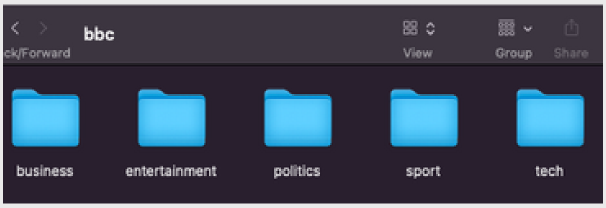
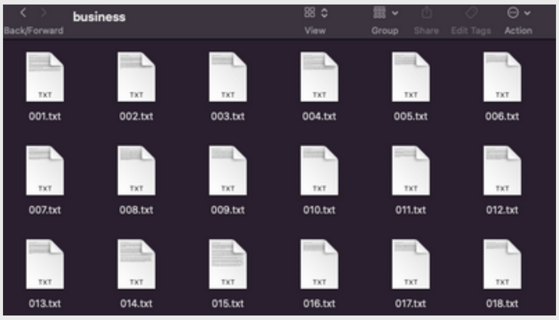
Efficient Hash

The purpose of this project is to store data that contains 2 keys 1 value and we wonder how many repetitions.

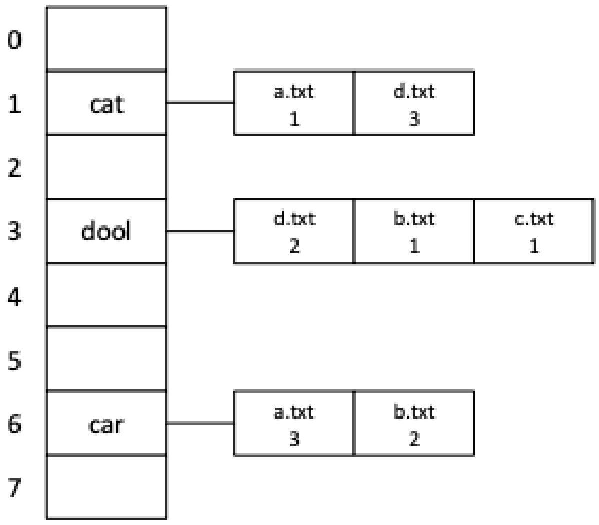
We have BBC documents converted to text and separated according to their categories.





Here, our first goal is to read and store the words one by one. While storing, the category as the 1st key and the name of a text as the 2nd key is taken into account.

While these processes are being carried out, the number of times that word is repeated in that text should be stored somewhere. So we need an additional node and counter, in the class.



The user decides what to use when storing (Simple Summation Function (SSF) or Polynomial Accumulation Function (PAF)) and what to use in collision handling (Linear Probing (LP) or Double Hashing (DH)).

Hash Functions

SSF

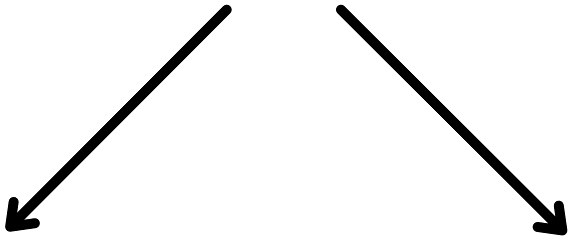
can generate the hash code of a string s with the length n simply by the following formula:

$$h(s) = \sum_{k=0}^{n-1} ch_k$$

PAF

The hash code of a string s can also be generated by using the following polynomial:
h(s) = *ch*0 ∗ *z n*−1 + *ch*1 ∗ *z n*−2 + . . . + *chn*−2 ∗ *z* 1 + *chn*−1 ∗ *z*^ 0
where *ch*0 is the leftmost character of the string, characters are represented as numbers in 1-26 (case insensitive), and n is the length of the string. The constant z is usually a prime number (31, 33, 37, and 41 are particularly good choices for working English words). When the z value is chosen as 31, the string "car" has the following hash value: *h*(car) = 3 ∗ 312 + 1 ∗ 31 + 18 ∗ 1 = 2932
Note: Using this calculation on the long strings will result in numbers that will cause overflow. Use Horner's rule to perform the calculation and apply the modulus operator after computing each expression in Horner's rule.

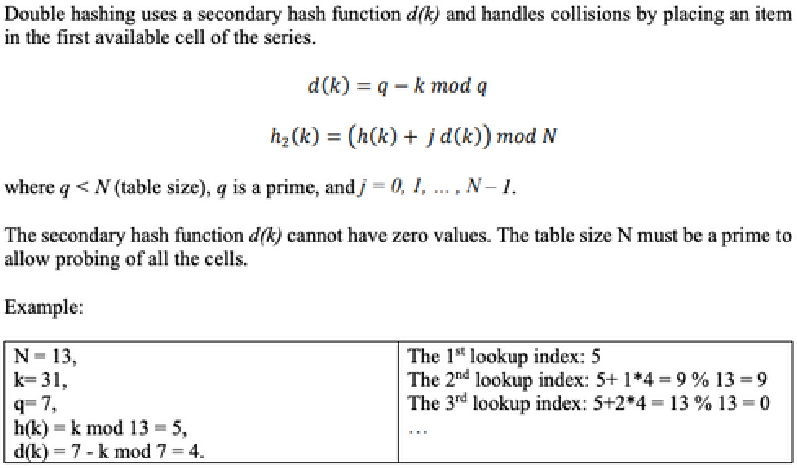
Collision Handling



Linear Probing (LP)

Linear probing handles collisions by placing the colliding item in the next (circularly) available table cell.

