Student_ID: 1805062

OFFLINE_HASHING_REPORT

Hash Functions Used:

1. **Hash Function 1** :

```
int hash1(String key, int m) {
    long p = 137;
    int h = 0;
    for (int i = 0; i < key.length(); i++) {
        char c = key.charAt(i);
        h = (int) ((h * p + c) % m);
    }
    return h;
}
```

2. Hash Function 2 :

```
int hash2(String key, int m) {
    long p = 0x811C9DC5L % m;
    int h = 0;
    for (int i = 0; i < key.length(); i++) {
        char c = key.charAt(i);
        h = (int) ((h * p) ^ c) % m;
    }
    return h;
}
```

3. Aux Function:

```
int auxHash(String key, int m) {
    long h = 0, a = 2, b = 3;
    long t;
    for (int i = 0; i < key.length(); i++) {
        char c = key.charAt(i);
        h =  (h * a + c) % m;
        a = (a + b) % m;
        t = a;
        a = b;
        b = t;
    }
    return (int)h;
}
```

Performance Analysis:

Table-1: Performance of some collision resolution with table size, N = 15013

	Hash1		Hash2	
	Number of Collisions	Average Probes	Number if Collisions	Average Probes
Chaining Method	3337	1.321	3366	1.347
Double Hashing	6374	1.643	6294	1.593
Custom Probing	6197	1.614	6816	1.642

Table-2: Performance of some collision resolution with table size, N = 20011

	Hash1		Hash2	
	Number of Collisions	Average Probes	Number if Collisions	Average Probes
Chaining Method	2547	1.25	2497	1.24
Double Hashing	3858	1.346	3801	1.401
Custom Probing	3796	1.376	3781	1.361

Table-3: Performance of some collision resolution with table size, N = 30013

	Hash1		Hash2	
	Number of Collisions	Average Probes	Number if Collisions	Average Probes
Chaining Method	1726	1.155	1691	1.168
Double Hashing	2219	1.215	2119	1.228
Custom Probing	2109	1.19	2182	1.202