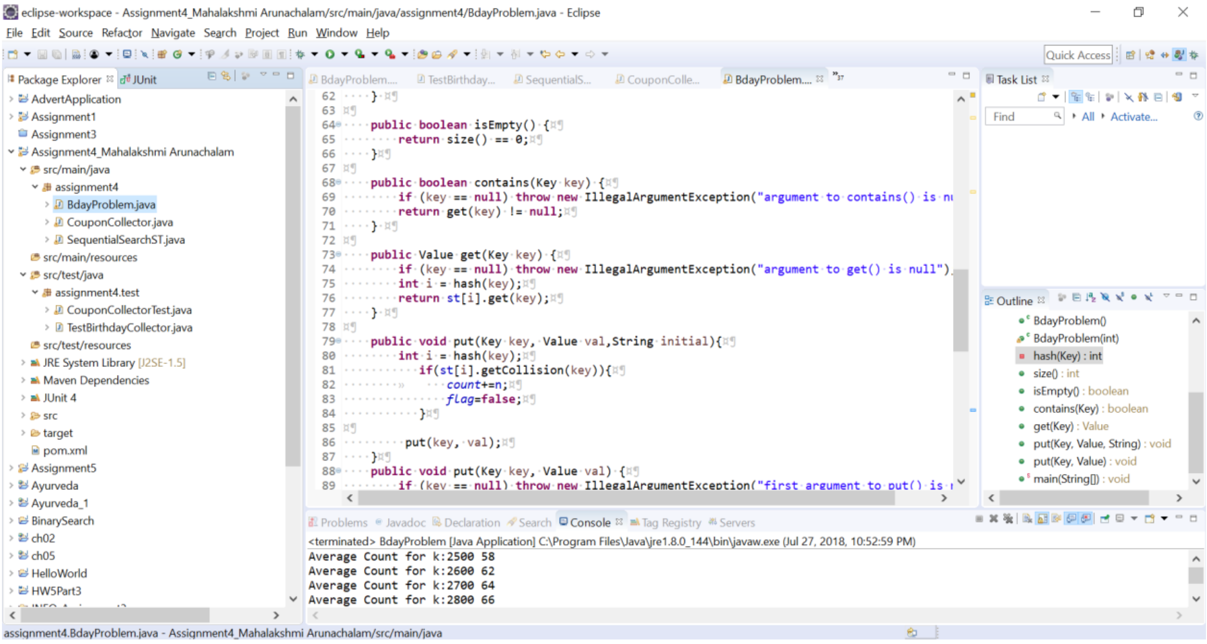
1. Birthday Problem : BdayProblem.java



```
    }
 
    public boolean isEmpty() {
        return size() == 0;
    }
 
    public boolean contains(Key key) {
        if (key == null) throw new IllegalArgumentException("argument to contains() is nu
        return get(key) != null;
    }
 
    public Value get(Key key) {
        if (key == null) throw new IllegalArgumentException("argument to get() is null").
        int i = hash(key);
        return st[i].get(key);
    }
 
    public void put(Key key, Value val,String initial){
        int i = hash(key);
            if(st[i].getCollision(key)){
                count+=n;
                flag=false;
            }
 
         put(key, val);
    }
    public void put(Key key, Value val) {
        if (key == null) throw new IllegalArgumentException("first argument to put() is n
```

```
<terminated> BdayProblem [Java Application] C:\Program Files\Java\jre1.8.0_144\bin\javaw.exe (Jul 27, 2018, 10:52:59 PM)
Average Count for k:2500 58
Average Count for k:2600 62
Average Count for k:2700 64
Average Count for k:2800 66
```

Output:

