



STIVE

Șl. Dr. Ing. Șerban Radu

Departamentul de Calculatoare

Facultatea de Automatică și Calculatoare



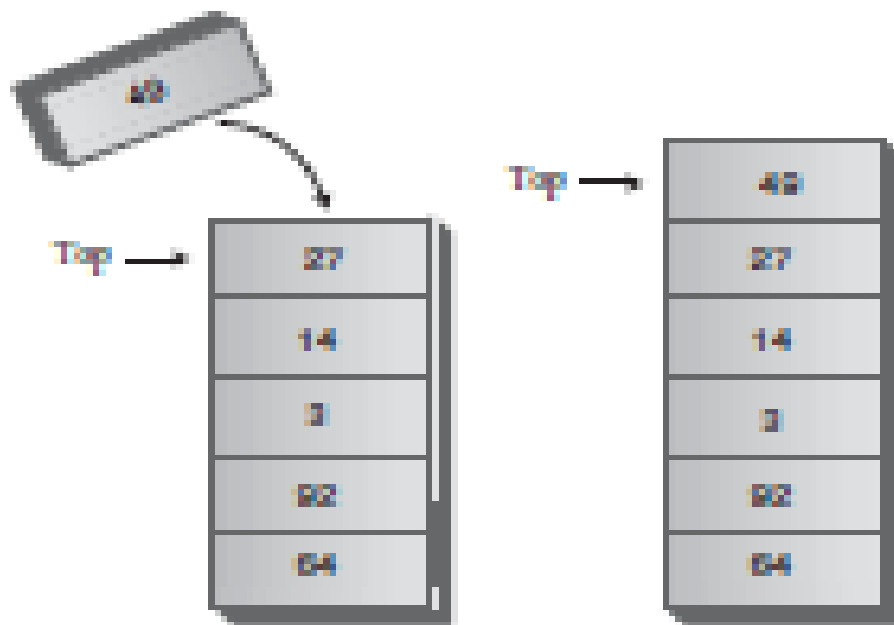
Introducere

- O stivă permite accesul la un singur element: cel care a fost inserat ultimul
- Dacă eliminăm acest element, atunci putem avea acces la elementul de lângă el
- Majoritatea microprocesoarelor au o arhitectură bazată pe conceptul de stivă

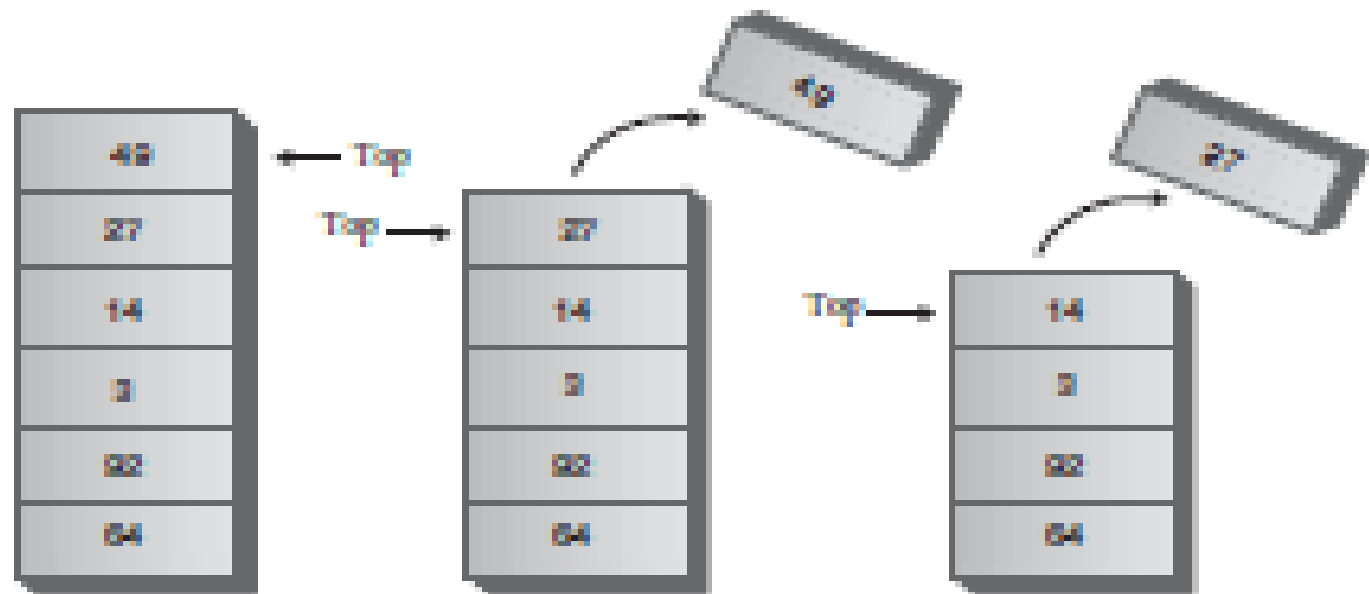


Introducere

- La apelul unei funcții, adresa de întoarcere din aceasta și parametrii funcției se introduc într-o stivă, în timp ce, la revenire, aceste informații sunt extrase din stivă
- Operațiile cu stiva sunt incorporate în microprocesor
- Vezi demonstrația Stack



