Tema 1 – Liste dublu înlănțuite, stive

Ioana Dabelea, Alexandru Tudor, Răzvan Șerb, Ștefan Turcu,
Alin Ichim, Andrei Voicu

Data postării: 28.03.2025
Deadline: 17.04.2025 ora 23:59

Încă de la începuturile Internetului cu primele browsere text-based, precum Lynx, până la Netscape Navigator și Internet Explorer, care au dominat anii '90, și ulterior la Chrome, Firefox și Edge, funcționalitățile de navigare pe web au evoluat continuu pentru a răspunde nevoilor utilizatorilor.

Pe măsură ce mediul online s-a extins și s-a diversificat, comportamentul utilizatorilor s-a schimbat semnificativ. Dacă la început navigarea era liniară și restrânsă la câteva pagini, astăzi utilizatorii deschid simultan zeci de taburi, fiecare urmând un traseu propriu de explorare. În acest context, istoricul de navigare nu mai este doar o listă de pagini accesate, ci un instrument care poate facilita regăsirea rapidă a informațiilor și organizarea conținutului online.

1 Descriere problemă

În această temă, vom explora o arhitectură de gestionare a istoricului unui browser web, inspirată din sistemele utilizate de browserele moderne. Mai departe sunt descrise elementele care vor ajuta la crearea unui sistem utilizat pentru a ține evidența istoricului de navigare al unui browser web care permite deschiderea mai multor taburi simultan, iar în cadrul unui tab se poate naviga între mai multe pagini web.

O pagină web este un document accesibil pe internet, care poate conține text, imagini, videoclipuri și alte elemente interactive. În această temă, vom defini o **pagină** printr-o **structură** ce include un *ID (număr întreg)*, *un șir de caractere ce reprezintă un URL* (maxim 50 de caractere) și *un șir de caractere de dimensiune variabilă ce reprezintă descrierea paginii* (va conține litere, cifre, blanc și semne de punctuație și se va termina cu newline).

Structura unei pagini este:

```
struct page {
    int id;              // ID
    char url[50];        // URL
    char *description;   // Descriere
};
```

Observație

Există o **pagină implicită** care va fi deschisă de fiecare dată când se creează un nou tab. Aceasta are următoarele câmpuri:

- ID - 0
- URL - "https://acs.pub.ro/"
- descriere - “Computer Science”

Important

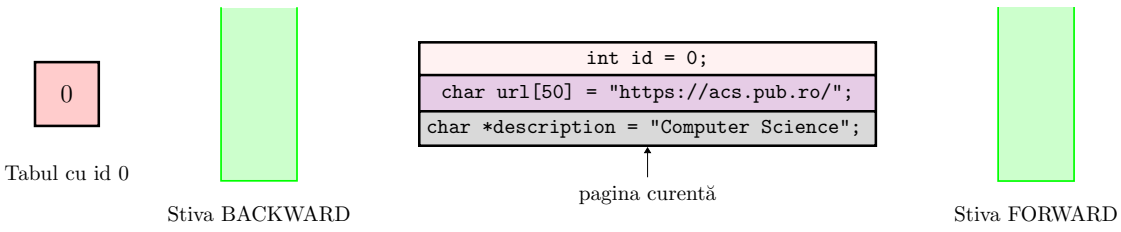
Toate paginile folosite în teste (mai puțin pagina implicită) vor fi citite din fișierul de intrare. **Vor exista maxim 50 de pagini și le puteți salva în orice mod doriți (spre exemplu listă sau vector de structuri).**

Un tab (filă) este o secțiune a browserului web care permite deschiderea și navigarea simultană între mai multe pagini web în aceeași fereastră. În această temă, vom defini un **tab** printr-o **structură** ce conține un *ID unic* (număr întreg) pentru identificarea fiecărui tab, ***un pointer către pagina curentă*** și ***două stive*** folosite pentru istoricul tabului. Una dintre stive (*BACKWARD*) conține **paginile** care au fost deschise pentru a se ajunge la pagina curentă, iar cealaltă stivă (*FORWARD*) conține **paginile** care au fost deschise anterior paginii curente. Stivele vor fi folosite pentru a memora exclusiv istoricul tabului în care sunt definite. O descriere mai detaliată este disponibilă în secțiunea următoare.

Structura unui tab este:

```
struct tab {
        ID          int id;
Pagina curentă      struct page *currentPage;
Stiva BACKWARD      struct stack *backwardStack;
Stiva FORWARD       struct stack *forwardStack;
};
```

O reprezentare a tabului inițial este următoarea:



Un browser (navigator web) este o aplicație care ne permite să accesăm și să navigăm pe internet. Browserul interpretează codul paginilor web și le afișează într-un mod vizual interactiv. Cu alte cuvinte, un browser permite deschiderea și utilizarea simultană a mai multor pagini web prin intermediul taburilor (***în browser se pot deschide taburi, iar în taburi se pot accesa pagini***). În această temă, vom defini un **tab** printr-o **structură** ce conține ***un pointer către tabul curent*** (deschis în browser) și ***o listă dublu înlănțuită circulară cu santinelă*** pentru gestionarea tuturor taburilor.

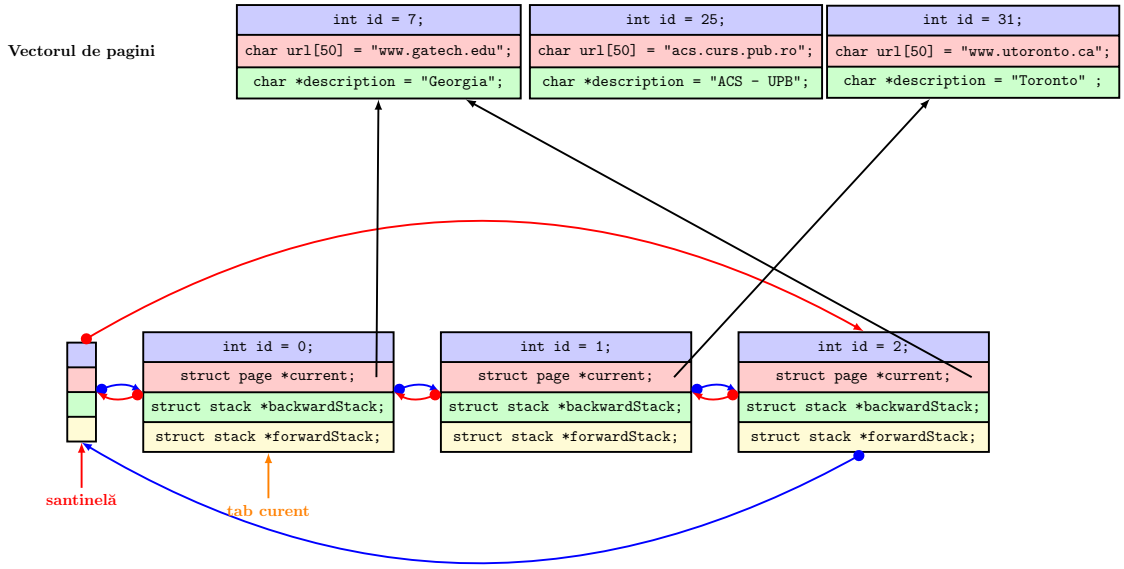
Structura unui browser este:

```
struct browser {
```

Tabul curent	`struct tab *current;`
Listă de taburi	`struct tabsList list;`

```
};
```

O reprezentare completă a unui browser este:



În ilustrația de mai sus este prezentat un browser care conține 3 taburi:

- Tabul cu ID 0, în care este accesată pagina cu ID 7, cu descrierea "Georgia Institute".
- Tabul cu ID 1, în care este accesată pagina cu ID 31, cu descrierea "MIT".
- Tabul cu ID 2, în care este accesată pagina cu ID 25, cu descrierea "University of Toronto".

Observație

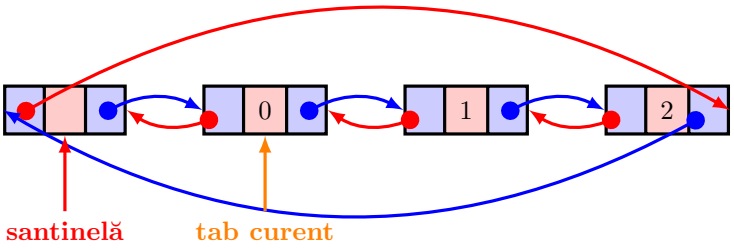
Inițial, în browser va fi deschis doar un singur tab (cu ID 0), în care este accesată pagina implicită.

