Hash-uri

Tabele cu adresare directă. Tabele de dispersie

Hash

Definiție

Funcția de hash = o funcție (matematică) care convertește un input de lungime arbitrară într-o valoare de dimensiune fixată

hash = codificarea unui input / output-ul unei funcții de hash

⇒ mapăm o mulțime (mare) de obiecte valori mici / ușor de procesat

Exemple:

h(x) = x % p

Tabele de dispersie

Complexități pe operații ale unor structuri de date:

	Inserare	Ștergere min	Ștergere cu pointer	Ștergere fără pointer	Afișare minim	Căutare	Succesor	Afișare sortat
Heap	O(log n)	O(log n)	O(log n)	O(n)	O(1)	O(n) :(	O(n) :(	O(n logn)
Arbori de căutare echilibrați	O(log n)	O(log n)	O(log n)	O(log n)	O(log n)	O(log n)	O(log n)	O(n)
Vector	O(1)	O(n)	O(?) O(1) sau O(n)	O(n)	O(n)	O(n)	O(n)	O(n log n)
Listă înlănțuită	O(1)	O(n)	O(1)	O(n)	O(n)	O(n)	O(n)	O(n log n)

Tabele cu adresare directă

Problemă: Se dau 2 tipuri de operații pe numere de la 1 la N ($N \leq 10^6$). Se dau până la $M \leq 10^6$ operații.

- Inserați numărul x
- Întrebare: numărul x se află între numerele date?

Soluție?

- (ineficientă) Insertion sort pe inserare, apoi căutare binară
  - O(n) pe inserare
  - O(log n) pe căutare

Tabele de dispersie

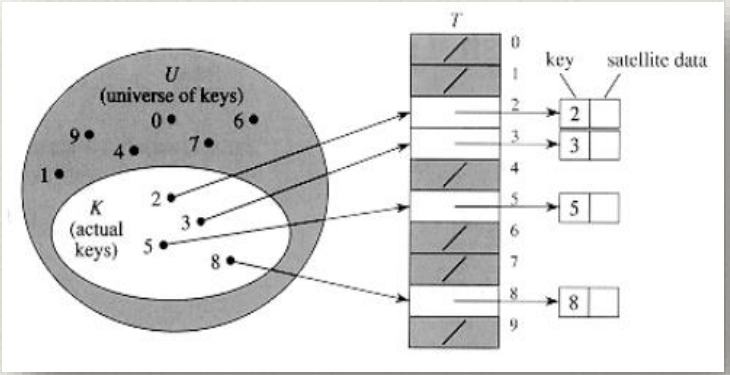
Recapitulare: heap-urile și arborii binari de căutare țin ordine… prea complicat

	Inserare	Ștergere min	Ștergere cu pointer	Ștergere fără pointer	Afișare minim	Căutare	Succesor	Afișare sortat
Heap	O(log n)	O(log n)	O(log n)	O(n)	O(1)	O(n) :(	O(n) :(	O(n log n)
Arbori de căutare echilibrați	O(log n)	O(log n)	O(log n)	O(log n)	O(log n)	O(log n)	O(log n)	O(n)
Hashuri	O(1)	???	O(1)	O(1)	???	O(1)	???	???

Tabele cu adresare directă

Problemă: Se dau 2 tipuri de operații pe numere de la 1 la N ($N \leq 10^6$). Se dau până la $M \leq 10^6$ operații.

- Inserați numărul x
- Întrebare: numărul x se află între numerele date?



Soluție?

- Putem ține un vector binar: a[i]=1, dacă elementul s-a dat, și a[i]=0, dacă elementul nu a fost dat. Inițial este totul 0.
  - Complexitate?
    - **O(1) inserare și căutare!!!**

Tabele cu adresare directă

Problemă: Se dau 2 tipuri de operații pe numere de la 1 la N ($N \leq 10^6$).

