Ryan Bajolllari
CPE 593 Assignment 5

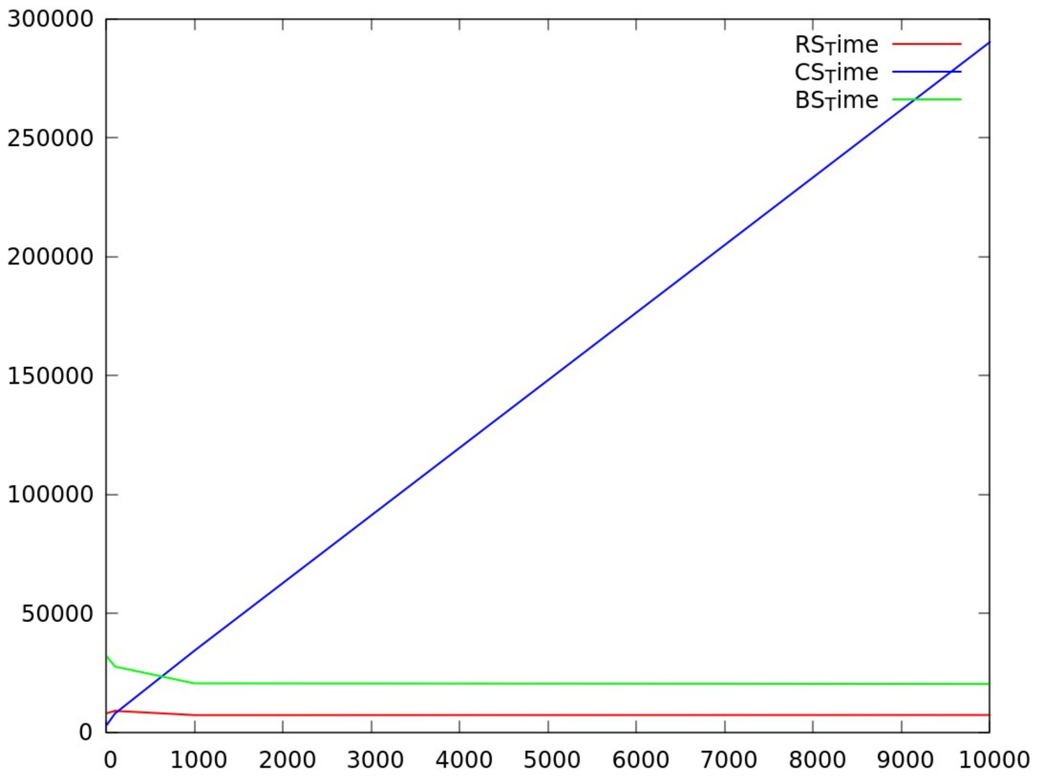
.dat Fle:

```
#n        RS_T      CS_T      BS_T      RS_L#        CS_L#        BS_L#        RS_UL#       CS_UL#       BS_UL#

10        7961      3101      31757     13(0.3)      7(0.3)       7(0.3)       7(0.1)       13(0.1)      13(0.1)
100       8922      7792      27564     13(0.12)     13(0.12)     13(0.12)     2(0.05)      12(0.05)     12(0.05)
1000      7163      34221     20494     12(0.085)    1(0.085)     1(0.085)     1(0.068)     4(0.068)     4(0.068)
10000     7197      290198    20245     12(0.0805)   10(0.0805)   10(0.0805)   1(0.0714)    7(0.0714)    7(0.0714)
```

I decided to use four different drawing amounts (n = 10, 100, 1000, 10000). The first three columns are the running times (in nanoseconds) for Random Selection (RS_T), Counting Sort (CS_T), and Bucket Sort (BS_T) for each value of n. The next three columns are the lucky numbers found in each method with the probability in parenthesis next to the respective number. The next three columns are the unlucky ones. As you can see the probabilities of getting an unlucky and a lucky number get closer as n gets higher, which is the result of the random distribution acting more uniform (as it should theoretically) with a higher sample size.

Plot:



Observations:
When calling the counting sort function I decided to make the max value parameter (int k) n, since n is the worst case maximum value of the number of times the same number is drawn. For this reason, the value of n determines the size of the count array created in the function which is why you see a basically linear relationship between the size of n and how fast count sort is getting the unlucky/lucky numbers. Randomized selection and bucket sort stayed around the same with RS being slightly better since the input array for both methods was constantly size of 13, just with higher values in each index.