Important

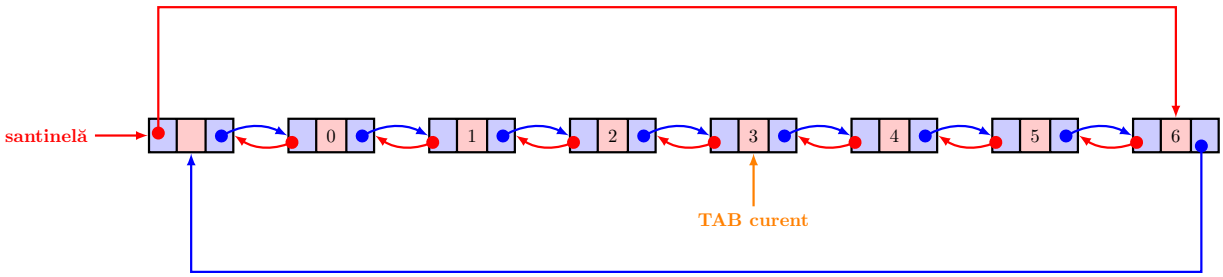
Pentru simplitate, **doar în ilustrațiile următoare** vom folosi doar ID-urile taburilor și paginilor. În locul acestora vor fi, de fapt, adresele structurilor ce reprezintă elementele descrise.

NU trebuie să folosiți ID-ul pe post de elemente și trebuie să lucrați cu structurile specificate!

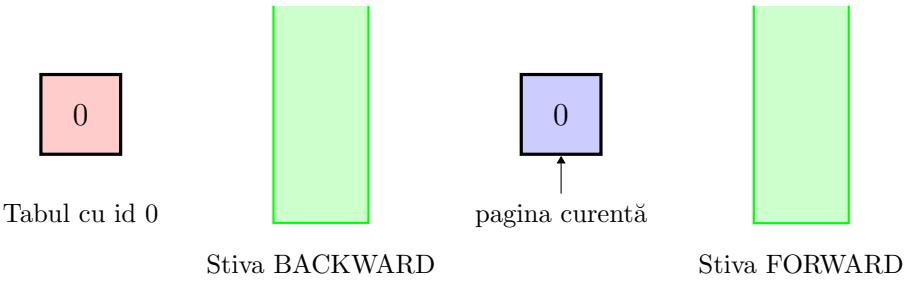
Astfel, putem ilustra structura browserului din exemplul anterior utilizând următoarea listă circulară cu santinelă:



Un alt exemplu, pentru un browser care conține 7 taburi, cu tabul cu ID 3 deschis (setat ca tab curent), este:

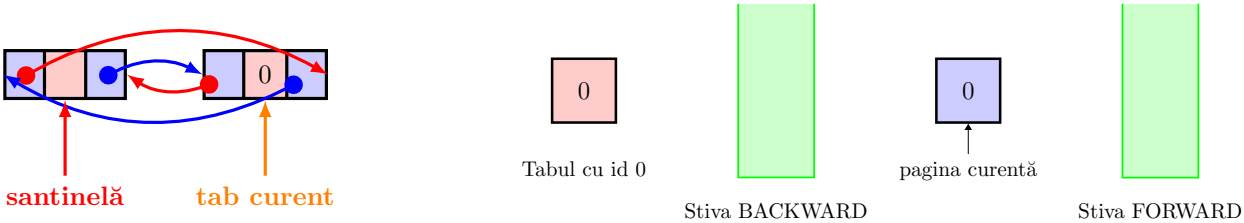


De asemenea, componentele tabului inițial (și a unui nou tab creat în browser) sunt reprezentate astfel:

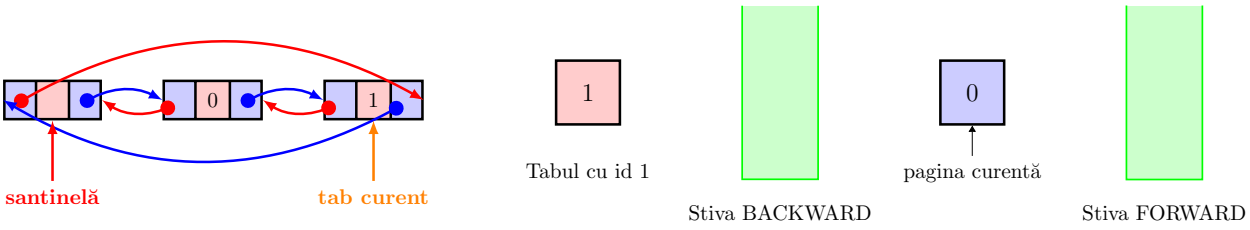


Pentru a înțelege mai bine cum se poate prelucra acest browser, vă propunem următorul exemplu:

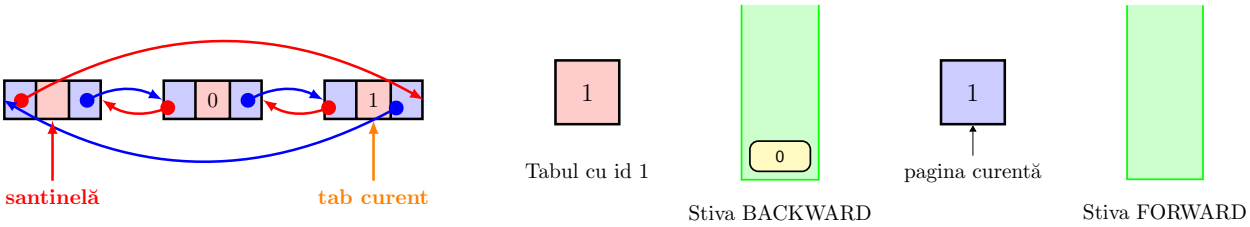
1. Pornim de la conținutul inițial al browserului.



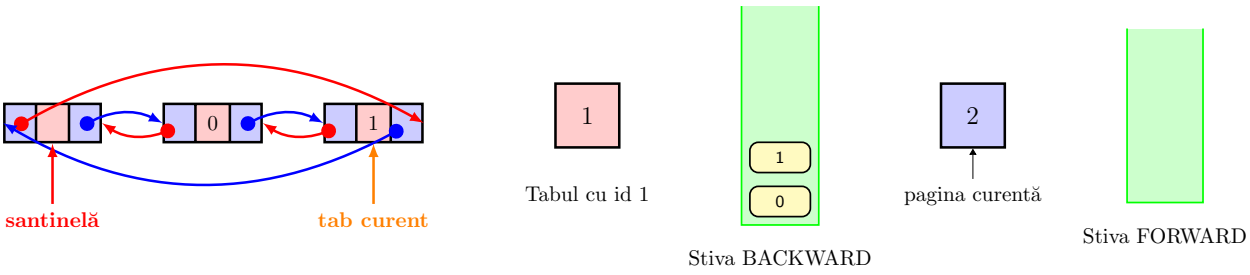
2. În browser se deschide un nou tab, cu ID 1, în care va fi accesată automat pagina 0.



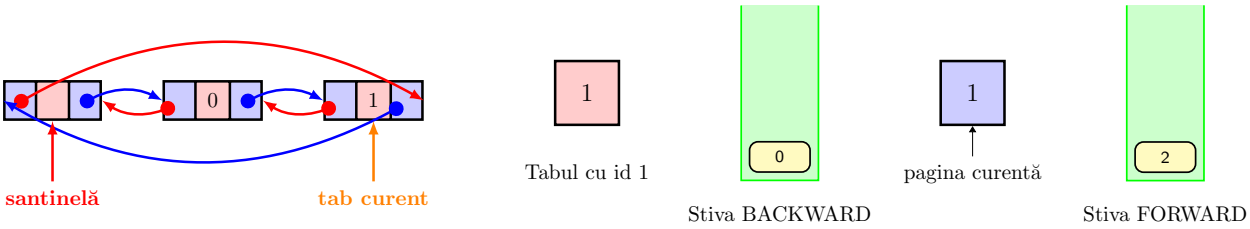
3. În tabul curent se deschide o nouă pagină, cu ID 1.