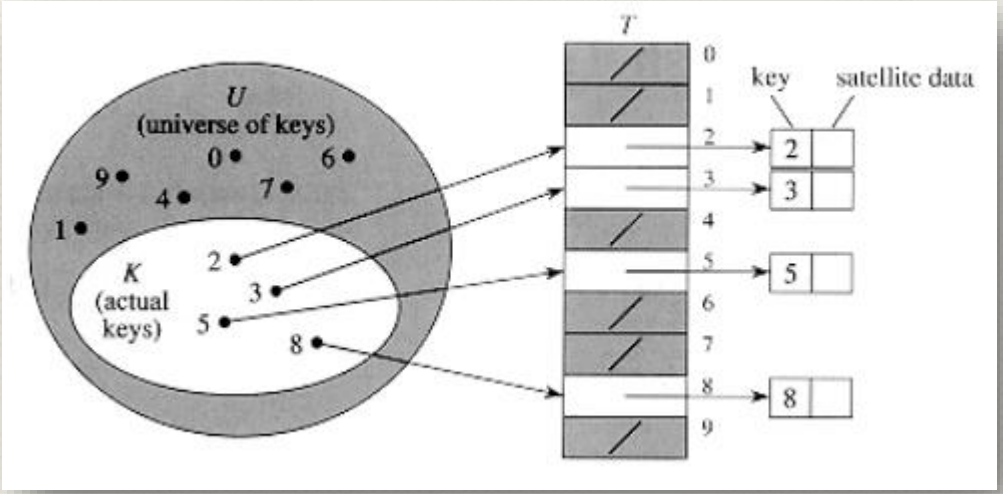
- Inserați numărul x
- Întrebare: numărul x se află între numerele date?
- **Ștergere: Elimin numărul x din numerele mele**

Soluție?

- Putem ține un vector binar: a[i]=1 dacă elementul s-a dat și a[i]=0 dacă elementul nu a fost dat. Inițial este totul 0.
  - Complexitate?
    - O(1) inserare și căutare!!! **Și ștergere!!!**

Tabele cu adresare directă

Implementare O(1) și foarte scurtă!




```
DIRECT-ADDRESS-SEARCH(T,k)
return T[k]
DIRECT-ADDRESS-INSERT(T,x)
T[key[x]] ← x
DIRECT-ADDRESS-DELETE(T,x)
T[key[x]] ← NIL
```

Tabele cu adresare directă

Problemă: Se dau 2 tipuri de operații pe numere de la 1 la N ($N \leq 10^6$).

- Putem ține un vector binar: a[i]=1, dacă elementul s-a dat, și a[i]=0 dacă elementul nu a fost dat. Inițial este totul 0.
  - **Complexitate?**
    - O(1) inserare, căutare și ștergere

Unde apare problema?

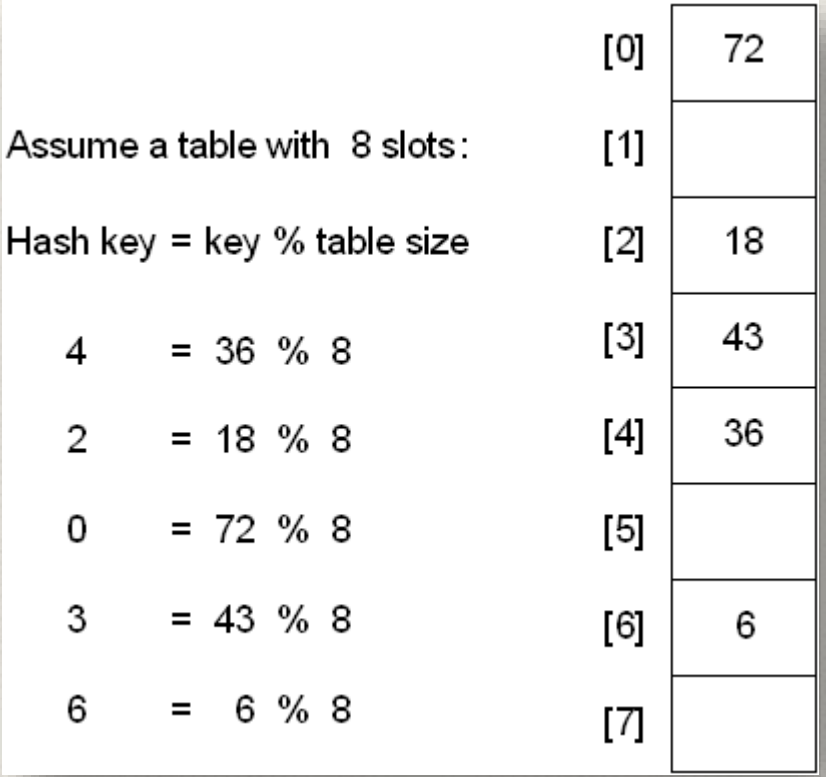
- **Limita de $N \leq 10^6$.**
  - Dacă am numere mai mari? Dacă am cuvinte sau altfel de obiecte? Dacă am numere negative?
- https://leetcode.com/problems/two-sum/ **(problemă clasică de interviuri)**
  (cod1 va merge doar pt N mic ~= 10^7, nu putem aloca oricâtă memorie)