Double Hashing (DH)



Double hashing uses a secondary hash function *d(k)* and handles collisions by placing an item in the first available cell of the series.

$$d(k) = q - k \bmod q$$

$$h_2(k) = \big(h(k) + j\, d(k)\big) \bmod N$$

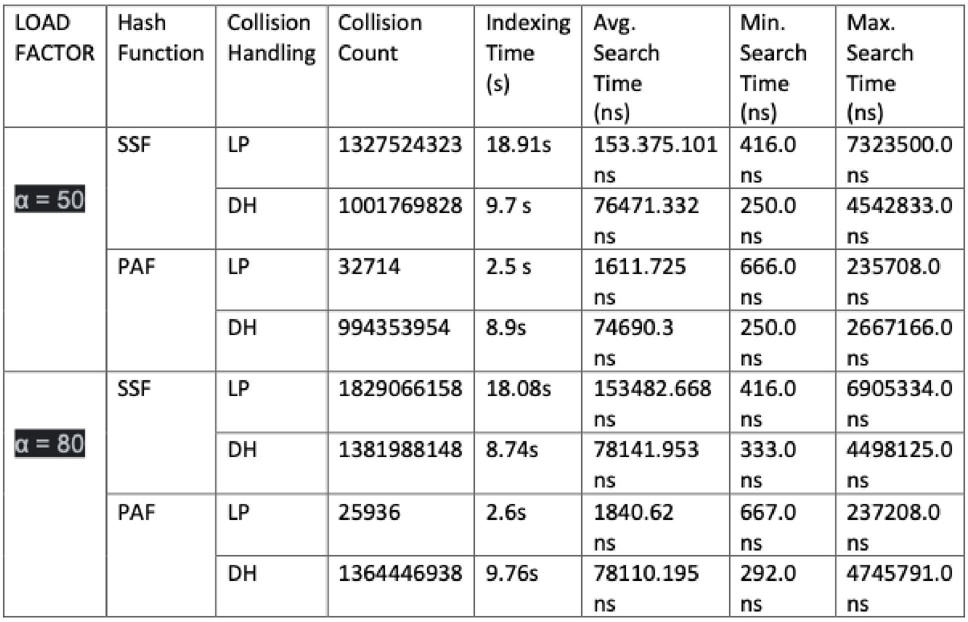
where $q < N$ (table size), q is a prime, and $j = 0, 1, \dots, N-1$.

The secondary hash function *d(k)* cannot have zero values. The table size N must be a prime to allow probing of all the cells.

Example:

N = 13,
k= 31,
q= 7,
h(k) = k mod 13 = 5,
d(k) = 7 - k mod 7 = 4.
	The 1st lookup index: 5
	The 2nd lookup index: 5+ 1*4 = 9 % 13 = 9
	The 3rd lookup index: 5+2*4 = 13 % 13 = 0
	…

