

23/10/2012

## לומפיאיה - הרצאה 1:

37/106, 13° - 16° צחצחים, מלח, נזיר גוף: אין נזיר גוף

www.little-lisper.org: ליט'

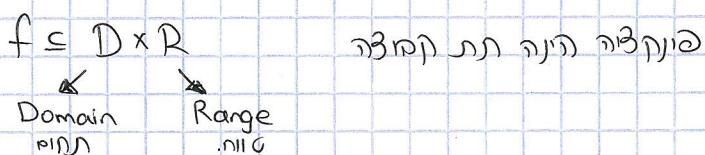
מונט ווילט: ג'יימס קראטמן

### עכירות פונקציונלי (Functional)

### תכנות קינטוטי (Imperative)

(\*) הטי לא תזכיר פונקציונלי, זה לא מגדיר לנו מה פונקציונליות

פונקציונליות



Strlen : string  $\rightarrow$  int

פונקציונליות Strlen

strlen ("Hello")  $\rightarrow$  5

("Hello", 5)  $\in$  Strlen

פונקציונליות printf

printf : string  $\rightarrow$  int

printf ("Hello")  $\rightarrow$  5

("Hello", 5)  $\in$  printf

פונקציונליות strlen ו-func printf מגדירים פונקציונליות

func printf (str, out) = (length str, out  $\wedge$  str);  
printf : (string x output)  $\rightarrow$  (int x output)

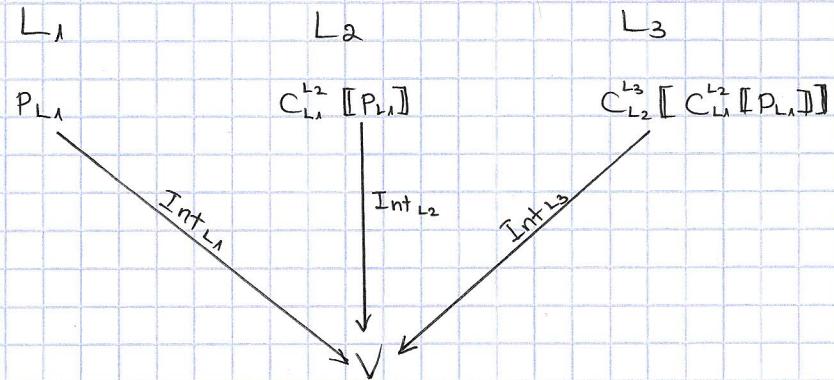
פונקציונליות printf

printf : (string x output)  $\rightarrow$  (int x output)

$\Rightarrow$  func printf (str, out) = (length str, out  $\wedge$  str);

(\*) רין גאנפער ניידום מתקנת אינפלריאט ג'תנער פונקציונלי.

## סימולציית כבויות



כבר:  $C_L^L$  - גומחה נובן  $L_i$  בסיס  $L_j$ .

$L_i$  נובן ב-  $\text{Int}_{L_i}$  -  $\text{Int}_{L_i}$

$V$  - פסוי.

הוכחת שכתחזק נובן  $L_i$  -  $P_{L_i}$

$\text{Int}_{L_1} [P_{L_1}] = \text{Int}_{L_2} [C_{L_1}^{L_2} [P_{L_1}]] = \text{Int}_{L_3} [C_{L_2}^{L_3} [C_{L_1}^{L_2} [P_{L_1}]]]$ : הוכחה נכונה (Valid)

(\*) סימולציה - מבחן יirk

(\*) תרשים -

נובן נובן סימולציה יפה בפונקיה וווער.

תמונה נובן נובן מוקדם או מוקדם כדי גי' זי' וווער.

## חישובות אקדמיות

Static vs. Dynamic ①

לעומת  
פראגטיון  
(כגון, דיבור, "מי" תחילה ובסוף)

Early Binding vs. Late Binding ②

אלאג' קורין -  
Early  
(בהתחלת סדר)  
(Assembly)  
או נאכ' גראונט -  
Late  
(בסוףgame)  
(Scheme)

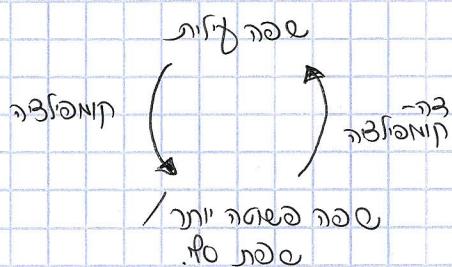
... הַלְוָה קָנֵת יְמִינֶךָ

```
# define C (...)  
for (n=100; i=0; c) {  
    printf ("Hello \n");  
}
```

二〇一三年

? #define C (i++ < n) ← כוונת הפעלה של פונקציית כרזה

כארהה הער, כיון שהוא מלהלך ונימצא גור יפה יותר.