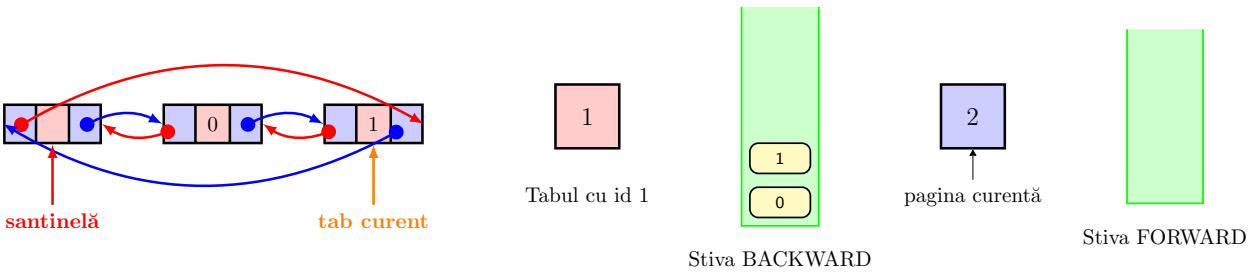
4. În tabul curent se deschide, din nou, o nouă pagină, cu ID 2.



5. Se accesează pagina anterioară folosind butonul BACKWARD.



6. Se accesează pagina următoare folosind butonul FORWARD.



2 Prezentare operații posibile

Va trebui să implementați mai multe funcționalități prin care va fi gestionat istoricul fiecărui tab. În continuare vom prezenta toate acțiunile posibile, specificând pentru fiecare ce particularități prezintă.

2.1 NEW_TAB

- Se creează un nou tab în lista de taburi.
- Inițial, în browser va fi deschis doar un singur tab (cu ID 0, în care este accesată pagina implicită).
- Se va seta pagina implicită pentru un tab nou creat, iar acesta va fi setat ca tab curent al browserului.
- Tabul nou creat va fi adăugat la sfârșitul listei și va primi automat un ID calculat prin incrementarea ID-ului ultimului tab adăugat în lista de taburi.

2.2 CLOSE

- Se închide tabul curent prin eliminarea sa din lista de taburi.
- În browser se va seta ca tab curent tabul aflat în stânga celui șters.
- Nu se va putea șterge niciodată tabul cu ID 0 (tabul inițial). Dacă se încearcă acest lucru, se va afișa mesajul de eroare.

2.3 OPEN <ID>

- Tabul curent al browserului devine tabul cu ID-ul specificat.
- Dacă nu există un tab cu acest ID, se va afișa mesajul de eroare.

2.4 NEXT

- Se deschide în browser tabul care urmează în listă după tabul curent. Astfel, noul tab care se deschide va fi notat ca tab curent al browserului.

2.5 PREV

- Se deschide în browser tabul care se află în listă înaintea tabului curent. Astfel, noul tab care se deschide va fi notat ca tab curent al browserului.

2.6 PAGE <ID>

- În tabul curent se deschide pagina cu ID-ul specificat. Dacă nu există o pagină cu acest ID, se va afișa mesajul de eroare.
- Pagina curentă, care va fi înlocuită de noua pagină, va fi pusă în stiva Backward.
- Conținutul stivei Forward va fi șters.

Observație

Explicație suplimentară
Presupunem că în tabul curent este deschisă pagina 0 și urmează să fie accesate, pe rând, paginile 1 și 2. Când se deschide pagina 1, aceasta va fi setată ca pagină curentă, iar pagina 0 va fi adăugată în stiva Backward (stiva Forward va fi golită). Când se accesează pagina 2, pagina 1 va fi și ea adăugată în stiva Backward (stiva Forward va fi golită).

2.7 `BACKWARD`

- În tabul curent, se va accesa ultima pagină adăugată în stiva Backward. Dacă stiva este goală, se va afișa mesajul de eroare.
- Pagina curentă, care va fi înlocuită de pagina extrasă din stiva Backward, va fi pusă în stiva Forward.

Observație

Explicație suplimentară
Presupunem că în tabul curent este deschisă pagina 2. În stiva Backward se află paginile 1 și 0. Când se execută comanda Backward, pagina 1 va fi scoasă din stivă și va fi setată ca pagină curentă, iar pagina 2 va fi adăugată în stiva Forward.

2.8 `FORWARD`

- În tabul curent, se va accesa ultima pagină adăugată în stiva Forward. Dacă stiva este goală, se va afișa mesajul de eroare.
- Pagina curentă, care va fi înlocuită de pagina extrasă din stiva Forward, va fi pusă în stiva Backward.

Observație

Explicație suplimentară
Presupunem că în tabul curent este deschisă pagina 2. În stiva Forward se află paginile 3 și 4. Când se execută comanda Forward, pagina 3 va fi scoasă din stivă și va fi setată ca pagină curentă, iar pagina 2 va fi adăugată în stiva Backward.

2.9 `PRINT`

- Se afișează **circular**, o singură dată, pe aceeași linie (separate de un spațiu), ID-urile tuturor taburilor deschise în browser, începând de la tabul curent, spre dreapta.
- Se afișează, pe o linie nouă, descrierea paginii curente a tabului curent.

2.10 `PRINT_HISTORY <ID>`

- Se afișează, pe linii separate, URL-urile paginilor accesate în tab-ul cu ID-ul specificat. Dacă acest tab nu există, se va afișa mesajul de eroare.
- Ordinea de afișare a URL-urilor începe cu paginile aflate în stiva Forward (de la prima adăugată), continuă cu pagina curentă și se termină cu paginile aflate în stiva Backward (de la ultima adăugată).

Important

Mesajul de eroare este "403 Forbidden".

Acesta va fi afișat pentru cazurile excepționale specificate la operațiile **CLOSE, OPEN <ID>, PAGE <ID>, BACKWARD și FORWARD**.

3 Bonus

BONUS

Se va acorda un bonus pentru implementările care nu au deloc probleme, erori la rularea testelor cu Valgrind.

4 Fișiere de intrare

Datele se citesc din fișierul **tema1.in** și rezultatele se vor scrie în fișierul **tema1.out**.

Fișierele de intrare vor conține, pe prima linie, numărul de pagini care pot fi folosite în test. Urmează apoi câte 3 linii pentru fiecare pagină - prima linie conține ID-ul, a doua linie conține URL-ul, iar a treia linie conține descrierea. **Pagina inițială (default) nu va fi citită - se asigură că toate paginile din fișierul de intrare au ID diferit de 0.**

Pe următoarea linie se află N, numărul de operații care vor fi executate, urmat de N linii ce conțin operațiile descrise anterior.

5 Testarea temei

Temele trebuie să fie încărcate atât pe **checker-ul automat** cât și pe **Moodle**, la secțiunea corespunzătoare. **NU** se acceptă teme trimise pe e-mail sau altfel decât prin intermediul **Moodle**.

O rezolvare constă într-o arhivă de tip **zip** care conţine toate fişierele sursă alături de un **Makefile**, ce va fi folosit pentru compilare, şi un fişier **README**, în care se vor preciza detaliile implementării.

Makefile-ul trebuie să aibă obligatoriu regulile pentru **build** şi **clean**. Regula `build` trebuie să aibă ca efect compilarea surselor şi crearea binarului **tema1**.

Va fi pus la dispoziție un checker prin care veți putea testa corectitudinea implementării. De asemenea, acest checker va verifica existența fișierului README, nu și conținutul, acesta urmând să fie verificat ulterior, manual.

6 Exemple

Exemplu NEW_TAB

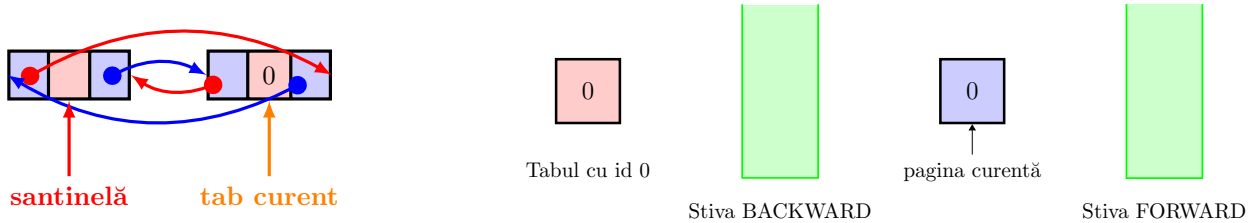
tema1.in

```
(1)     3
(2)     1
(3)     https://www.stanford.edu/
(4)     A Mission Defined by Possibility
(5)     2
(6)     https://www.ox.ac.uk/
(7)     University of Oxford
(8)     3
(9)     https://www.harvard.edu/
(10)    Harvard University
(11)    4
(12)    NEW_TAB
(13)    NEW_TAB
(14)    NEW_TAB
(15)    PRINT
```

