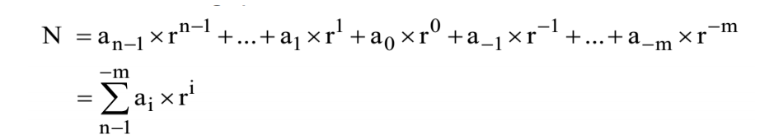
**.   Thuật toán Hash:**

**a.   Ký hiệu:**

* Tập hợp các chữ cái được sử dụng: Σ.
* Đoạn văn bản: T[1. . m],
* Đoạn mẫu: P[1. . n],
* Đoạn con từ i  đến j  của một xâu s: s[i. . j]
* Chúng ta cần tìm ra tất cả các vị trí i  (1 ≤ i  ≤ m − n + 1) thỏa  mãn: T[i. . i  + n − 1] = P.

b. Mô tả thuật toán:

Để đơn giản, giả sử rằng Σ = {a, b, … , z},  nghĩa là Σ chỉ gồm các chữ cái Latin  in thường. Để biểu diễn một xâu, thay vì dùng chữ cái, chúng ta sẽ chuyển sang biểu diễn dạng số. Chọn hệ cơ số để biểu diễn sang dạng số.



Ví dụ: xâu aczd được viết dưới dạng số là 1 dãy số: ( 0, 2, 25, 3);

Ví dụ 2: giả sử ta nhập vào 2 xâu a và b từ bàn phím: hỏi xâu a và b có giống nhau hay không?

* Chuyển đổi 2 xâu này về dạng số và so sánh 2 số này có bằng nhau hay không?

|  |
| --- |
| #include <bits/stdc++.h>  using namespace std;  string s;  long long base = 1e9+7;  main()  {  cin>>s;  int m=s.size();  s=" "+s;  int hashp;  for(int i=1;i<=m;i++)  {  hashp=(hashp\*26+s[i]-'a')%base;  }  cout<<hashp;  } |

* Code 2 xâu:

|  |
| --- |
| #include <bits/stdc++.h>  using namespace std;  string a,b;  long long base = 1e9+7;  long long has (string s)  {  int n=s.size();  s==" "+s;  long long hasp=0;  for(int i=1;i<=s.size();i++)  {  hasp=(hasp\*26 + s[i]-'a')% base;  }  return hasp;  }  main()  {  cin>>a>>b;  if(has(a)==has(b)) cout <<"YES";  else cout<<"NO";  } |

Để tính mã Hash của xâu P chúng ta chỉ cần làm đơn giản như sau:

hashP = 0

for (i : 1 .. n)

hashP = (hashP \* 26 + P[i] - 'a') mod base

Phần khó hơn của thuật toán Hash là: Tính mã Hash của một đoạn con từ i  đến j T[i. . j] của xâu T (1 ≤ i  ≤ j  ≤ N).

pow[0] = 1

for (i : 1 .. m)

pow[i] = (pow[i-1] \* 26) mod base

hashT[0] = 0

for (i : 1 .. m)

hashT[i] = (hashT[i-1] \* 26 + T[i] - 'a') mod base

Để lấy mã Hash của T[i. . j] ta viết hàm sau:

function getHashT(i, j):

return (hashT[j] - hashT[i - 1] \* pow[j - i + 1] + base \* base) mod base

II. BÀI TẬP

**Xâu con ( SUBSTR )**

Cho xâu A và xâu B chỉ gồm các chữ cái thường. Xâu B được gọi là xuất hiện tại vị trí i của xâu A nếu: A[i] = B[1], A[i+1] = B[2], ..., A[i+length(B)-1] = B[length(B)].

Hãy tìm tất cả các vị trí mà B xuất hiện trong A.

Dữ liệu vào:

* Dòng 1: xâu A.
* Dòng 2: xâu B.

Độ dài A, B không quá 1000000.

Dữ liệu ra:

Ghi ra các vị trí tìm được trên 1 dòng (thứ tự tăng dần). Nếu B không xuất hiện trong A thì bỏ trắng.

Ví dụ:

|  |  |
| --- | --- |
| **SUBSTR.INP** | **SUBSTR.OUT** |
| aaaaa  aa | 1 2 3 4 |

Nguồn : Spoj.

**Ý tưởng:**

Dùng hash để kiểm tra xâu con có nằm trong xâu ban đầu không. Độ phức tạp O(n).

**BÀI 02:**

**Xâu đối xứng - STRPALIN ( Bài 3 lớp10 đề Olimpic trại hè HV2015)**

Tham dự thi trại hè lần này, bạn Sơn Tùng đăng ký tham dự trò chơi RCV, phần thưởng cho người chiến thắng là một chiếc thẻ nhớ 16GB từ ban tổ chức (BTC). Trò chơi như sau: BCT đưa ra một chuỗi các ký tự chỉ bao gồm các chữ cái in thường từ **a** đến **z**, và một số bộ **l**, **r**. Người chơi sẽ trả lời một bộ các câu hỏi của BTC có dạng: xâu con nhận được kể từ vị trí **l** đến vị trí **r** có đối xứng hay không?

Bạn hãy giúp Sơn Tùng dành chiến thắng cuộc thi này.

**Dữ liệu:**

* Dòng đầu ghi xâu ký tự không cho biết độ dài.
* Dòng tiếp theo ghi số nguyên – số câu hỏi của BTC.
* dòng tiếp theo, mỗi dòng ghi hai số nguyên **l**, **r**

**Kết quả:** Gồm dòng, mỗi dòng ghi tương ứng câu trả lời, nếu có ghi **1**, ngược lại ghi **-1**.

**Ví dụ:**

|  |  |
| --- | --- |
| **STRPALIN.INP** | **STRPALIN.OUT** |
| **abxbagredcnmooojhggohoreomodioibba**  **5**  **1 5**  **10 23**  **25 27**  **29 31**  **13 15** | **1**  **-1**  **1**  **1**  **1** |

**Ràng buộc:**

* 30% số test có độ dài xâu
* 30% số test có xâu có độ dài 103
* 40% số test độ dài xâu không quá 105 và ,