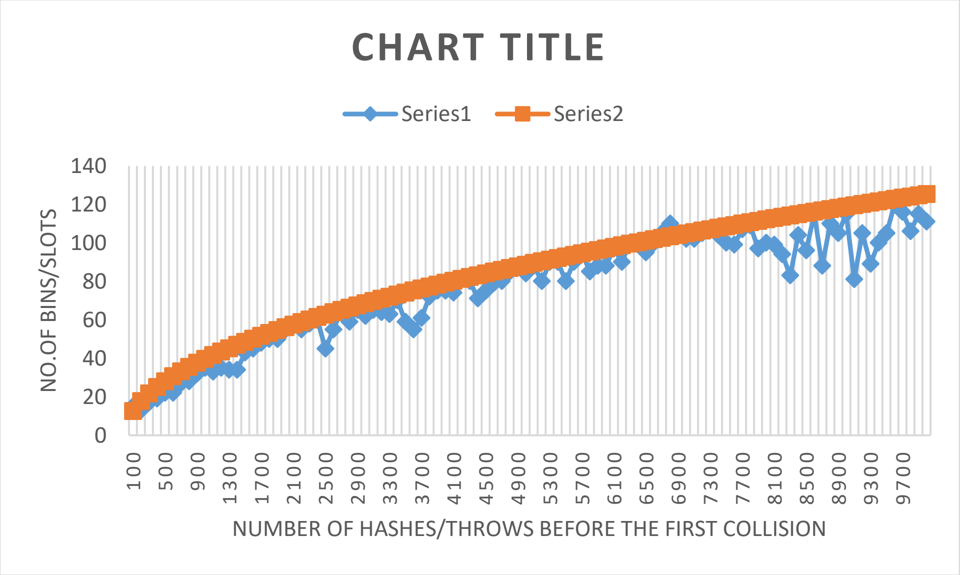
X axis : Number of Bins/slots

Y axis : number of hashes/throws before the first collision

Values in orange are theoretical values

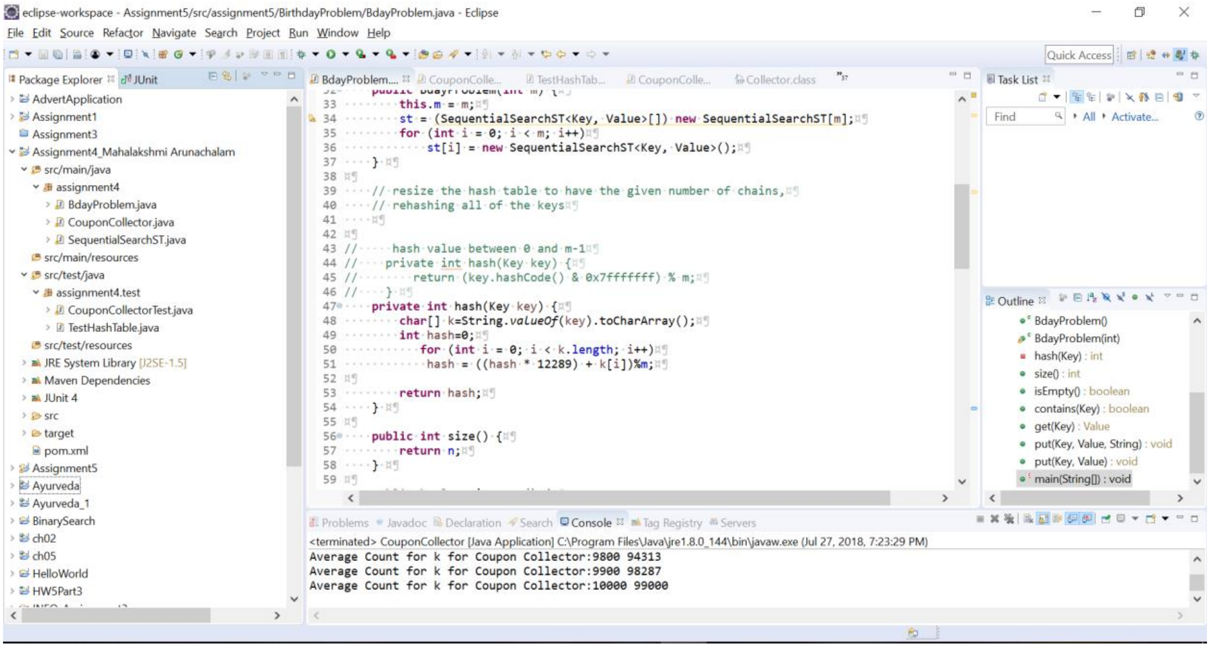
Values in blue are practical values

The graph show the relation between practical and theoretical values which are approximately equal.



2. Coupon Collector Problem: CouponCollector.java

Output:



```
    public BdayProblem(int m) {
        this.m = m;
        st = (SequentialSearchST<Key, Value>[]) new SequentialSearchST[m];
        for (int i = 0; i < m; i++)
            st[i] = new SequentialSearchST<Key, Value>();
    }

    // resize the hash table to have the given number of chains,
    // rehashing all of the keys
    

//     hash value between 0 and m-1
//    private int hash(Key key) {
//        return (key.hashCode() & 0x7fffffff) % m;
//    }
    private int hash(Key key) {