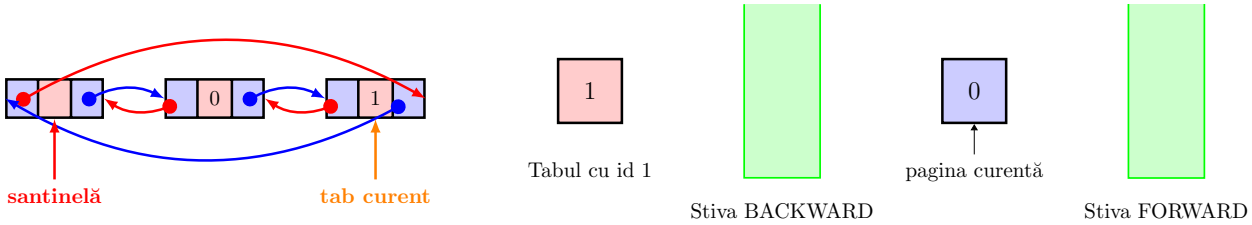
tema1.out

```
(1)     3 0 1 2
(2)     Computer Science
```

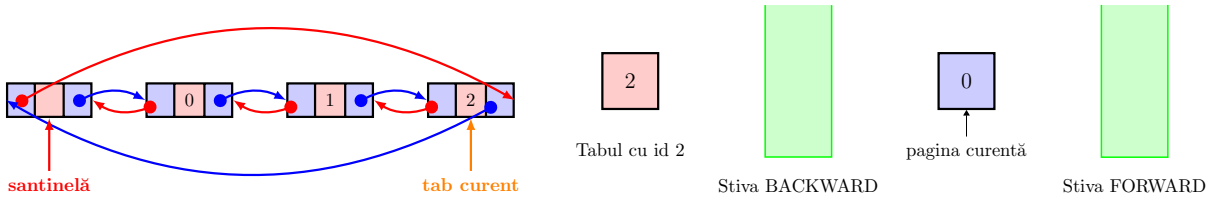
Explicație



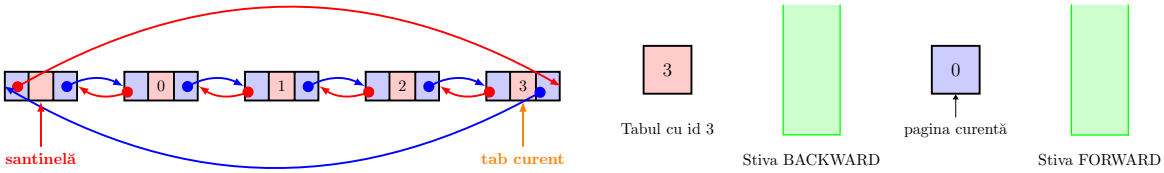
Inițial



(12) NEW_TAB



(13) NEW_TAB



(14) NEW_TAB

(15) Comanda PRINT afișează ID-urile tuturor taburilor începând de la tabul curent și va merge spre dreapta până parcurge toate taburile (linia 1 din output). De asemenea, se va afișa și descrierea paginii curente (pagina cu ID 0) din tabul curent (tabul cu ID 3).

Exemplu CLOSE

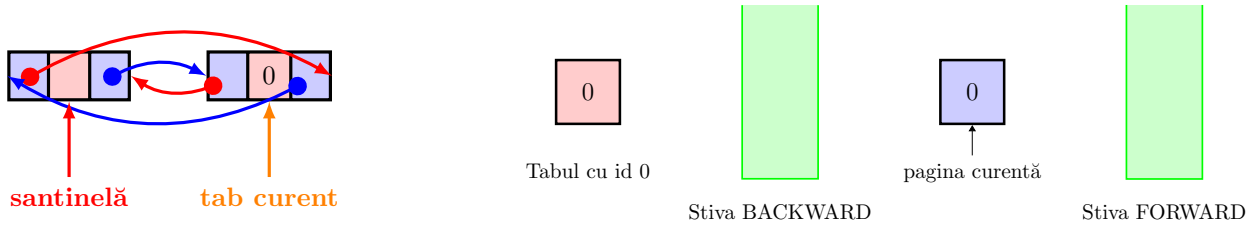
tema1.in

```
(1)     3
(2)     1
(3)     https://www.stanford.edu/
(4)     A Mission Defined by Possibility
(5)     2
(6)     https://www.ox.ac.uk/
(7)     University of Oxford
(8)     3
(9)     https://www.harvard.edu/
(10)    Harvard University
(11)    5
(12)    NEW_TAB
(13)    NEW_TAB
(14)    CLOSE
(15)    CLOSE
(16)    CLOSE
```

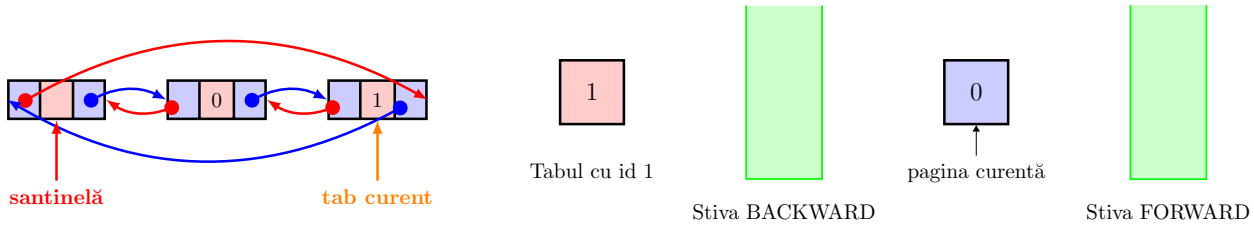
tema1.out

```
(1)     403 Forbidden
```

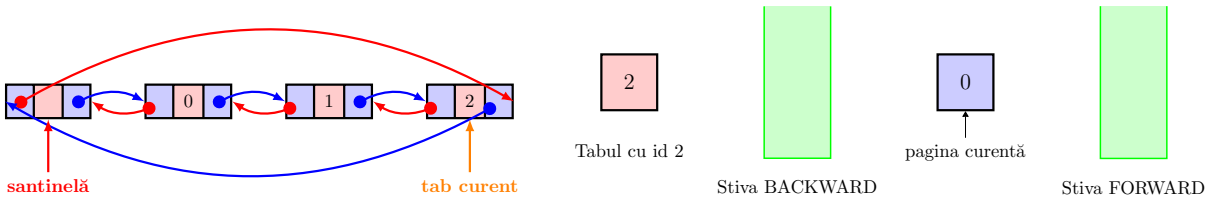
Explicație



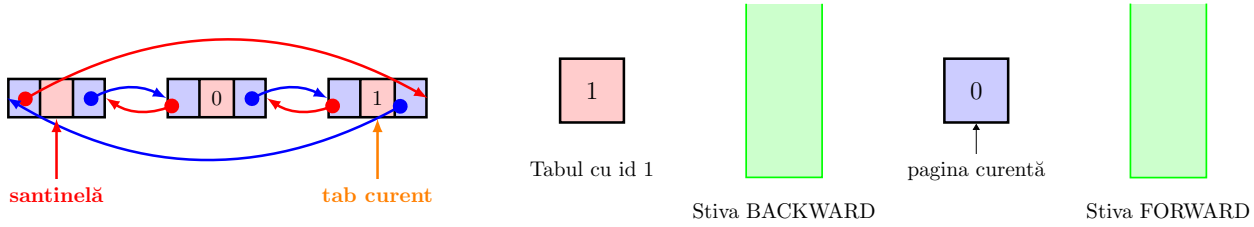
Inițial



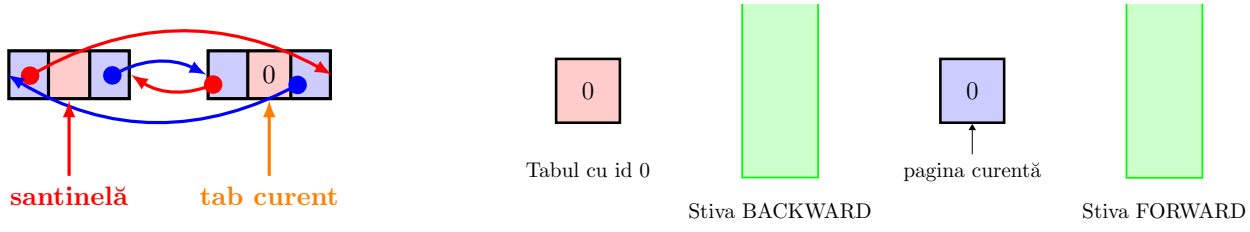
(12) NEW_TAB



(13) NEW_TAB



(14) CLOSE



(15) CLOSE

(16) Comanda CLOSE nu poate fi executată pe tabul cu ID 0.

