scanner scanJ = new Scanner(jf);
            while (scanR.hasNext()) {
                    String word = scanR.next();
                    R.add(word);
            }
            while (scanJ.hasNext()) {
                    String word = scanJ.next();
                    J.add(word);
            }
    }

    private void calculateScore() {
            Set<String> intersect = new HashSet<String>(J);
            intersect.retainAll(R);
            Set<String> diff = new HashSet<String>(J);
            diff.removeAll(R);
            score = intersect.size() - diff.size();
            System.out.println("The score is: " + score);
    }
}
```

Question assigned to the following page: 6

6)
a. Arrays.sort(names);

b. Arrays.sort(numbers);

c. Their natural order is basically the points with the greatest y and x coordinates to the ones with the least.

d.
```
Comparator<Point2D> radius = Comparator.comparing(Point2D::r);
Arrays.sort(points, radius);
Arrays.sort(points, points.rOrder());
```

e.
```
List<Account> accounts = new LinkedList<Account>();
accounts.add(new BankAccount("Joe", 100, 100));
accounts.add(new BankAccount("Sue", 101, 200));
Comparator<BankAccount> balance = Comparator.comparing(BankAccount::getBalance());
Collections.sort(accounts, balance);
```

Question assigned to the following page: 7

7)

a. An interface can declare instance variables.	True
b. Any method in an interface must be public.	True
c. An interface can have no methods at all.	True
d. An interface can extend another interface.	True
e. An interface can declare constructors.	False
f. A class may extend more than one class.	True
g. A class may implement more than one interface.	True
h. A class may extend one class and implement one interface.	True
i. An interface may implement some of its methods.Ignore the new Java 8 features "default methods for interfaces" and "static methods in interfaces" here.	False
j. All methods in an interface must have a void return type.	False
k. Object is an interface.	False
l. Comparable is an interface.	True