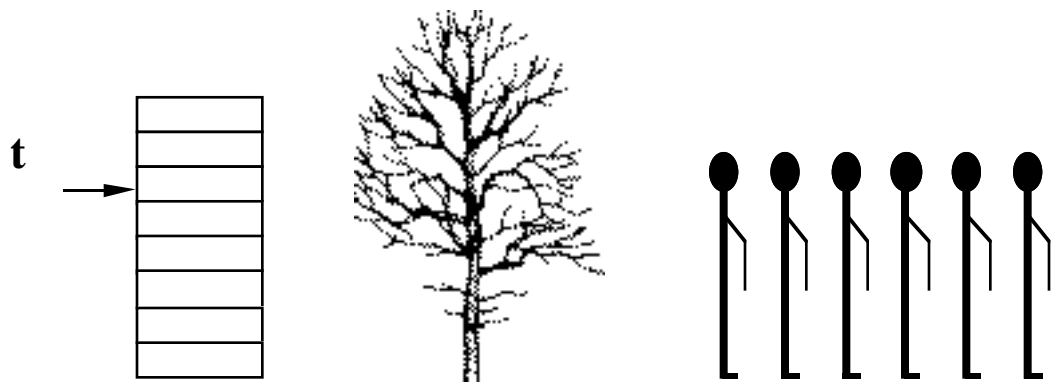
© Itai November 2, 1997

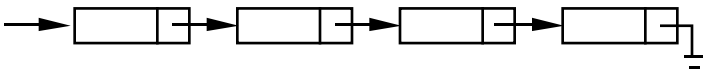
מבני נתונים

מטרות הקורס:

- **ההבדל בין מבנה נתונים ומימושיו.**
- **הכרות עם מבנים ומימושים יעילים.**
- **אנליזה ופיתוח כלים לניתוח יעילות.**
- **בחירת מבנים מתאימים כשלב בפיתוח.**



$$\begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \\ a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} \\ a_{4,1} & a_{4,2} & a_{4,3} & a_{4,4} \end{bmatrix}$$



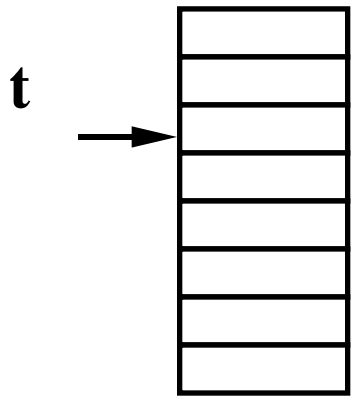
<u>מבנים</u>: מערך, רשימה מקושרת, מחסנית, תור, עץ בינארי, ערימה,...

<u>שימושים</u>: חיפוש, מיון, מימוש שפות תכנות, ארגון קבצים,...

מבנה נתונים =
אוסף של פעולות עם השפעות הדדיות, הנשמרות לאורך חיי המבנה.

מימוש של מבנה נתונים:
נוהל לכל פעולה של המבנה,
המשתמש במבני נתונים "פשוטים" יותר.

מהי מחסנית?
מערך עם מצביע



הוספת איבר: `t=t+1; a[t]=x;`
ראש המחסנית: `a[t]`
הוצאת איבר: `t=t-1;`

לא!

זהו מימוש של מחסנית בעזרת מערך ומצביע.

מחסנית כמבנה נתונים

Last In – First Out :LIFO

אחרון נכנס - ראשון יוצא.

מחסנית מוגדרת ע"י הפעולות:

create( ) - מחזיר מחסנית ריקה חדשה.
push(s,x) - הכנסת איבר בעל ערך x למחסנית s.
top(s) - ערך האיבר בראש המחסנית s
(s לא משתנה).
pop(s) - הוצאת האיבר שבראש המחסנית s.
is_empty(s) - האם המחסנית s ריקה?

כללים:

top(s) - מחזיר את האיבר האחרון שהוכנס.
pop(s) - מוציא את האיבר האחרון שהוכנס.
אפשר לבצע top,pop רק על מחסנית לא ריקה.
מיד אחרי create(s) , is_empty(s) נותן ערך true.
לאחר ביצוע push ואחריו pop, המחסנית לא משתנה.

כל מימוש חייב לאפשר את הפעולות ולקיים את הכללים.
יש טענות שנובעות רק מהכללים, <u>בלי תלות במימוש</u>.

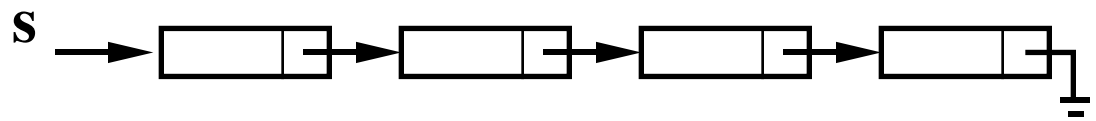
<u>דוגמא:</u>

```
s = create( );
push(s,17);
pop(s);
print is_empty(s);
```

מה יודפס?