Exemplu OPEN

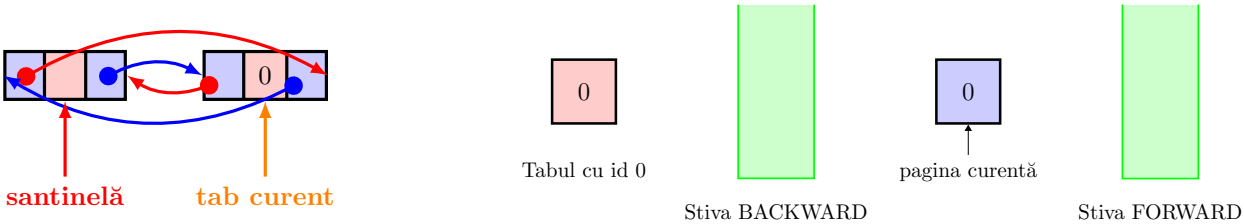
tema1.in

```
(1)     3
(2)     1
(3)     https://www.stanford.edu/
(4)     A Mission Defined by Possibility
(5)     2
(6)     https://www.ox.ac.uk/
(7)     University of Oxford
(8)     3
(9)     https://www.harvard.edu/
(10)    Harvard University
(11)    5
(12)    OPEN 6
(13)    NEW_TAB
(14)    NEW_TAB
(15)    OPEN 1
(16)    PRINT
```

tema1.out

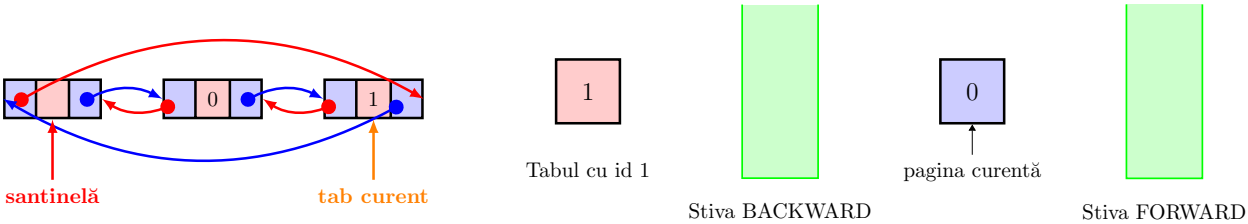
```
(1)     403 Forbidden
(1)     1 2 0
(2)     Computer Science
```

Explicație

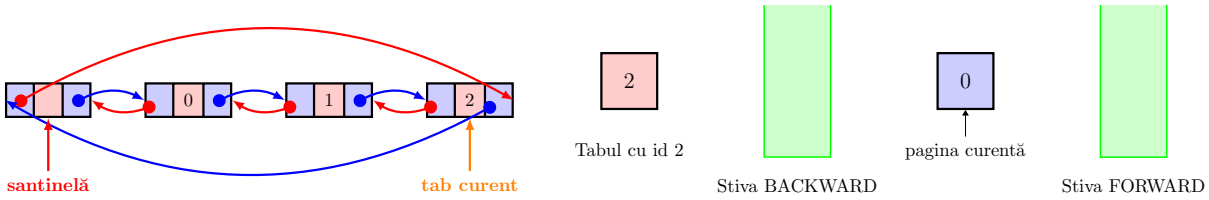


Inițial

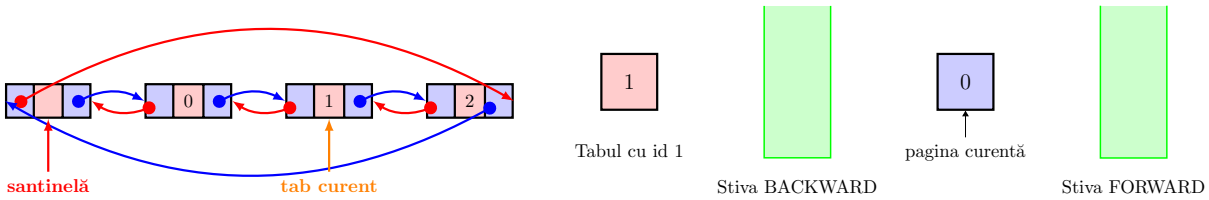
(12) Se încearcă deschiderea tabului cu ID 6, doar că în browser există doar tabul inițial cu ID 0, deci se va afișa mesajul de eroare.



(13) NEW_TAB



(14) NEW_TAB



(15) OPEN 1

(16) În urma comenzii PRINT se vor afișa ID-urile taburilor începând de la tabul 1 și va afișa descrierea paginii care este accesată în tabul curent (în acest caz, pagina implicită).

Exemplu NEXT și PREV

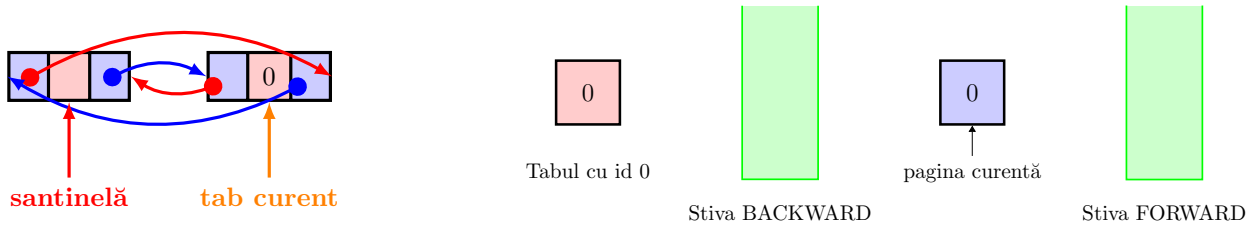
tema1.in

```
(1)     3
(2)     1
(3)     https://www.stanford.edu/
(4)     A Mission Defined by Possibility
(5)     2
(6)     https://www.ox.ac.uk/
(7)     University of Oxford
(8)     3
(9)     https://www.harvard.edu/
(10)    Harvard University
(11)    10
(12)    NEW_TAB
(13)    NEW_TAB
(14)    NEW_TAB
(15)    OPEN 2
(16)    PRINT
(17)    PREV
(18)    PREV
(19)    PREV
(20)    NEXT
(21)    PRINT
```

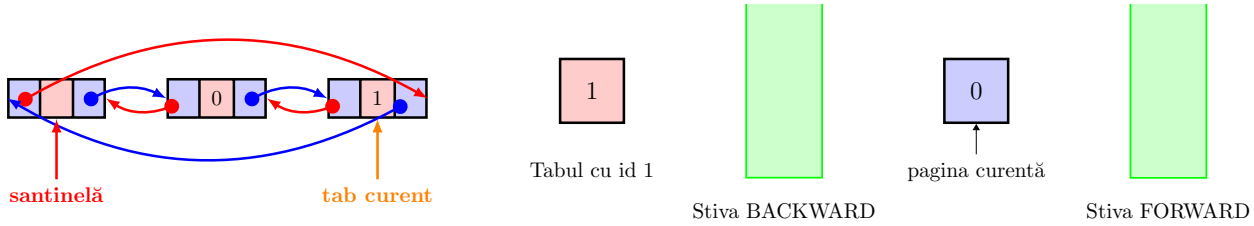
tema1.out

```
(1)     2 3 0 1
(2)     Computer Science
(3)     0 1 2 3
(4)     Computer Science
```