• אוניברסיטאות ומוסדות מחקר מודדים נזקם בפער בין הנקודות שקבעו ונקודות שקבעו סטטיסטיקאים : Bootstrapping

פָּאָגָה לְתַּרְבִּיתָן קָרְבָּה: גַּוְיָה רַבָּה תְּמִימָה הַגְּדוֹלָה יֵצֶר מִקְרָבָה.

(היפותזה:  $\mu_{\text{פונקציונליות}} = \mu_{\text{טיפוס}}$  או הnull hypothesis:  $\mu_{\text{פונקציונליות}} \neq \mu_{\text{טיפוס}}$ )

.fc.exe (0)) .( TC - FEN )

סימני נסיעה מודולריים (FC) הנקראים גם סימני פונקציית כוח (fc.function keys) נמצאים בדרך כלל בחלק העליון של מסך המחשב.

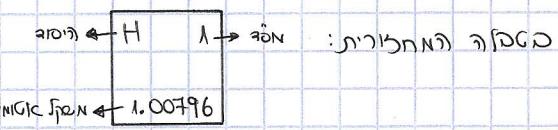
הנחיות לארון נזק נזק ותשלוחם:

> (molecular-weight "Al<sub>2</sub>(CO<sub>3</sub>)<sub>3</sub>")

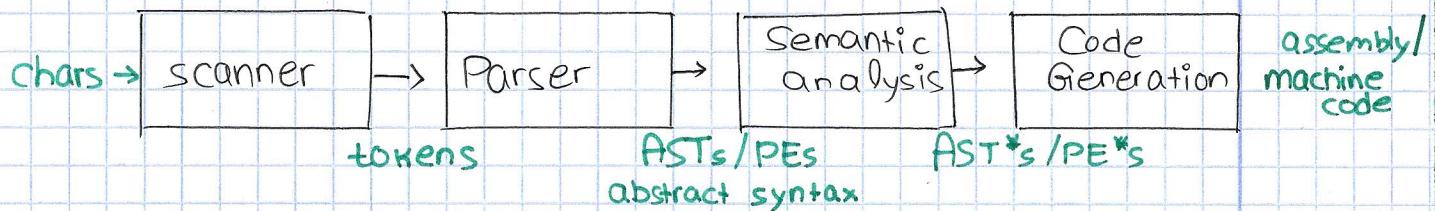
Scheme → גבירות → גבירות → גבירות

תְּמִימָנָה (תְּמִימָנָה) נִזְבְּנָה בְּ 6.022 \times 10^{22} אֵינְזֶרֶס - כַּאֲמִתָּה וְכַיְדָה.

$$[(\text{Al}_2(\text{CO}_3)_3] = 2 * [\text{Al}] + 3 * ([\text{C}] + 3 * [\text{O}])$$



2 גָּתָה 3 גָּתָה



= הינה הקופה שהיא מכנה כalgo נלנעות. token

נוסף סכਊר, ה-tokens-ם הם ה-JSON, סינון ו-EOF...

לפניהם נקבעו tokens-וּ מילים נזקcid ננה ומלוקט נובאם בזיהוי.

ע"י סקנער (scanner) ניתן לסקור

• סדר גיון נאיבר, אוניברסיטה נורווגית (NUPI) ו-UNIVERSITY OF OSLO (אוניברסיטת אוסלו)

: Scanner - ייח גלאי

12..3 מוכר ①

$$0 \neq$$

6.02e23

לפוזה כי שולחן ו כיס

char \* s = "moshe";

200

20

20

```
printf("Hello");
```

רשות Java -> Scanner ← Java -> קידום וטיהר טקסט

. will see following input.

tokens to  $\beta_3$ : output.