New item pushed on stack



Two items popped from stack



Operații cu stive

- Inițializare stivă vidă (initstack)
- Test stivă vidă (emptystack)
- Pune un obiect pe stivă (push)
- Extrage obiectul din vârful stivei (pop)
- Obține valoarea obiectului din vârful stivei, fără scoatere din stivă (top/peek)




Implementarea stivelor

- Stivele pot fi implementate în două moduri
 - Static, folosind tablouri
 - Dinamic, folosind pointeri
- Implementarea sub formă de tablou are dezavantajul lungimii finite a stivei, pentru că se declară de la început dimensiunea maximă a stivei

Exemplu de implementare statică

- Se citește un cuvânt de la tastatură, care se introduce într-o stivă, apoi se citește din stivă cuvântul introdus și se afișează pe ecran



```
#include <stdio.h>
#include <string.h>
#include <conio.h>
struct stiva {
    int varf;
    char elem[100];
} s;
```




```
int main() {
    char sir[20];
    int i;
    printf("Introduceti un cuvant\n");
    gets(sir);
    s.varf = 0;
    for (i = 0; i < strlen(sir); i++)
        s.elem[s.varf++] = sir[i];
    printf("Cuvantul inversat este\n");
    while (s.varf != 0)
        printf("%c ", s.elem[--s.varf]);
    getch();
}
```




Exemplu de implementare dinamică

- Se consideră o stivă implementată sub forma unei liste înlănțuite, în care se introduc ca elemente numere întregi și se fac prelucrări asupra elementelor din stivă



```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
typedef int EType;
typedef struct cell {
    EType elem;
    struct cell *next;
} Stack;
```




```
void error(char *msg);
```

```
Stack* initstack(Stack *s) {  
    s = NULL;  
    return s;  
}
```


```
int emptystack(Stack *s) {  
    return (s == NULL);  
}
```

```
Stack* push(EType e, Stack *s) {  
    Stack *nou;  
    nou = (Stack*) malloc(sizeof(Stack));  
    if (nou == NULL) {error("Nu mai exista spatiu  
disponibil");  
        return NULL; }  
    nou->elem = e;  
    if (s == NULL) {nou->next = NULL;  
        s = nou;  
    }  
    else {  
        nou->next = s;  
        s = nou; }  
    return s;  
}
```

```
Stack* pop(Stack *s) {
    ElType e;
    Stack *aux;
    aux = s;
    if (s == NULL) {error("Stiva vida");
        return 0;
    }
    e = aux->elem;
    printf("Se scoate elementul din varful stivei cu
valoarea %d\n", e);
    s = aux->next;
    free(aux);
    return s;
}
```




```
void top(Stack *s) {  
    ElType e;  
    Stack *aux;  
    aux = s;  
    if (s == NULL) {error("Stiva vida");  
        return;  
    }  
    e = aux->elem;  
    printf("Elementul din varful stivei este %d\n", e);  
}
```




```
void printstack(Stack *s) {
    Stack *aux;
    aux = s;
    if (aux == NULL) printf("\n Stiva vida \n");
    else {
        printf("Continutul stivei este: \n");
        while (aux != NULL) {printf("%d\n", aux->elem);
                               aux = aux->next;
                               } } }
```

```
void error(char *msg) {
    printf("\n %s \n", msg);
}
```

```
int main() {  
    Stack *s = NULL;  
    s = initstack(s);  
    if (emptystack(s)) printf("Stiva este vida\n");  
    else printf("Stiva nu este vida\n");  
    s = push(100, s);  
    s = push(200, s);  
    s = push(300, s);  
}
```



```
printstack(s);  
s = pop(s);  
printstack(s);  
top(s);  
s = pop(s);  
printstack(s);  
top(s);  
if (emptystack(s)) printf("Stiva este vida\n");  
else printf("Stiva nu este vida\n");  
getch();  
}
```

Aplicații ale stivelor

- Aplicații în care datele memorate temporar în stivă se vor utiliza în ordine inversă punerii lor în stivă, cum ar fi în memorarea unor comenzi date sistemului de operare (ce pot fi readuse spre execuție), memorarea unor modificări asupra unui text (ce pot fi anulate ulterior prin operații de tip “undo”), memorarea paginilor Web afișate (pentru a se putea reveni asupra lor)



Concluzii

- O stivă permite accesul la ultimul element inserat
- Operațiile principale asupra unei stive sunt introducerea unui element în vârful stivei și extragerea elementului din vârful stivei
- O stivă poate fi implementată utilizând tablouri (array-uri) sau liste înlănțuite