Tabele de dispersie

Trebuie să luăm elementele și să le dispersăm.

- Unde sunt problemele?
  - Mai multe elemente pică pe același câmp → **coliziune**
  - Cam toate elementele pică în același loc → **funcție de dispersie proastă**!
    - Ex: %100 la prețuri de televizor

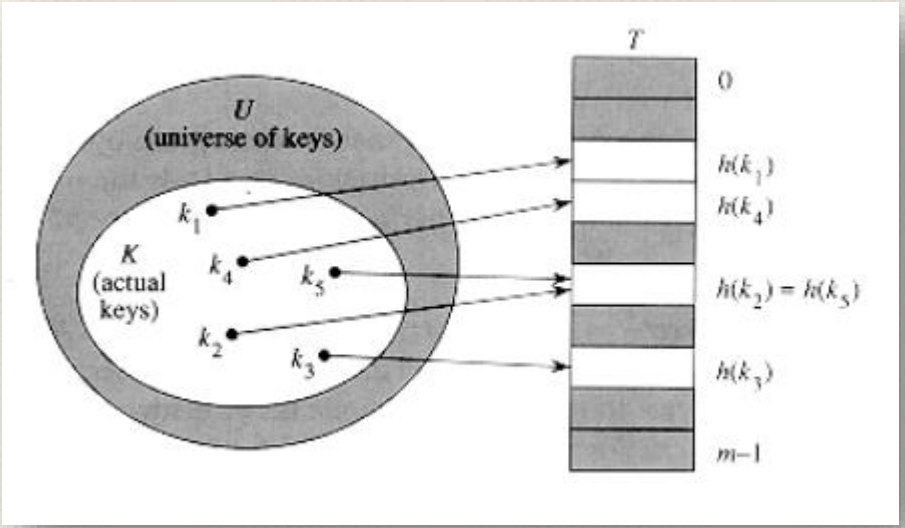
- **Cum le rezolvăm?**



Tabele de dispersie

Trebuie să luăm elementele să le dispersăm

- În realitate elementele se suprapun



Assume a table with 8 slots:

Hash key = key % table size

4 = 36 % 8

2 = 18 % 8

0 = 72 % 8

3 = 43 % 8

6 = 6 % 8

[0]	72
[1]	
[2]	18
[3]	43
[4]	36
[5]	
[6]	6
[7]	

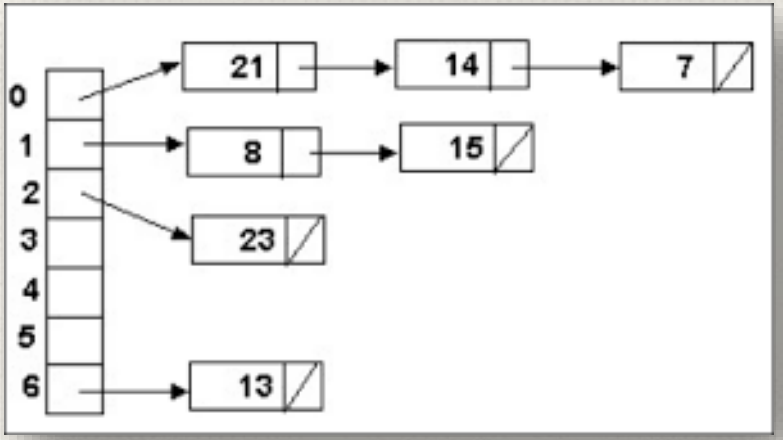
Tabele de dispersie

- Există mai multe metode de dispersie
- Astăzi ne vom axa pe una simplă și eficientă
- În practică, h(x) = x % p , unde p este un număr prim
- Vom discuta în detaliu la cursul următor despre metodele de alegere ale funcției de dispersie.

