Tema 1 Multi-platform Development

Scopul temei

- Recapitularea lucrului cu funcțiile ANSI C:
  - lucrul cu fișiere
  - alocare dinamică de memorie
- Folosirea pointerilor

Enunț

Să se implementeze **în C** o tabelă de dispersie (hashtable) ce va conține cuvinte. Operațiile ce trebuie implementate pentru tabelă sunt următoarele:

Operația	**Descrierea operație**
add <cuvânt>	adaugă cuvântul la hashtable (nu se permit dubluri)
remove <cuvânt>	șterge cuvântul din hashtable (nu e obligatoriu să existe cuvântul)
find <cuvânt> [<fișier_ieșire>]	caută cuvântul în hashtable → scrie True sau False pe o linie nouă în fișierul specificat sau la consolă dacă acest parametru lipsește
clear	golește tabela
print_bucket <index_bucket> [<fișier_ieșire>]	scrie cuvintele din bucketul respectiv, pe o singură linie și separate de spațiu în fișierul specificat sau la consolă dacă acest parametru lipsește, index_bucket este valid
print [<fișier_ieșire>]	printează toate bucket-urile pe linii diferite, începând cu bucketul 0, în fișierul specificat sau la consolă dacă acest parametru lipsește
resize double	dublează dimensiunea hash-ului (bucket-urile vor fi parcuse în ordine și cuvintele sunt redistribuite)
resize halve	înjumătățește dimensiunea hash-ului (bucket-urile vor fi parcuse în ordine și cuvintele sunt redistribuite, memoria în surplus va fi dealocată)

Aceste comenzi se vor regăsi una per linie.

Programul va primi o serie de parametri:

- Primul parametru este lungimea inițială a hashtable-ului
- Următorii parametri sunt opționali și reprezintă o listă de fișiere de intrare din care se face citirea. Dacă aceștia lipsesc citirea se face de la STDIN. Atenție, în cazul în care sunt specificate mai multe fișiere de intrare, toate operațiile se aplică **aceluiași** hash, în oridinea în care au fost transmiși din linia de comandă. Dacă un fișier nu există, el este ignorat.

Exemplu:

```
 ./tema1 256 hash1.in hash2.in
 ./tema1 100 < hash1.in

hash1.in:
add tema
add hash
print hash.out
find tema hash.out
remove tema
find tema hash.out
print hash.out
resize double
print
print_bucket 185 hash2.out
```

Hashtable-ul implementat va conține SIZE bucketuri. Fiecare bucket va conține cuvintele în ordinea în care ele au fost introduse. Pentru operația de *resize* bucketurile vor fi parcurse în ordine și redistribuite. Cuvintele din bucket vor fi parcurse începând cu cel mai vechi și terminand cu cel mai recent.

Precizări generale

- Valorile introduse în hashtable sunt cuvinte [A-Za-z].
- Un tablou **nu** poate conține duplicate.
- Nu există limitări pentru lungimea unui bucket.
- Inserarea într-un tablou (bucket) se face la finalul acestuia.
- Funcția de hash ce **trebuie** folosită (în întreaga temă) este definită in *hash.c*. Nu poate fi folosită altă funcție.
- Programul trebuie să execute comenzile în ordine, așa cum au fost primite/citite din fișier(e).
- Liniile goale din fișierul de intrare trebuie ignorate (programul nu face nimic și trece la linia următoare)
- În fișiere se va scrie în modul *append*.
- Dacă dimensiunea hash-ului este impară (2k+1), după înjumătățire dimensiunea lui va fi k.
- Lungimea hash-ului și a unui cuvânt vor fi reprezentate pe un număr pe 32 de biți **(fără semn)**.
- Șirul vid nu este valid.
- Dimensiunea hash-ului va fi întotdeauna pozitivă.
- Executabilul generat va purta numele **`tema1`** pe Linux și **`tema1.exe`** pe Windows.
- Dimensiunea maximă a unei comenzi (operația și cuvântul asociat) este de 20000 de caractere.
- Bufferul folosit pentru citirea comenzilor poate fi declarat cu dimensiune statică.
- Comportamentul dorit la *resize* este unul echivalent cu următorul: se creează un nou hash, se iterează prin bucketurile din vechiul hash și se adaugă în noul hash.
- Hashtable-ul **NU** poate fi implementat folosind vectori statici.
- În cazul în care un bucket este gol, NU trebuie inserată o linie goală
- Orice eroare trebuie raportată la STDERR cu un mesaj sugestiv. Puteți folosiți macro-ul DIE.