Name: Zhenzhen Su

HW 11

Problem 1: optimize sieve method

Code:

```
public static ArrayList<Integer> sieveOfEratosthenes(int max) {

    ArrayList<Integer> number = new ArrayList<Integer>();
    int i = 1;
    number.add(2);
    while(2*i+1 <= max){
        number.add(2*i+1);
        i++;
    }

    //start with 2 , check everyone after that
    int j = 0;
    while(Math.pow(number.get(j),2) < max && j < number.size()){
        //check other elements
        int h = j+1;

        while(h < number.size()) {
            if (number.get(h) % number.get(j) == 0) {
                number.remove(h);
            }else{
                h++;
            }
        }
        j ++;

    }
    return number;
}


public class Main {

    public static void main(String[] args) {
        System.out.println(sieveOfEratosthenes(30));
    }
}
```

Console:

[2, 3, 5, 7, 11, 13, 17, 19, 23, 29]

Problem 7: Find common elements in both lists

Code:

```
public static int countCommon(ArrayList<Integer> list1, ArrayList<Integer> list2){
    Set<Integer> set1 = new HashSet<Integer>(list1);
```

```
    Set<Integer> set2 = new HashSet<Integer>(list2);

    HashSet<Integer> commonSet = new HashSet<Integer>(set1);
    commonSet.retainAll(set2);
    return commonSet.size();
}


public class Main {

  public static void main(String[] args) {

      ArrayList<Integer> list1 = new ArrayList<Integer>();
      list1.add(3);
      list1.add(7);
      list1.add(3);
      list1.add(-1);
      list1.add(2);
      list1.add(3);
      list1.add(7);
      list1.add(2);
      list1.add(15);
      list1.add(15);

      ArrayList<Integer> list2 = new ArrayList<Integer>();
      list2.add(-5);
      list2.add(15);
      list2.add(2);
      list2.add(-1);
      list2.add(7);
      list2.add(15);
      list2.add(36);

      System.out.println(countCommon(list1, list2)) ;
  }
}
```

Console:

4

Problem 11: Find elements that exists in only each one list

Code:

```
public class Main {

    public static void main(String[] args) {
       Set<Integer> set1 = new HashSet<Integer>();
       set1.add(1);
       set1.add(4);
       set1.add(7);
       set1.add(9);

       Set<Integer> set2 = new HashSet<Integer>();
       set2.add(2);
       set2.add(4);
       set2.add(5);
```

```
        set2.add(6);
        set2.add(7);

        System.out.println(SymmetricSetDifference(set1,set2));
        }
}


public static Set<Integer> SymmetricSetDifference(Set<Integer> set1, Set<Integer>
set2){
    Set<Integer> diff1 = new HashSet<Integer>(set1);
    Set<Integer> diff2 = new HashSet<Integer>(set2);

    diff1.removeAll(set2);
    diff2.removeAll(set1);
    diff1.addAll(diff2);
    return diff1;
}
```

Console:

[1, 2, 5, 6, 9]

Problem 14: Find key-value pairs that exist in both maps.

Code:

```
//14
public static HashMap<String,Integer> intersect(HashMap<String,Integer> map1,
HashMap<String, Integer> map2){
    Set<String> map1Key = map1.keySet();
    Set<String> map2Key = map2.keySet();

    HashMap<String,Integer> common = new HashMap<String, Integer>();
    for(String key1 : map1Key){
        for(String key2 : map2Key){
            if(key1 == key2 && map1.get(key1) == map2.get(key2)){
                common.put(key1,map1.get(key1));
            }
        }
    }
    return common;
}


public class Main {

    public static void main(String[] args) {
        HashMap<String,Integer> map1 = new HashMap<String, Integer>();
        map1.put("Janet",87);
        map1.put("Logan",62);
        map1.put("Whitaker",46);
        map1.put("Alyssa",100);
        map1.put("Stefanie",80);
        map1.put("Jeff",88);
        map1.put("Kim",52);
        map1.put("Sylvia",95);
```

```
        HashMap<String,Integer> map2 = new HashMap<String, Integer>();
        map2.put("Logan",62);
        map2.put("Whitaker",52);
        map2.put("Stefanie",80);
        map2.put("Jeff",88);
        map2.put("Kim",52);
        map2.put("Brian",60);
        map2.put("Lisa",83);
        map2.put("Sylvia",87);
        System.out.println(intersect(map1,map2));

    }
}
```

Console:

{Logan=62, Stefanie=80, Jeff=88, Kim=52}

Problem 17: Reverse key value pairs

```
Code:
public static HashMap<String,HashSet<String>> reverse(HashMap<String, String> map1){
    HashMap reverMap = new HashMap<String, HashSet<String>>();
    for(Map.Entry entry : map1.entrySet()){
        if(reverMap.containsKey(entry.getValue())) {
            Set set = new HashSet<String>();
            set.addAll((Set)reverMap.get(entry.getValue()));
            set.add(entry.getKey());
            reverMap.put(entry.getValue(), set);

        }else{
            Set valueSet = new HashSet<String>();
            valueSet.add(entry.getKey());
            reverMap.put(entry.getValue(), valueSet);
        }

    }


    return reverMap;
}


public class Main {

    public static void main(String[] args) {
        HashMap<String, String> map = new HashMap<String, String>();
        map.put("42", "Marty");
        map.put("81", "Sue");
        map.put("17", "Ed");
        map.put("31", "Dave");
        map.put("56", "Ed");
        map.put("3", "Marty");
        map.put("29", "Ed");

        System.out.println(reverse(map));
    }
```

}

Console:

{Sue=[81], Marty=[3, 42], Dave=[31], Ed=[56, 17, 29]}