Rezolvarea coliziunilor

- Vom discuta în cursul următor mai multe metode, pentru moment, voi alege înlănțuire.

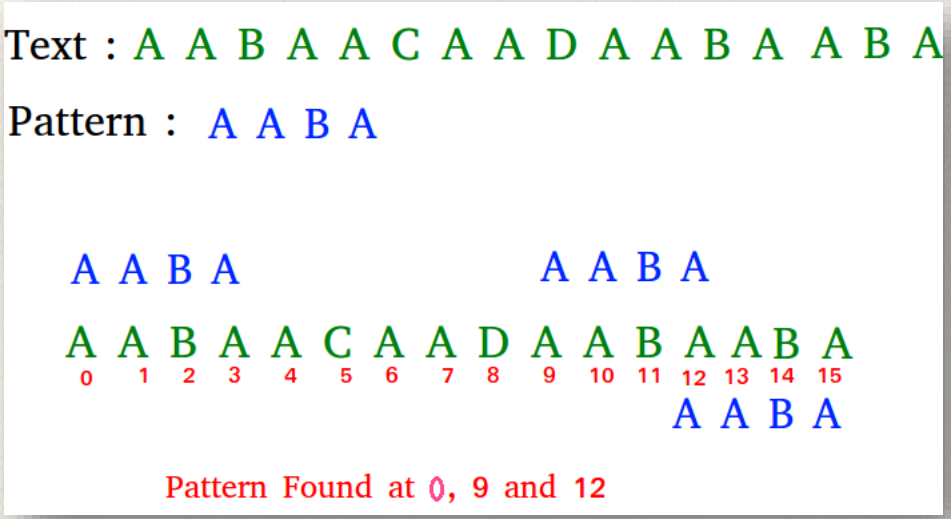
- Hai să codăm :)
  - https://leetcode.com/problems/two-sum/
  - Practic implementăm de mână inserarea și căutarea într-un hash



Problemă

- https://www.infoarena.ro/problema/strmatch
- Vrem să calculăm toate aparițiile unui șir mai mic (subșir) într-un șir mai mare

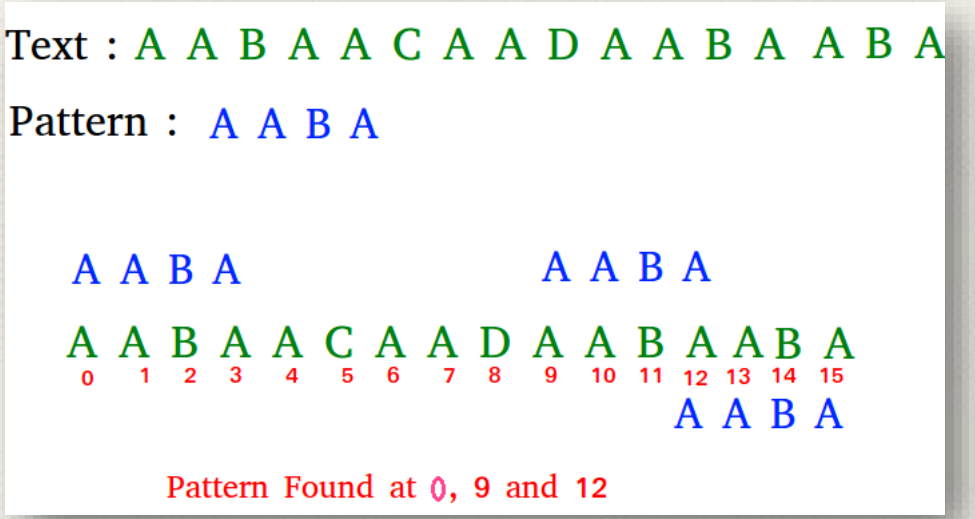
- **Soluții?**



Pattern Matching

Algoritmul **Rabin Karp!**

1. Calculăm hash-ul pentru șirul mai mic
2. Calculăm hash-ul pentru toate șirurile de aceeași lungime din șirul mai mare



Pattern Matching

Algoritmul **Rabin Karp!**

1. Calculăm hash-ul pentru șirul mai mic
2. Calculăm hash-ul pentru toate șirurile de aceeași lungime din șirul mai mare

set_size = 10 (decimal)