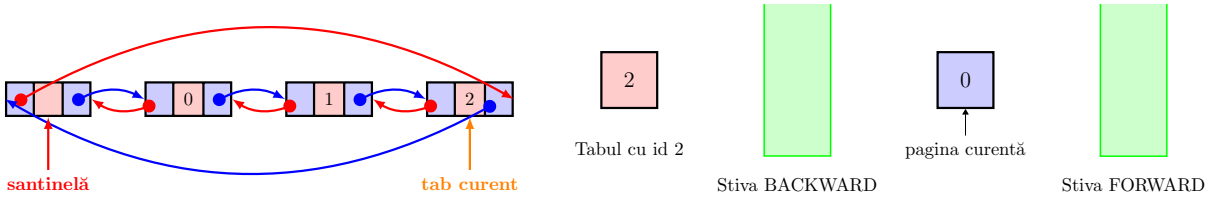
Explicație



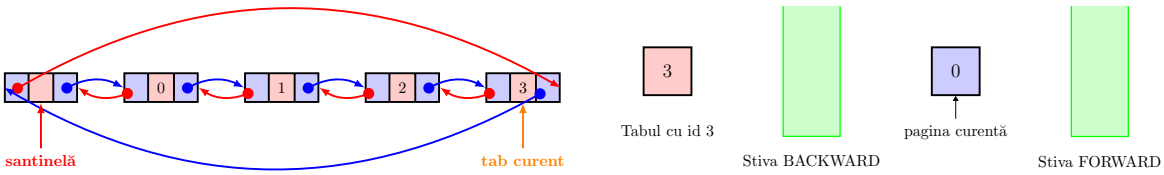
Inițial



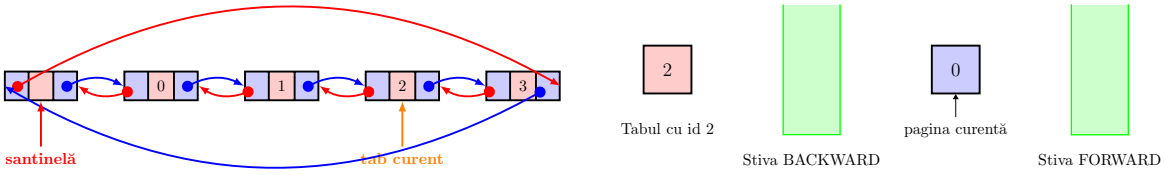
(12) NEW_TAB



(13) NEW_TAB

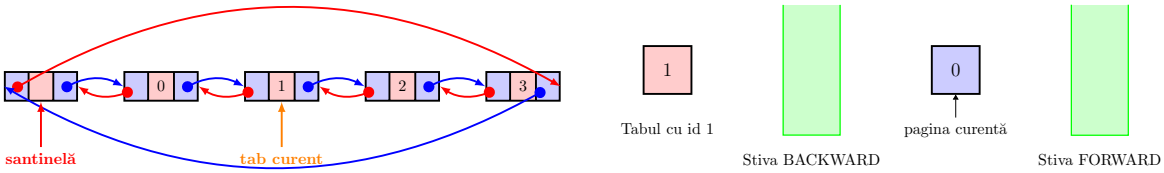


(14) NEW_TAB

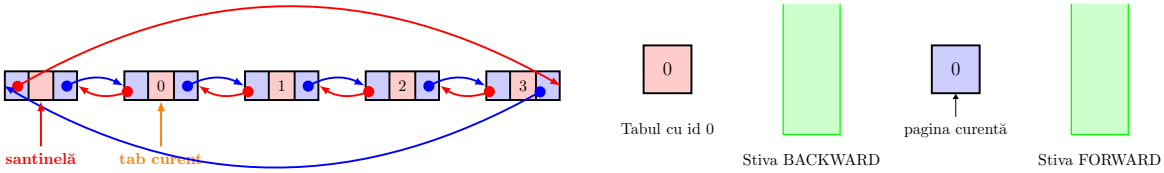


(15) OPEN 2

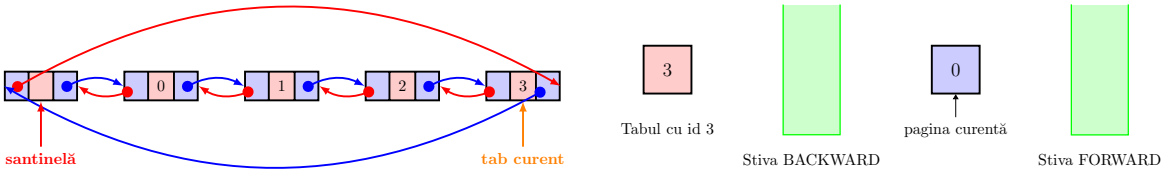
(16) Se afișează ID-urile taburilor începând ID-ul 2. urmate de descrierea paginii curente a tabului cu ID 2.



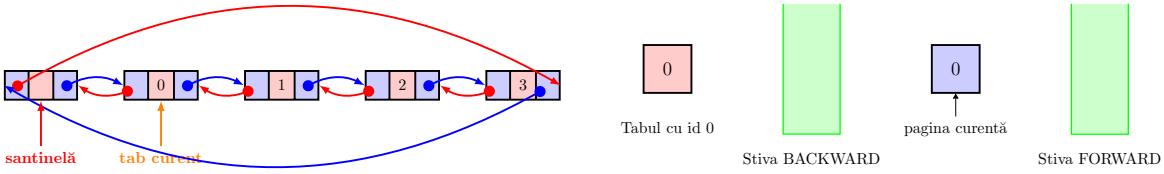
(17) PREV



(18) PREV



(19) PREV



(20) NEXT

(21) Se afișează ID-urile taburilor începând ID-ul 0, urmate de descrierea paginii curente a tabului cu ID 0.

Exemplu PAGE

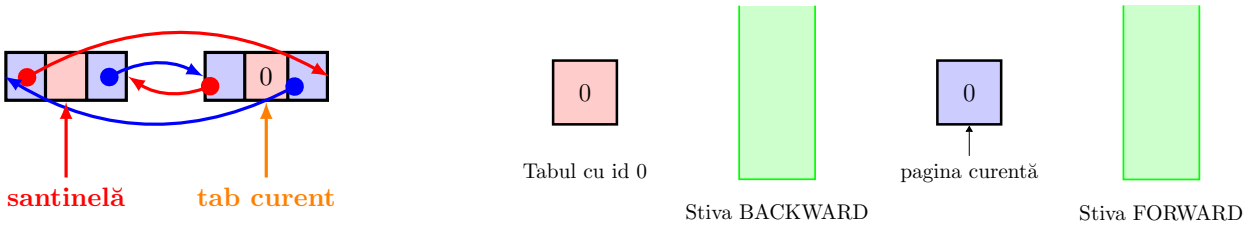
tema1.in

```
(1)     3
(2)     1
(3)     https://www.stanford.edu/
(4)     A Mission Defined by Possibility
(5)     2
(6)     https://www.ox.ac.uk/
(7)     University of Oxford
(8)     3
(9)     https://www.harvard.edu/
(10)    Harvard University
(11)    3
(12)    PAGE 3
(13)    PAGE 2
(14)    PRINT_HISTORY 0
```

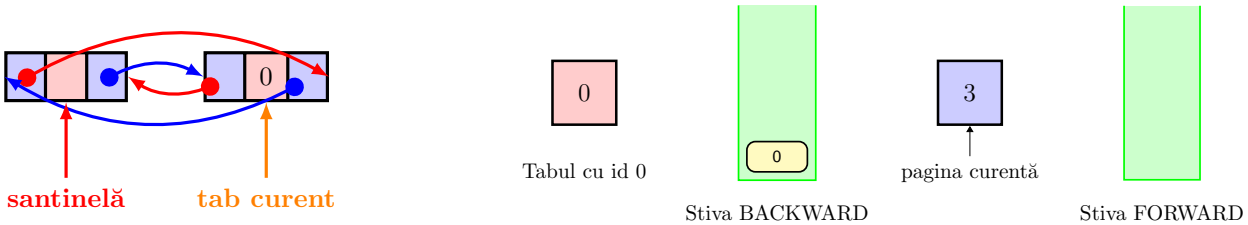
tema1.out

```
(1)     https://www.ox.ac.uk/
(2)     https://www.harvard.edu/
(3)     https://acs.pub.ro/
```

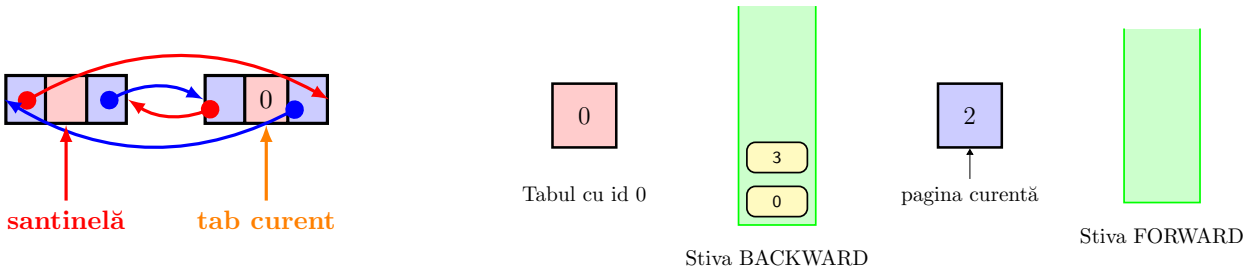
Explicație



Inițial



(12) PAGE 3



(13) PAGE 2

(14) Se va afișa URL-ul tuturor paginilor: prima dată vor fi afișate cele din stiva Forward (goală în acest caz), apoi URL-ul paginii curente (ID 2), urmat de URL-ul paginilor cu ID 3 și 0 din stiva Backward.