        char[] k=String.valueOf(key).toCharArray();
        int hash=0;
           for (int i = 0; i < k.length; i++)
            hash = ((hash * 12289) + k[i])%m;

        return hash;
    }

    public int size() {
        return n;
    }

```

```
<terminated> CouponCollector [Java Application] C:\Program Files\Java\jre1.8.0_144\bin\javaw.exe (Jul 27, 2018, 7:23:29 PM)
Average Count for k for Coupon Collector:9800 94313
Average Count for k for Coupon Collector:9900 98287
Average Count for k for Coupon Collector:10000 99000
```

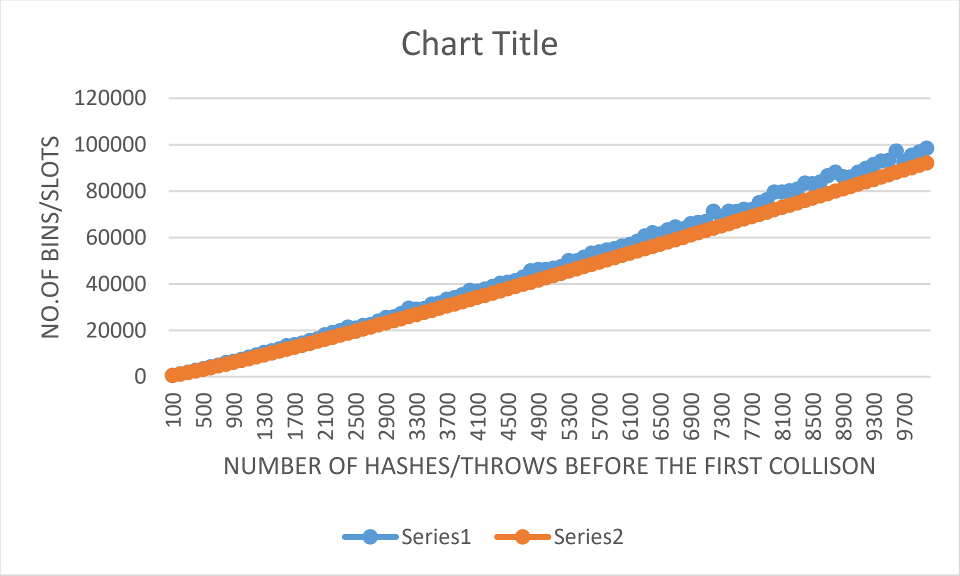
X axis: Number of Bins/slots

Y axis: number of hashes/throws before the first collision

Values in orange are theoretical values

Values in blue are practical values

The graph show the relation between practical and theoretical values which are approximately equal.



Hence practical values were verified against the theoretical values.