CS211 ALGORITHMS & DATA STRUCTURES II

LAB 7

Dr. Phil Maguire

HASHING

Pen and Paper Exercise

	Linear Probing	Quadratic Probing	Double Hashing
444	34	56	55
669	23	45	26
880	55	55	5

Programming Exercise

```
     public static int getHashKey(String word){
//this is the primary hash key function - it should return a number which is
a slot in the hash table
//for words, a good strategy is to raise each character to successive powers
of the alphabet size
//assume that the alphabet is ASCII characters - a total of 256 possibilities
//each successive character value should be raised to successive powers of
256
//the whole thing is added together and then moduloed to create a hash table
index
        int total = 0;
        for(int i=0; i<word.length();i++){


total=total+modMult((int)word.charAt(i),modPow(128,i,size),size);
//raise each letter to successive powers of 128
//this will create a unique value for every possible string because there are
only 127 ASCII characters
         }
         return total%size;
//now 'hash' this unique value down to the size of the hash table
      }
```

```
     public static int getDoubleHashKey(String word){
//this method should be different to the primary hash function
//it should return a different number for words which generated the same
primary hash key value
//for example, you could just add up all of the letters in the word
//the number should be hashed into a particular range which is the range of
the jump size
        int total = 0;
        for(int i=0; i<word.length();i++){

             total=total+(int)word.charAt(i);
//add up all of the letters
         }
//let's say the range of the jump size is up to 113
//but we have to make sure we never return 0 so the range has to be 1 - 113
//the following formula will always return a number from 1 - 113
         int maxjump = 113;
         return (maxjump - (total % maxjump));
     }
```