מימושי מחסנית

1. מימוש בעזרת מערך: כפי שהוסבר
2. מימוש בעזרת רשימה מקושרת (linked list):



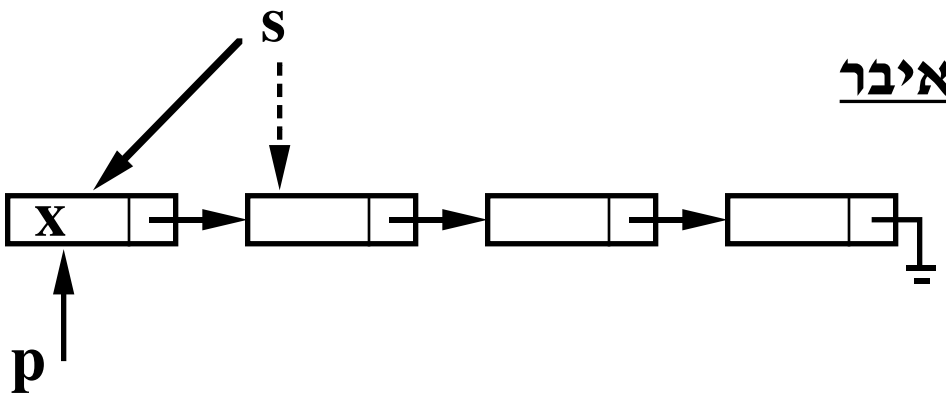
הכרזה

```
#define NULL 0
typedef struct node{
   int info;
   struct node *next;
} NODE;
typedef NODE *STACK;
```

מחסנית חדשה

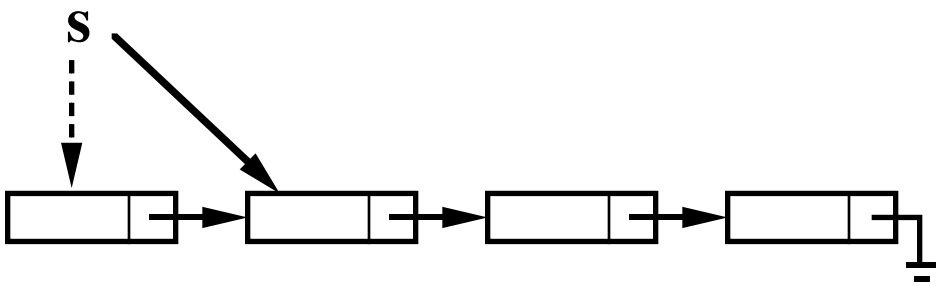
```
STACK create(void){
   return NULL;
}
```

הכנסת איבר




```
STACK push(STACK s, int x){
   NODE *p;
   p=malloc(sizeof(NODE));
   p->info = x;
   p->next = S;
   return (STACK)p;
}
```

הוצאת איבר




```
STACK pop(STACK S){
   return S->next;
}

int top(STACK S){
   return S->info;
}
```

חסר:

- בדיקת תנאי גבול
  - האם s אינה ריקה לפני pop, top?
- בזמן pop ביצוע free לצומת העליון.

המשתמש במבנה:
- **מכיר את הפעולות והשפעתן**
- **אינו מכיר את פרטי המימוש**

<u>הסתרת מידע</u> כשיטת פיתוח:
מאפשר החלפת המימוש, מבלי לפגוע בשימושים.

כיצד נעריך את איכות המימוש?
- **ניתוח יעילות:**
  - **זמן - מספר הצעדים**
  - **מקום - כמות הזכרון הדרושה.**

- **פשטות.**

חישוב זמן של אלגוריתם

זמן הריצה של אלגוריתם A לקלט x: $t_A(x)$.

זמן למקרה הגרוע (worse case)

$$t_A(n) = \max\{t_A(x) : |x| = n\}$$

<u>דוגמה</u>: נתונה רשימה באורך n ובה a אחד ו-$n-1$ b-ים.
מצא את מקומו של ה-a הראשון.

קלט i:

$$x_i = \underbrace{b\ldots b}_{i-1}a\underbrace{b\ldots b}_{n-i}$$

זמן החיפוש לקלט i:

$$t_A(x_i) = i$$

$$t_A(n) = \max_{i\le n}\{t_A(x_i)\} = \max_{i\le n}\{i\} = n$$

<u>מקרה ממוצע</u>: יהי $X_n = \{x : |x| = n\}$
אם לכל $x \in X_n$ אותה ההסתברות (התפלגות אחידה)
הזמן הממוצע: בהנחה שיש N איברים בגודל n:

$$\mathrm{E}\, t_A(n) = \frac{1}{N} \sum_{x \in X_n} t_A(x)$$

<u>דוגמא</u>: רשימה כמו קודם

$$E\, t_A(n) = \frac{1}{N} \sum_{i=1}^{n} t_A(x_i) = \frac{1}{n} \cdot \frac{n(n+1)}{2} = \frac{n+1}{2}$$

<u>במקרה הכללי</u>, ל-x הסתברות $P(x)$

$$\mathrm{E}\, t_A(n) = \sum_{x \in X_n} P(x) \cdot t_A(x)$$

בדוגמא הקודמת

הסתברות הקלט ה-i: $p(x_i) = \frac{1}{n}$

זמן ממוצע: $E\, t_A(n) = \sum_{i=1}^{n} p(x_i) t_A(x_i) = \frac{1}{n} \cdot \frac{n(n+1)}{2} = \frac{n+1}{2}$

דוגמה: כמו קודם, אך יש k a-ים ו-$n-k$ b-ים
x_i = ה-a הראשון מופיע במקום i.

$$p(x_i) = \binom{n-i}{k-1} \Big/ \binom{n}{k}$$

$$E\, t_A(n) = \sum_{i=1}^{n} p(x_i) t_A(x_i) = \frac{n}{k+1}$$