Exemplu BACKWARD și FORWARD

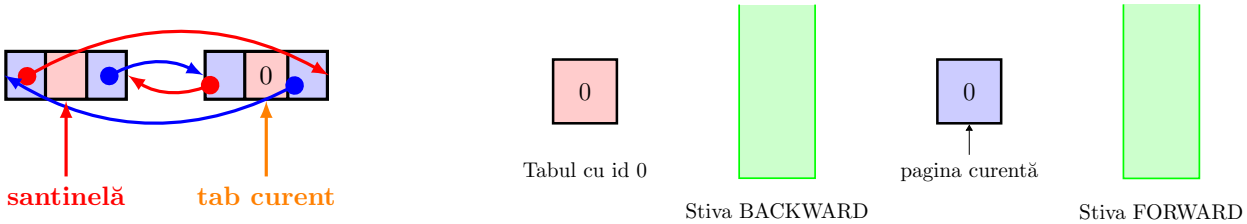
tema1.in

```
(1)     3
(2)     1
(3)     https://www.stanford.edu/
(4)     A Mission Defined by Possibility
(5)     2
(6)     https://www.ox.ac.uk/
(7)     University of Oxford
(8)     3
(9)     https://www.harvard.edu/
(10)    Harvard University
(11)    10
(12)    PAGE 1
(13)    PAGE 2
(14)    PAGE 3
(15)    PAGE 3
(16)    BACKWARD
(17)    BACKWARD
(18)    BACKWARD
(19)    PRINT_HISTORY 0
(20)    FORWARD
(21)    PRINT
```

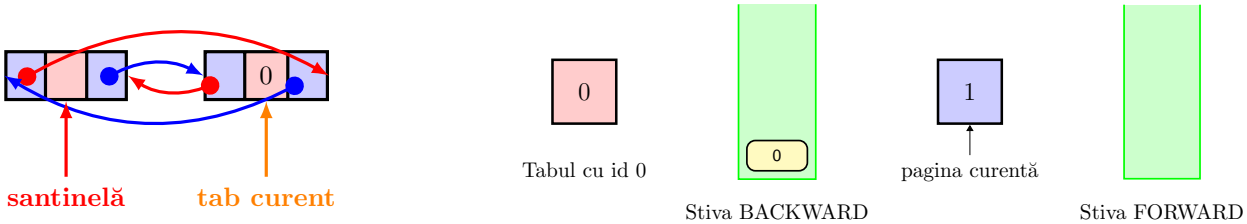
tema1.out

```
(1)     https://www.harvard.edu/
(2)     https://www.harvard.edu/
(3)     https://www.ox.ac.uk/
(4)     https://www.stanford.edu/
(5)     https://acs.pub.ro/
(6)     0
(7)     University of Oxford
```