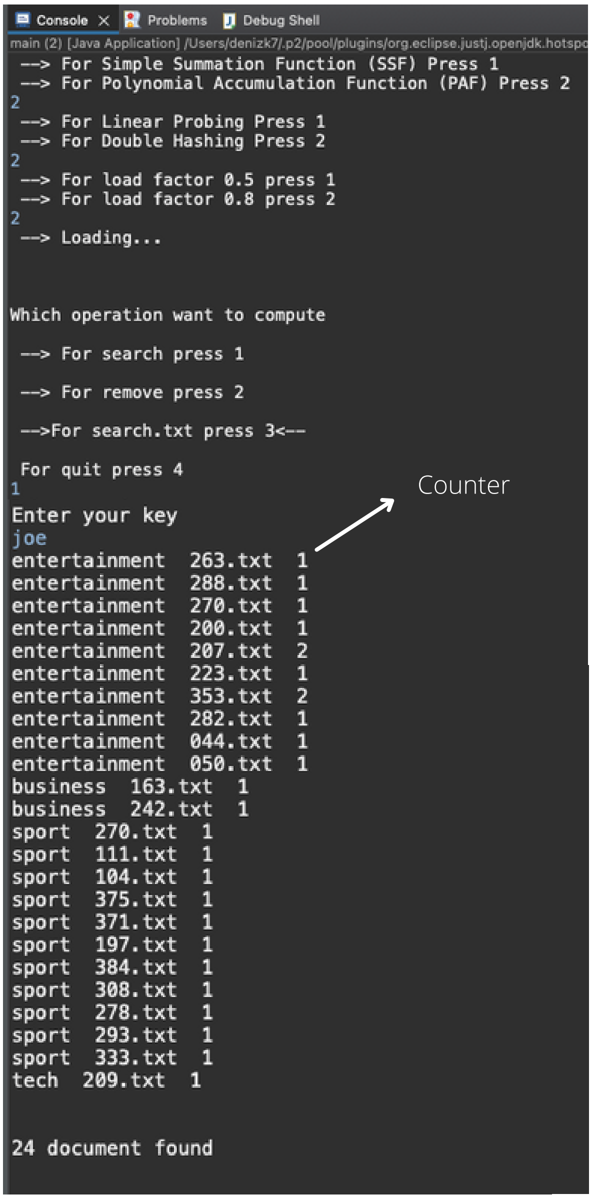
Performance monitoring



LOAD FACTOR	Hash Function	Collision Handling	Collision Count	Indexing Time (s)	Avg. Search Time (ns)	Min. Search Time (ns)	Max. Search Time (ns)
α = 50	SSF	LP	1327524323	18.91s	153.375.101 ns	416.0 ns	7323500.0 ns
		DH	1001769828	9.7 s	76471.332 ns	250.0 ns	4542833.0 ns
	PAF	LP	32714	2.5 s	1611.725 ns	666.0 ns	235708.0 ns
		DH	994353954	8.9s	74690.3 ns	250.0 ns	2667166.0 ns
α = 80	SSF	LP	1829066158	18.08s	153482.668 ns	416.0 ns	6905334.0 ns
		DH	1381988148	8.74s	78141.953 ns	333.0 ns	4498125.0 ns
	PAF	LP	25936	2.6s	1840.62 ns	667.0 ns	237208.0 ns
		DH	1364446938	9.76s	78110.195 ns	292.0 ns	4745791.0 ns

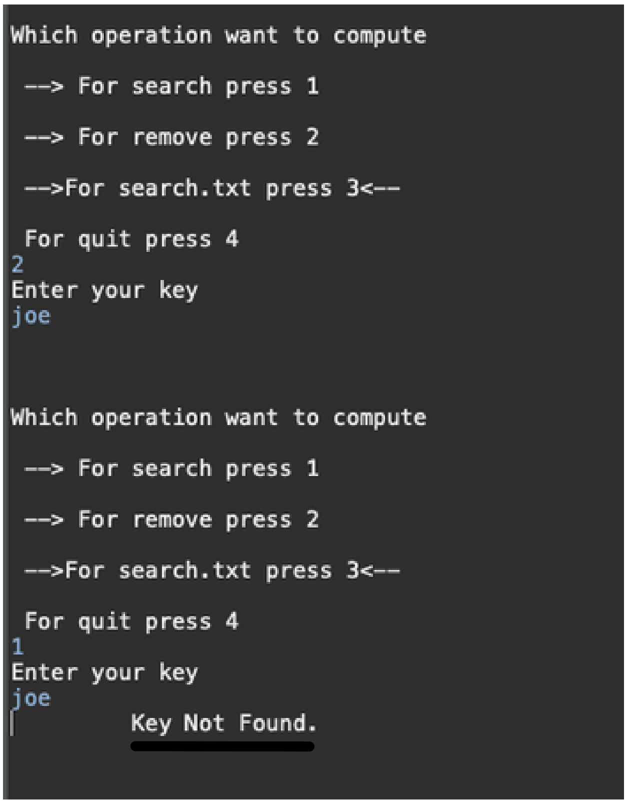
Console Screenshots

Choosing and search



```
main (2) [Java Application] /Users/denizk7/.p2/pool/plugins/org.eclipse.justj.openjdk.hotspo
 --> For Simple Summation Function (SSF) Press 1
 --> For Polynomial Accumulation Function (PAF) Press 2
2
 --> For Linear Probing Press 1
 --> For Double Hashing Press 2
2
 --> For load factor 0.5 press 1
 --> For load factor 0.8 press 2
2
 --> Loading...



Which operation want to compute

 --> For search press 1

 --> For remove press 2

 -->For search.txt press 3<--

 For quit press 4
1
Enter your key
joe
entertainment  263.txt  1
entertainment  288.txt  1
entertainment  270.txt  1
entertainment  200.txt  1
entertainment  207.txt  2
entertainment  223.txt  1
entertainment  353.txt  2
entertainment  282.txt  1
entertainment  044.txt  1
entertainment  050.txt  1
business  163.txt  1
business  242.txt  1
sport  270.txt  1
sport  111.txt  1
sport  104.txt  1
sport  375.txt  1
sport  371.txt  1
sport  197.txt  1
sport  384.txt  1
sport  308.txt  1
sport  278.txt  1
sport  293.txt  1
sport  333.txt  1
tech  209.txt  1


24 document found
```

Delete and search again



```
Which operation want to compute

 --> For search press 1

 --> For remove press 2

 -->For search.txt press 3<--

 For quit press 4
2
Enter your key
joe



Which operation want to compute

 --> For search press 1

 --> For remove press 2

 -->For search.txt press 3<--

 For quit press 4
1
Enter your key
joe
          Key Not Found.
```

By: **Deniz Küçükkara**