Homework 5

Note: You may collaborate on the assignment. If you do collaborate on the assignment, list your collaborators. All duplicate assignments without collaborators listed will be flagged for plagiarism. Additionally, presenting others work – including a Chegg expert answer – as your own work *is* a violation of the academic dishonesty policy.

This homework assignment is designed to prepare you for the in class quiz.

1 Input Space Partitioning

1.1 Evaluating Characteristics

Correct the following input space partitioning until you feel that the characteristics and blocks satisfy the disjointness and completeness properties. You are free to change the blocks and make new characteristics in your attempts to fix the partitions.

Input Partition 1: House

- Characteristics 1: Location
  - Blocks: Urban, Rural, Suburbs
- Characteristics 2: Material
  - Blocks: Wood, Brick
- Characteristics 3: Size
  - Blocks: 1 bedroom, 2 bedrooms, 3+ bedrooms, town-house

Input Partition 2: Finding min element in a list

- Characteristics 1: Location in list
  - Blocks: Min elements is the first element, min element is the last element, min element is neither the first nor last element.

Input Partition 3: Withdraw money from a bank account

- Characteristics 1: Withdraw Amount
  - Blocks: Half the amount in the account, the amount in the account, double the amount in the account

1.2 Making the Characteristics and Tests - Binary Search Tree

Given the following interface:

```
public BinarySearchTree ();
public void add (Object X);
public boolean remove(Object x);
public void inOrderTraversal ();
public int getHeight ();
public void clearTree ();
```

(a) List all of the input variables, including the state variables (reminder: our `GenericStack` had a state variable aka the stack itself).

(b) Define characteristics of the input variables. Make sure you cover all input variables.

(c) Partition the characteristics into blocks (make sure they are disjoint and complete). Designate one block in each partition as the "Base" block.

(d) Define values for each block.

(e) Define a test set that satisfies Base Choice Coverage (BCC). Write your tests with the values from the previous step. Be sure to include the test oracles (test oracle is telling me the expected output).

(f) Define a test set that satisfies Pair Wise Coverage (PWC). Write your tests with the values from the previous step. Be sure to include the test oracles (test oracle is telling me the expected output).

1.3 Making the Characteristics and Tests - HashMap

Given the following interface:

```
public HashMap<K,V> ();
public void put(K key, V value);
public Object get(Object key)
public boolean containsKey(Object key);
public boolean  containsValue(Object value);
public void putAll(Map<? extends K,? extends V> m);
```

(a) List all of the input variables, including the state variables.

(b) Define characteristics of the input variables. Make sure you cover all input variables.

(c) Partition the characteristics into blocks (make sure they are disjoint and complete). Designate one block in each partition as the "Base" block.

(d) Define values for each block.

(e) Define a test set that satisfies Base Choice Coverage (BCC). Write your tests with the values from the previous step. Be sure to include the test oracles (test oracle is telling me the expected output).

(f) Define a test set that satisfies Pair Wise Coverage (PWC). Write your tests with the values from the previous step. Be sure to include the test oracles (test oracle is telling me the expected output).