Assignment 3 Hash/Heap

Dylan Taylor

October 24, 2021

1. What occupancy ratio should you use?

   My hash function was as follows:

   ```
   int hashed = 0;
   int multiplier = 1;
   for(int i = 0; i ¡ word.length(); i++){
   hashed += (int)word[i] * multiplier;
   multiplier +=13;
   }
   hashed = abs(hashed);
   hashed = hashed % SIZE;
   ```

   I tested the following: Array size, collision rate, time to create the hash table, and finally the time it took to sort the hash table with heap sort.

   Array size: 100000000 cr:15% time 43 milli sort: 6 milli
   Array size: 100 cr:98% time 63 milli sort: 3 milli
   Array size: 1000 cr:88% time 45 milli sort: 2 milli
   Array size: 10000 cr:41% time 38 milli sort: 2 milli
   Array size: 100000 cr:16% time 38 milli sort: 2 milli

   My findings show that an array size of 100,000 with my hash function seemed to be the best for collision and for run time.

2. How do you set up a hash table of variable size?

   You would either have to create a linked list which in reality would no longer be a hash table, or you would have to create a new array of a larger size and then copy all the old values over and continue.

3. What hash function?

```
int hashed = 0;
int multiplier = 1;
for(int i = 0; i ¡ word.length(); i++){
hashed += (int)word[i] * multiplier;
multiplier +=13;
}
hashed = abs(hashed);
hashed = hashed % SIZE;
```

4. What if there are collisions in the table?

   I used chaining so I made my hash table an array of linked lists. I found this easier to keep track of and to handle infinite collisions. Double hashing with large amounts of data can lead to errors.

5. Do you need an interface file (a ".h" file) for these functions?

   You do not "need" an interface file, however I used multiple as I found it easier and clearer to read and edit my program.