Redispersió en una taula de dispersió amb sinònims encadenats indirectes i nova mida nombre primer S17017_ca

Es disposa d'un diccionari que emmagatzema únicament claus de tipus `string`, implementat mitjançant una taula de dispersió amb encadenaments indirectes (els sinònims estan en llistes simplement encadenades ordenades per clau).

Entre els fitxers que s'adjunten en aquest exercici, trobaràs:

- `dicc.hpp`: especificació pública de la classe `dicc`.
- `dicc.old.rep`: representació interna (estructura de dades i atributs privats).
- `dicc.cpp`: implementació de la classe (mètodes ja resolts).

La teva feina és:

- Canviar el nom del fitxer de representació:

  ```
  mv dicc.old.rep dicc.rep
  ```

  Així podràs modificar la representació de la classe i afegir mètodes privats.

- Crear el fitxer `solution.cpp`: fitxer on has d'implementar **la teva solució**.

Has d'implementar el mètode següent de la classe `dicc`:

```
/* Mètode modificador.
   Realitza la redispersió de la taula de dispersió.
   La mida de la nova taula serà el primer nombre primer
   major o igual que 2*_M + 1, on _M és la mida actual de la taula.
   Tots els elements del diccionari es reubiquen a la nova taula
   utilitzant la funció de dispersió corresponent. */
void redispersio();
```

Aquest mètode ha de realitzar la **redispersió** de la taula de dispersió:

1. La nova mida de la taula ha de ser el **primer nombre primer** major o igual que

   $$2 \cdot M + 1,$$

   on M és la mida actual de la taula.

2. S'han de **recol·locar tots els elements** del diccionari a la nova taula, recalculant el seu índex amb la funció de dispersió i la nova mida.

Observacions

- No es pot perdre cap element del diccionari durant la redispersió.
- El nombre d'elements del diccionari (consultable amb `quants()` o mitjançant l'atribut corresponent) **no ha de variar**.
- No s'han de crear claus noves ni duplicar-les.
- Es poden definir funcions i mètodes auxiliars si es considera necessari.
- Per calcular el valor de dispersió utilitza el mètode `h` que ja està implementat i que permet calcular un valor de dispersió entre 0 i `LONG_MAX` (el valor long int més gran que permet el compilador) a partir d'una clau.
- **IMPORTANT**: En la representació només pots afegir mètodes auxiliars o atributs, no pots modificar cap altra cosa.
- Recorda que has d'afegir a tots els mètodes que creïs les corresponents **Precondició** (Pre) i **Postcondició** (Post) i també el cost.
- També pots trobar entre els fitxers que s'adjunten amb l'exercici el fitxer `main.cpp` (programa principal) que crea la taula de dispersió i afegeix elements. .
- Per a pujar la teva solució, has de crear el fitxer `solution.tar` així:

```
tar cf solution.tar solution.cpp dicc.rep
```

Entrada

L'entrada té tres línies: la primera conté un natural positiu amb la dimensió inicial de la taula de dispersió i les altres dos contenen elements separats amb espais, són els elements que s'insereixen en el diccionari.

Sortida

Escriu el contingut del diccionari dos vegades: després d'inserir el primer conjunt d'elements i després d'inserir el segon conjunt d'elements. Cada vegada es mostra en diferents línies la quantitat d'elements que té i els elements que conté cadascuna de les llistes de sinònims encadenats indirectes (els elements de cada llista apareixen separats amb un espai i en el mateix ordre en que es guarden).

Per més detalls de com és l'entrada i la sortida consulta els jocs de proves públics.

Exemple d'entrada 1

```
13
roma amor mora mor mar 7art rata tres maro
7art cert crit cor cop rata tros armo
```

Exemple de sortida 1

```
Nº elem: 9
0:
1:
2: amor maro mora roma
3:
4: tres
5: 7art
6:
7:
8: mar rata
```

```
9: mor
10:
11:
12:
-----------
Nº elem: 15
0:
1: mar
2:
3: cop
4:
5: 7art cor
6:
7:
8:
9:
10:
11: tres
12:
13:
14:
15: mor
16:
17:
18: rata
19:
20:
21: tros
22:
23:
24: cert
25: amor armo maro mora roma
26:
27:
28: crit
-----------
```

Exemple d'entrada 2

```
29
5 -5 3 -3 9 -9 2 -2 -5 5 1 -1 7 -7 0 4 -4 8 -8 6 -6
2 11 4 12 8 14 0 10 17 13
```

Exemple de sortida 2

```
Nº elem: 19
0:
1:
2:
3:
4:
5:
6:
7: -1
8: -2
9: -3
10: -4
11: -5
12: -6
13: -7
14: -8
15: -9
16:
17:
18:
19: 0
20: 1
21: 2
22: 3
23: 4
24: 5
25: 6
26: 7
27: 8
28: 9
-----------
Nº elem: 25
0:
1:
2:
3:
4:
5:
6:
7:
```

```
8:
9:
10:
11:
12:
13:
14:
15:
16:
17:
18:
19:
20:
21:
22:
23:
24:
25:
26:
27:
28:
29:
30:
31:
32:
33:
34:
35: -1
36: -2
37: -3
38: -4 10
39: -5 11
40: -6 12
41: -7 13
42: -8 14
43: -9
44:
45: 17
46:
47:
48: 0
49: 1
50: 2
51: 3
52: 4
53: 5
54: 6
55: 7
56: 8
57: 9
58:
-----------
```

Exemple d'entrada 3

```
13
pikachu bulbasaur charmander squirtle eevee pikachu eevee snorlax gengar onix
mew mewtwo eevee snorlax dragonite charizard blastoise venusaur pikachu raichu alakazam machamp
```

Exemple de sortida 3

```
Nº elem: 8
0: pikachu eevee snorlax gengar onix
1:
2: eevee
3:
4: gengar onix
5: charmander squirtle
6:
7:
8: snorlax
9:
10:
11:
12: bulbasaur
-----------
Nº elem: 23
0: dragonite eevee
1: charmander
2: machamp
3:
4: bulbasaur
5: lapras
6:
7:
8: mewtwo
9: blastoise
10: mew
11: onix
12:
13:
14:
15:
```

```
16: pikachu
17: gyarados
18:
19: gengar squirtle venusaur
20:
21: snorlax
22: alakazam
23: pidgeot
24: charizard
25:
26: ditto
27: raichu
28: rattata zubat
-----------
```

Informació del problema

Autoria: Bernardino Casas

Generació: 2026-01-25T20:33:30.987Z

© *Jutge.org*, 2006–2026.
https://jutge.org