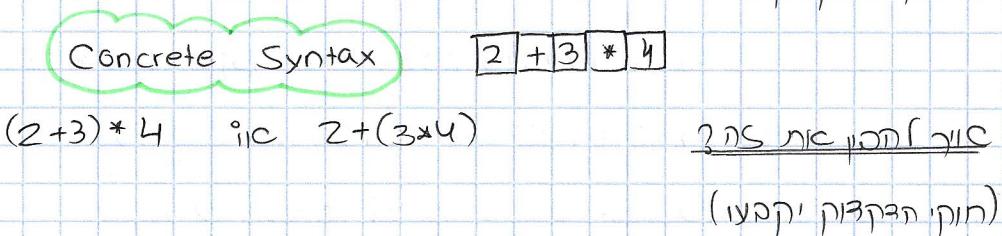
$2["moshe"]$

$* (2 + "moshe")$  - סעיפים. 'ס' - סעיף נעלם C-ה

$T A[i] \Rightarrow * (A + i \cdot \text{sizeof}(T))$

## Parsing

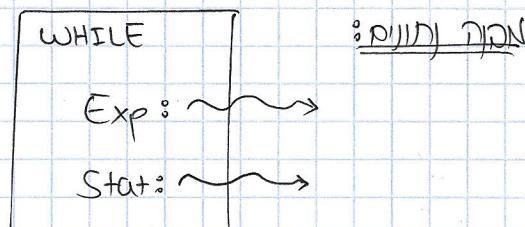
כל חוקי תרגול



abstract syntax

$\text{Stat} \rightarrow \underline{\text{while}} \ L \ \underline{\text{exp}} \ R$

$\text{Stat}$

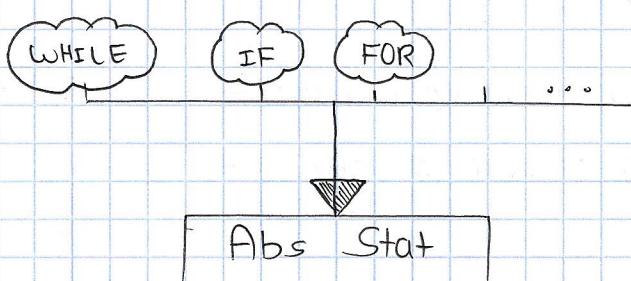


abstract parser  
tree

כרגע מוכן (הכו נטה) parser - תרשים (תורין כונן ופונקציונלי).

(הנשלך שחקים וכו') כו. IF (עליהם, יסויירנו).

parser  $\rightarrow$  סעיפים



## סימנטיק איסוף (parser-semantic analysis)

• מילויים.

• נכליג if / for / while

• מילויים (filling)

if() foo();

else goo();

if-else ↑ ↓

וליתן:

# define foo()

do {

} while ↑

foo(); מילויים יתבצעו; מילויים יתבצעו;

## Semantic Analysis

( $\lambda(x)$ )

( $x (\lambda(\ast))$ )

( $x (\lambda(x))$ )

( $x (\lambda(\lambda))$ )

warning: מילויים יתבצעו ← while ( $x=4$ ) {

}

- סינטקטית נזק -

- מילויים יתבצעו -

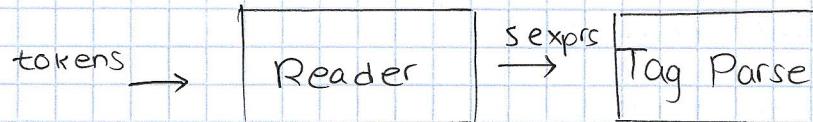
. type inference

- ציון זר -

## סבב 2

parser (o scanner) (o reader)

: pick up plain parser , Scheme ->



Numeric computing

הוילג:

Symbolic computing (List)

\* נקי (clean) סימולטור

\* מילוי (filling)

. garbage collector - אוניברסלי לירוק

\* אוניברסלי

>(read)

4

4

>(read)

(

3 4

)

(3 4)

. data -> se parser (o) read

sexprs (o) data -> se reader

Bool #t #f

Char #\a ...

Symbol "abc"

Nil ()

Pairs (2.3)

Vectors #(2 3 \*)

. Sexprs (3) (o) reader (o) Scheme ->

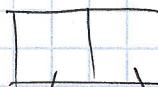
.sexps - ה-ספנס יוכן יוכן וירטואלי.