1234

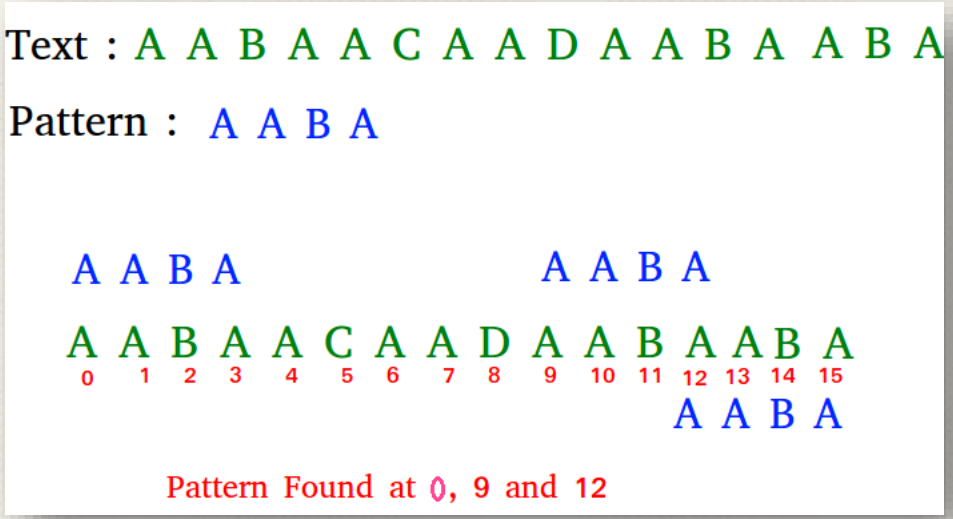
$1x10^3+2x10^2+3x10^1+4x10^0$

$(1x10^2+2x10^1)10+3x10^1+4x10^0$

$((1x10^2+2x10^1)+3)10+4x10^0$

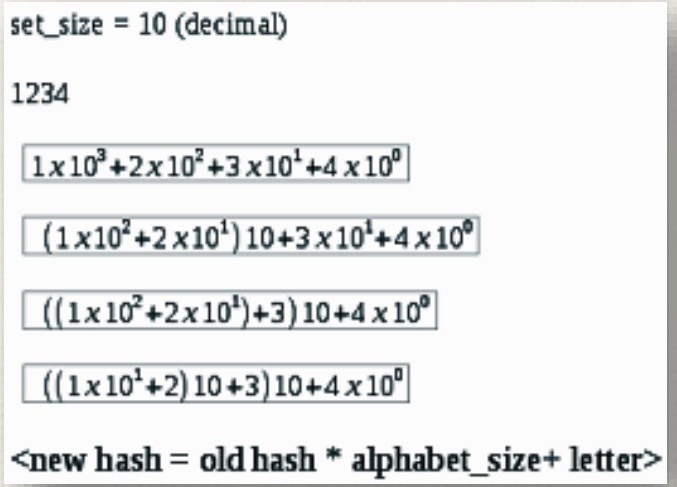
$((1x10^1+2)10+3)10+4x10^0$

<new hash = old hash * alphabet_size+ letter>



Pattern Matching

- Cum calculăm rolling hash?
- Ce probleme ar putea apărea?
  - Dacă două șiruri au hash-uri egale? Sunt egale?
- **Soluții?**
  - Verificăm fiecare potrivire
    - Ce se întâmplă dacă avem:
      - aaa
      - aaaaaaaaaaa
      - O(n*m)
  - Facem 2 hash-uri și vedem dacă ambele sunt egale
    - Dacă ambele sunt egale, atunci suntem OK.



Codăm:

- https://www.infoarena.ro/problema/strmatch
- implementare posibilă

Kahoot