LAB 5: Hash and Searching

Họ và tên: Hồ Minh Đăng
MSSV: 22127050

Exercise 1

Giải thích về hàm djb2:

```
unsigned long djb2hash(unsigned char *str){
    unsigned long hash = 5381;
    int c;
    while(c = *str++)
        hash = ((hash << 5) + hash) + c; // hash * 33 + c
    return hash;
}
```

Hàm trên bắt đầu với việc gán `hash` bằng 5381.
Tiếp đó với mỗi ký tự của của `str` thực hiện thao tác sau:

- nhân `hash` với 33 (thay bằng $(hash << 5) + hash$ bởi vì phép dịch bit được máy tính xử lí nhanh hơn).
- cộng với giá trị **ASCII** của ký tự hiện tại.

Cuối cùng là trả về giá trị không âm của `hash`.
Số 5381 và 33 được chọn bởi qua thực nghiệm nó cho kết quả ít va chạm hơn (Hash colliosion) và có hiệu ứng tuyết lở tốt hơn (Avalanche effect).
Nguồn: tại đây

Giải thích cách chọn độ lớn của mảng băm

Cách chọn độ lớn của mảng băm:

- Nhân số lượng phần tử tối đa được thêm vào mảng băm với 1.3 .
- Chọn số nguyên tố nhỏ nhất lớn hơn số bên trên.

Độ lớn mảng băm là số trên. Chọn số trên bởi vì nếu độ lớn của mảng băm quá nhỏ thì sẽ tạo sự va chạm (collision) ảnh hưởng tới tốc độ truy cập. Còn nếu độ lớn mảng băm quá lớn sẽ gây hao phí. Chọn số nguyên tố để thuận lợi trong việc chọn biến băm của **Double Hashing**.
Nguồn: tại đây

Giải thích cách chọn biến băm thứ hai trong Double Hashing

Nếu mảng băm có độ lớn M, biến băm thứ 2 nên chọn từ 1 đến M-1.
Nếu M là số nguyên tố, một lựa chọn phổ biến là:

```
    H2(K) = 1 + ( (K/M) mod (M-1) )
```

Cách lựa chọn trên sẽ giúp giảm sự va chạm và luôn tìm được vị trí mới nếu có va chạm.
Nguồn: tại đây

Bảng so sánh

Methods	**Run time (ms)**
Linear Search	1763
Binary Search	8
Chaining Approach	2
Linear Probing	2
Quadratic Probing	4
Double Hashing	3

Bảng 1: So sánh thuật toán tìm kiếm với input 18500 phần tử

Methods	**Run time (ms)**
Linear Search	6327
Binary Search	20
Chaining Approach	5
Linear Probing	6
Quadratic Probing	6
Double Hashing	11

Bảng 2: So sánh thuật toán tìm kiếm với input 50000 phần tử

Nhận xét:

- Linear Search có tố độ thực thi lâu nhất nhất.
- Binary Search có tốc độ thực thi lâu thứ hai
- Còn các cách xử lí Hash đều có tốc độ thực thi gần tương tự nhau.

Nguyên nhân:

- Độ phức tạp của Linear Search là O(N).
- Độ phức tạp của Binary Search là O($log_2 N$)
- Độ phức tạp khi tìm kiếm trong điều kiện bình thường của Hash là O(1) (trong trường hợp tệ nhất là O(N)), sự khác biệt giữa các phương thức xử lí va chạm khác nhau do dữ liệu đầu.