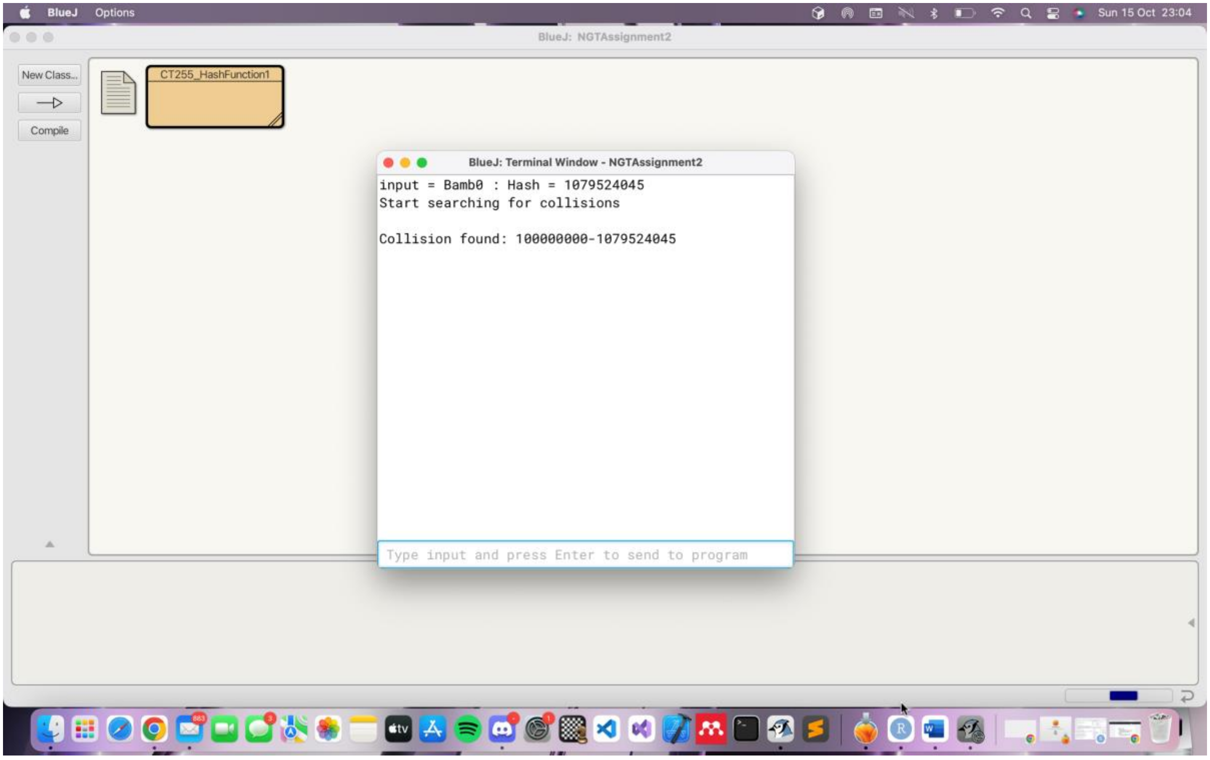
NGT Assignment 2 Hash Functions
Problem 2.

```
/**
 *
 * @author Michael Schukat
 */
public class CT255_HashFunction1 {

    public static void main(String[] args) {
        int res = 0;

        if (args != null && args.length > 0) { // Check for <input> value
            res = hashF1(args[0]); // call hash function with <input>
            if (res < 0) { // Error
                System.out.println("Error: <input> must be 1 to 64 characters long.");
            }
            else {
                System.out.println("input = " + args[0] + " : Hash = " + res);
                System.out.println("Start searching for collisions");
                int collisionHash = 0; //variable that holds the hash code of the hash collision
                int checkHash = 0; //a number that is converted to a string and passed to the
hashf1 function to look for a hash collision
                int num = 0; //just a variable that is used to count and stop the while loop when it
reaches 10 hash collisions

                while (num <= 10 || checkHash < Integer.MAX_VALUE) { //this while loop stops
when it finds 10 hash collisions, and if it cant find 10 it will stop when it has checked every
number up as far as the max integer value
                String collisionString = Integer.toString(checkHash); //the checkHash number is
converted to a string and stored in collisionString
                collisionHash = hashF1(collisionString); //the string is passed to the hashf1 function
                if(collisionHash == res) {
                    System.out.println("\nCollision found: " + Integer.toString(checkHash) + "-" +
collisionHash);
                    num++;//if a collision is found its printed to the screen and the counter variable
increases
                }
                checkHash++; //checkHash is increased so the next iteration starts
            }
            }
            }

        else { // No <input>
            System.out.println("Use: CT255_HashFunction1 <Input>");
        }
    }
```

```
  private static int hashF1(String s){
    int ret = -1, i;
    int[] hashA = new int[]{1, 1, 1, 1};

    String filler, sIn;

    filler = new
String("ABCDEFGHABCDEFGHABCDEFGHABCDEFGHABCDEFGHABCDEFGHABCDEFGHABCDEF
GH");

    if ((s.length() > 64) || (s.length() < 1)) { // String does not have required length
      ret = -1;
    }
    else {
      sIn = s + filler; // Add characters, now have "<input>HABCDEF..."
      sIn = sIn.substring(0, 64); // // Limit string to first 64 characters
      // System.out.println(sIn); // FYI
      for (i = 0; i < sIn.length(); i++){
        char byPos = sIn.charAt(i); // get i'th character
        hashA[0] += (byPos * 17); // Note: A += B means A = A + B
        hashA[1] += (byPos * 31);
        hashA[2] += (byPos * 101);
        hashA[3] += (byPos * 79);
      }

      hashA[0] %= 255;  // % is the modulus operation, i.e. division with rest
      hashA[1] %= 255;
      hashA[2] %= 255;
      hashA[3] %= 255;

      ret = hashA[0] + (hashA[1] * 256) + (hashA[2] * 256 * 256) + (hashA[3] * 256 * 256 *
256);
      if (ret < 0) ret *= -1;
    }
    return ret;
  }
```

```
}
```



```
input = Bamb0 : Hash = 1079524045
Start searching for collisions

Collision found: 100000000-1079524045
```