**Câu 4 (2.5 điểm):**

**a. Anh/Chị hãy cho biết ưu điểm bảng băm khi so với cấu trúc dữ liệu đã học (mảng, danh sách liên kết đơn, cây).**

**b. Cho bảng băm A kích thước 7 phần tử và tập khóa K = {76, 40, 48, 5, 20}. Hãy vẽ bảng băm khi thêm từng khóa K vào bảng A với hàm băm hf(k) =k % 7, trong trường hợp xảy ra đụng độ, hãy sử dụng phương pháp dò bậc hai (Quadratic Probing Method) để giải quyết đụng độ.**

**Giải:**

**a.**

Một số ưu điểm của bảng băm so với các cấu trúc dữ liệu khác như mảng, danh sách liên kết đơn và cây:

- **Tìm kiếm nhanh:** Truy cập và tìm kiếm trong bảng băm có thể được thực hiện với thời gian hằng số O(1), với điều kiện rằng hàm băm được triển khai tốt và không có quá nhiều đụng độ.

- **Dùng ít bộ nhớ:** So với cấu trúc dữ liệu như danh sách liên kết đơn và cây, bảng băm thường dùng ít bộ nhớ hơn. Bởi vì không cần lưu trữ các con trỏ hay tham chiếu giữa các phần tử, chỉ cần lưu trữ các khóa và giá trị tương ứng.

- **Xử lý đụng độ:** Bảng băm cung cấp cơ chế xử lý đụng độ, cho phép lưu trữ các khóa có cùng giá trị băm trong cùng một ô. Phương pháp xử lý đụng độ có thể được lựa chọn để đảm bảo hiệu quả tốt nhất trong trường hợp cụ thể.

- **Thao tác chèn và xóa linh hoạt:** Bảng băm cho phép thực hiện các thao tác chèn (insert) và xóa (delete) với độ phức tạp trung bình O(1). Điều này là do thời gian tìm kiếm và truy cập vào ô băm được giữ ở mức độ hằng số.

**- Lưu ý:** Trong trường hợp tốt nhất, việc tìm kiếm trong bảng băm có thể được thực hiện với thời gian hằng số O(1). Tuy nhiên, điều này đòi hỏi một số giả định:

+ Giả định về hàm băm hoàn hảo: Hàm băm hoàn hảo là một hàm băm đảm bảo không xảy ra đụng độ, có nghĩa là mỗi khóa sẽ được ánh xạ vào một vị trí duy nhất trong bảng băm. Trong trường hợp này, việc tìm kiếm chỉ cần một phép tính băm và truy cập vào vị trí tương ứng, cho nên có thể được thực hiện với thời gian hằng số O(1).

+ Giả định về truy cập ngẫu nhiên: Giả định rằng việc truy cập các khóa trong bảng băm được thực hiện ngẫu nhiên, không có mẫu tìm kiếm đặc biệt hoặc tập truy vấn có thể tạo ra đụng độ tập trung. Trong trường hợp này, việc tìm kiếm trong bảng băm vẫn có thể đạt được thời gian hằng số O(1) trung bình.

**b.**

- Với kích thước 7 ô, ta có các hàm băm sau:

+ Hàm băm: hf(k) = k % 7

+ Hàm băm lại: hfi(k) = [hf(k) + i2] % 7

- Cách làm cụ thể được minh hoạ dưới bảng sau (hàng trên là vị trí i, hang dưới là giá trị k)

+ Với **k lần lượt là** **76 và 40** thì ta được vị trí i là 6 và 5

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **0** | **1** | **2** | **3** | **4** | **5** | **6** |
| **NULL** | **NULL** | **NULL** | **NULL** | **NULL** | **40** | **76** |

+ Với **k = 48:**

* hf(k) = 6 => xảy ra đụng độ với hf(key = 76), do đó ta sẽ dò bậc 2 để xử lý.
* hfi(k) = [hf(k) + i2] % 7 = (6 + i2) % 7
* hfi(k) = (6 + i2) % 7. Với i = 1, ta được hf1(48) = 0.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **0** | **1** | **2** | **3** | **4** | **5** | **6** |
| **48** | **NULL** | **NULL** | **NULL** | **NULL** | **40** | **76** |

**+** Với **k = 5:**

* hf(k) = 5 => xảy ra đụng độ với hf(key = 40), do đó ta tiếp tục dò bậc 2 để xử lý.
* hfi(k) = [hf(k) + i2] % 7 = (5 + i2) % 7
* hfi(k) = (5 + i2) % 7. Với i = 2, ta được hf2(5) = 2.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **0** | **1** | **2** | **3** | **4** | **5** | **6** |
| **48** | **NULL** | **5** | **NULL** | **NULL** | **40** | **76** |

**+** Với **k = 20:**

* hf(k) = 6 => xảy ra đụng độ với hf(key = 76), do đó ta tiếp tục dò bậc 2 để xử lý.
* hfi(k) = [hf(k) + i2] % 7 = (6 + i2) % 7
* hfi(k) = (6 + i2) % 7. Với i = 2, ta được hf2(20) = 3.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **0** | **1** | **2** | **3** | **4** | **5** | **6** |
| **48** | **NULL** | **5** | **20** | **NULL** | **40** | **76** |

**=> Bảng băm sau khi được thêm các key vào là:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **0** | **1** | **2** | **3** | **4** | **5** | **6** |
| **48** | **NULL** | **5** | **20** | **NULL** | **40** | **76** |