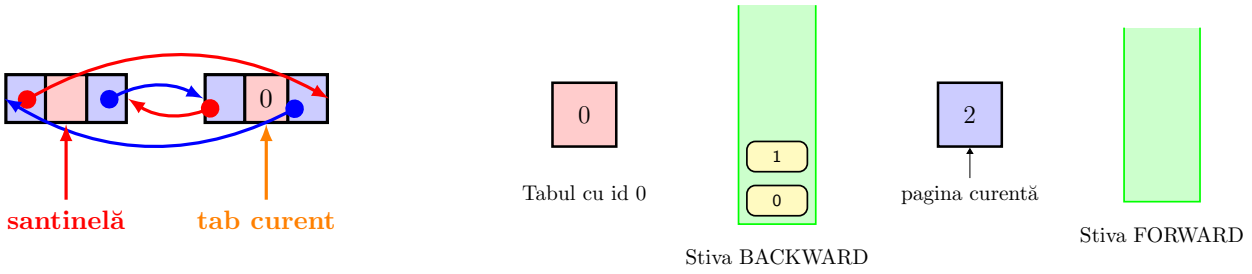
Explicație



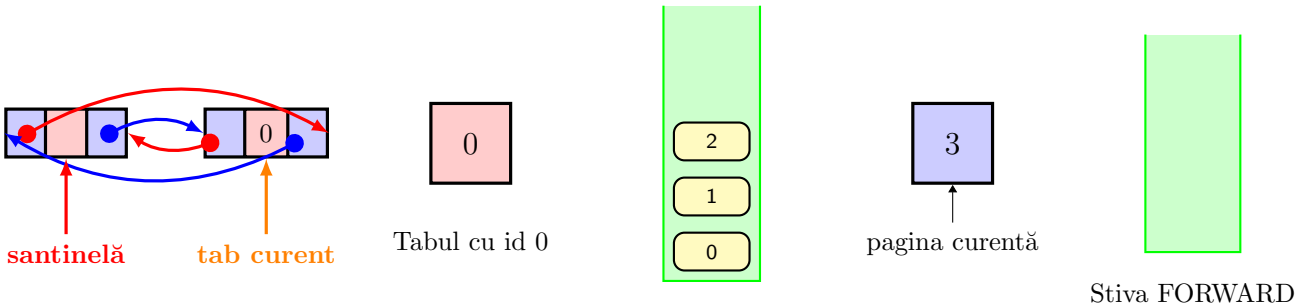
Inițial



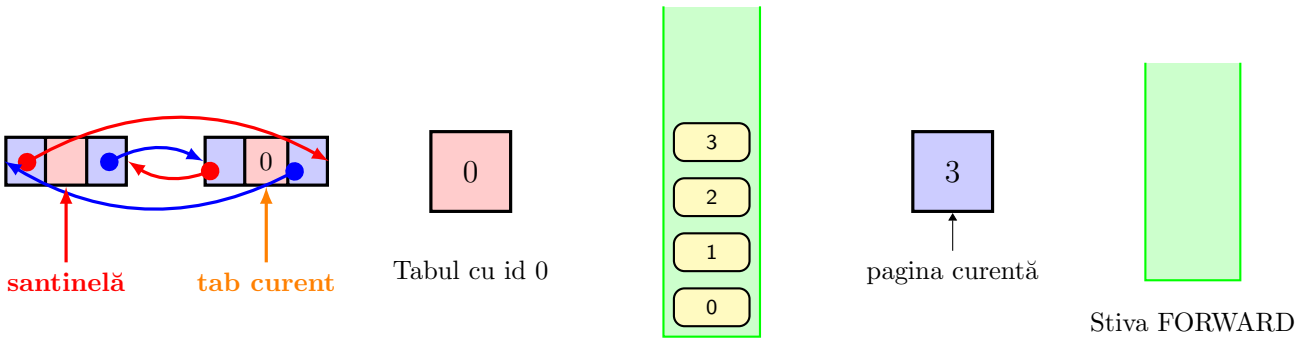
(12) PAGE 1



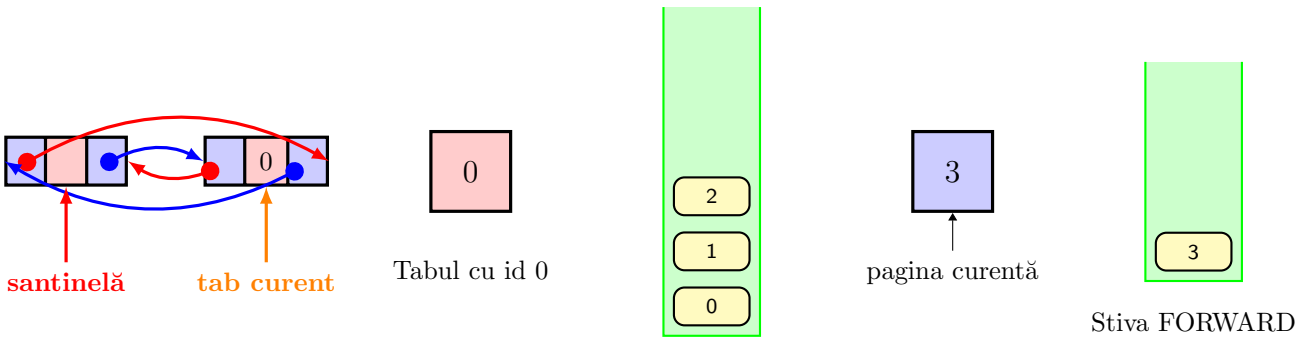
(13) PAGE 2



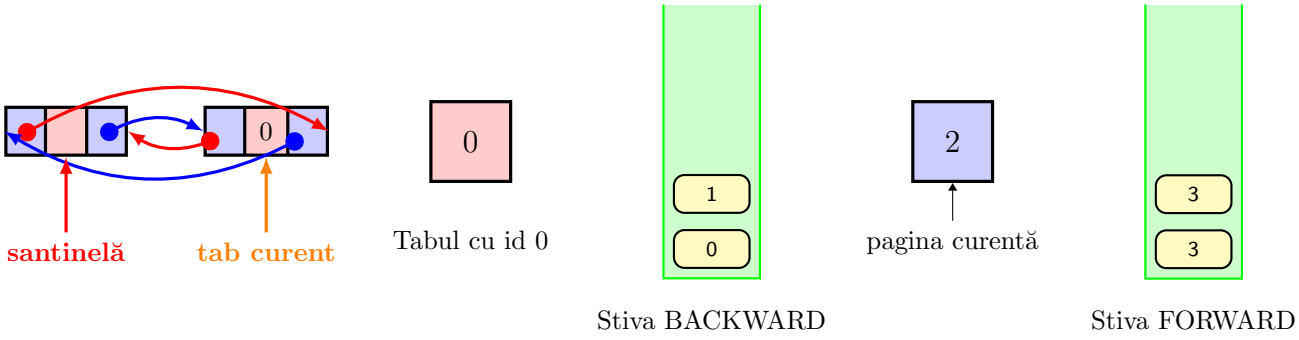
(14) PAGE 3



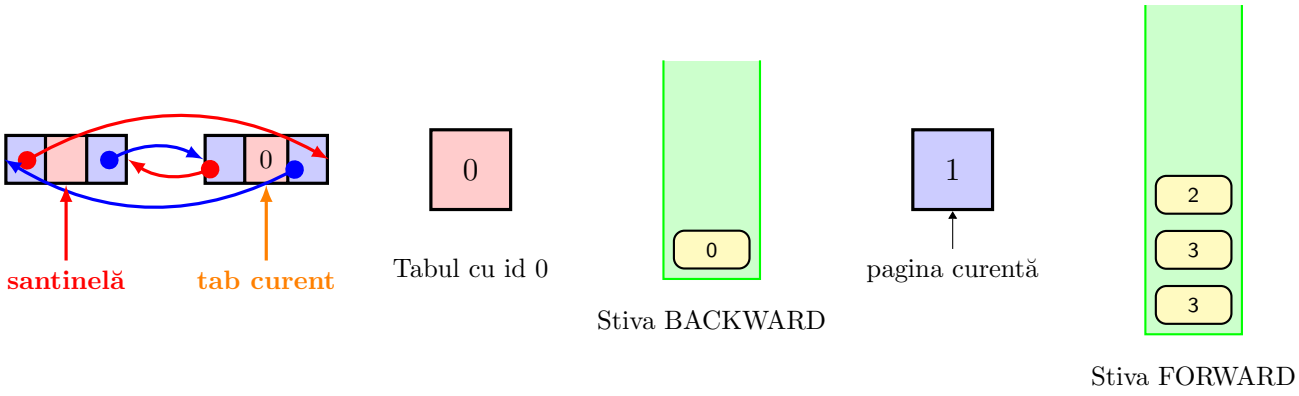
(15) PAGE 3



(16) BACKWARD

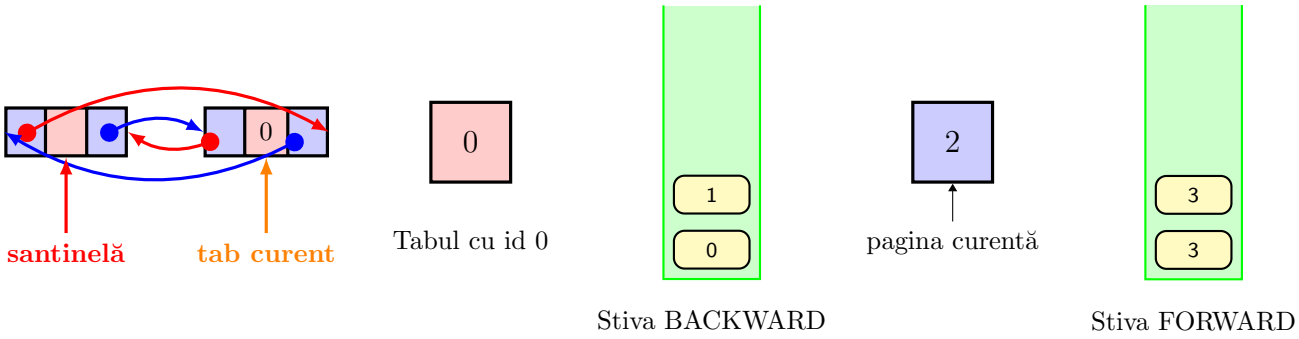


(17) BACKWARD



(18) BACKWARD

(19) Se va afișa URL-ul tuturor paginilor: prima dată vor fi afișate cele din stiva Forward (ID 3, ID 3 și ID 2), apoi URL-ul paginii curente (ID 1), urmat de URL-ul paginilor din stiva Backward (ID 0)



(20) FORWARD

(21) Se vor afișa ID-urile taburilor existente (doar un singur tab în acest caz) și descrierea paginii curente (ID 2).

7 Punctaj

O temă perfectă valorează 100 de puncte. 95 de puncte se vor acorda pentru teste și 5 puncte pentru README. Vor exista atât teste simple, care verifică o operație specifică, dar și teste complexe, în care există majoritatea operațiilor.

În urma corectării manuale, punctajul acordat de checker-ul automat poate fi scăzut cu maxim 15 puncte pentru coding style. De asemenea, conținutul fișierului README va fi verificat manual, iar punctajul obținut pentru README poate fi diminuat.

Punctajul pe teste este următorul:

Cerința	Punctaj
Comenzi `NEW_TAB`	10 puncte
Comenzi `CLOSE / OPEN <ID>`	20 puncte, 10 puncte (CLOSE), 10 puncte (OPEN)
Comenzi `NEXT / PREV`	20 puncte (10 puncte fiecare comandă)
Comenzi `PAGE <ID>`	10 puncte
Comenzi `BACKWARD / FORWARD`	20 puncte (10 puncte fiecare comandă)
Comanda `PRINT`	5 puncte
Comenzi `PRINT_HISTORY <ID>`	10 puncte
README	5 puncte
BONUS (testat cu valgrind)	20 puncte
TOTAL	**120 puncte**

Atenție!

- Orice rezolvare care nu conţine structurile de date specificate **NU** este punctată.
  - **browserul** va fi implementat printr-o structură care va conține o listă dublu înlănțuită circulară cu santinelă și adresa celulei care indică tabul curent;
  - **tabul** va fi implementat ca o structură ce conține un număr întreg, adresa paginii curente accesată în tab și două stive;
  - **pagina** va fi implementată ca o structură ce conține un număr întreg, un șir de maxim 50 de caractere și un șir de caractere de dimensiune variabilă;
  - **Pentru orice altă soluție se vor anula punctajele obținute pe checker-ul automat și nu se vor primi puncte pentru README sau codying style**.
- Descrierea unei pagini se va termina întotdeauna cu un caracter newline.
- **Pentru descrierea paginii trebuie să alocați exact atâta memorie cât este necesară.**
- Toate paginile folosite în teste (mai puțin pagina implicită) vor fi citite din fișierul de intrare. Vor exista maxim 50 de pagini și le puteți salva în orice mod doriți (listă, vector de structuri, etc.).
- **Folosirea variabilelor statice și/sau globale este interzisă!**
- Temele vor fi punctate doar pentru testele care sunt trecute pe checker-ul automat.
- Nu lăsaţi warning-urile nerezolvate, deoarece veţi fi depunctaţi.
- **Tema este individuală! Toate soluţiile trimise vor fi verificate, folosind o unealtă pentru detectarea plagiatului.**