.sexps - ספנס יוכן יוכן וירטואלי (טבז) Reader ->

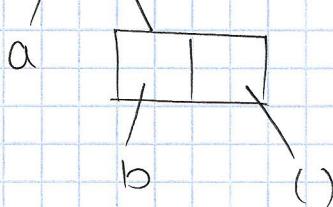
... F, Lambda -> טבז פירסינג Tag Parse

### : Scheme -> List

(a) : פונקציית ספנס  $\Leftarrow (a.( ))$  פון



(a b c)  $\Leftarrow (a.(b.c))$  פון



> '(a b c) = > (quote (a b c))

int \*A = {4,9,6,3,5,1};

ולא רק שטח הפון יוכן וירטואלי (טבז) יוכן וירטואלי (טבז) קומפקטי

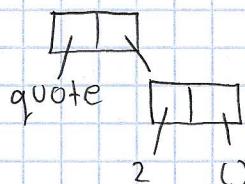
בקבוצה

> (+ '1 '2)

3  
↓

נקה (טבז)  
הכuous קומפקטי  
לעומת 1

וכי זו איזה היען כוונת 2, כיון שהפון 2" (quote 2)



> '(1 (+ 2 3) 4)

(1 (+ 2 3) 4)

> ` (1 (+ 2 3) 4) (back quote)

(1 5 4)

> `(1, @reverse '(a bc)) 2)

(1 cba 2)

.ןוילען ניכרונן @

(def fact

(lambda (n)

(if (zero? )

`(\* ,n ,(fact (- n 1))))

(...זינקן אטמי)

(def foo

(lambda (e)

(cond ((pair? e)

`(cons ,(foo (car e))

,(foo (cdr e))))

((or (null? e) (symbol? e)) | e)

(else e))))

> (foo '(1 2))

לפנינו מיצג של פונקציית

(1 2)

(cons 1 (cons 2 '()))

(lambda (e)

$((\lambda(x) \cdot (x, x)))$

$((x(x) \cdot (x, x)))$

: אוסף כל הפעולות

13/11/2012

# 3 נס 3 נס

טבליות ונתונים

AOC

Reader

Parser

Anomaly of Quoted AOC

!ונן אם, אם בפיהו יכולנו לאן side-effect

set-car!, set-cdr!, ... ורשות להזין Scheme ->

```
(def last-pair
```

```
(x e)
```

```
(if (null? (cdr e))
```

```
e
```

```
(last-pair (car e))))
```

למעשה מוכיחים last-pair -

```
(def foo
```

```
(lambda ()
```

```
(let ((x (list 'he'said:)))
```

```
(set-cdr! (last-pair x)
```

```
(list 'ha 'ha))
```

```
x)
```

>(foo)

(he said: ha ha)

>(foo)

(he said: ha ha)

```
(def g00
  (λ ()
    (let ((x '(he said:)))
      (set-cdr! (last-pair x)
                 (list 'ha 'ha))
      x)))
```

> (g00)

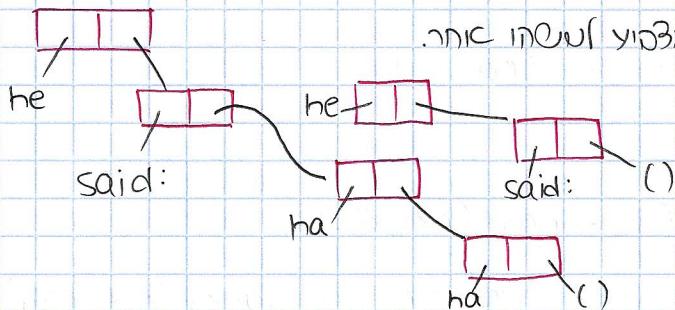
(he said: ha ha)

> (g00)

(he said: ha ha ha ha)

⋮

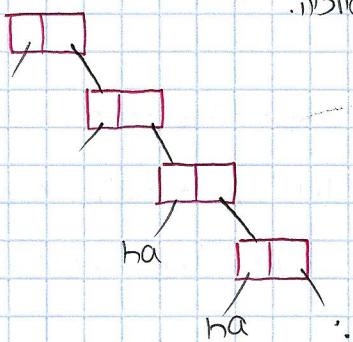
quette (o עיתק) \*



...nic means you x, FOO -> CLOOLO \*

\* כוונת o ל nic ב x הוא כוונת o (g00 -> (he said)) / (he said) כוונת o ל nic ב x הוא כוונת o (he said: ha ha) כוונת o ל nic ב x הוא כוונת o (he said: ha ha ha ha)

הכוונה הוגה נטול נזיר



(def b00

(λ ()

(let ((x '(he said:)))

(set-cdr! (last-pair x)

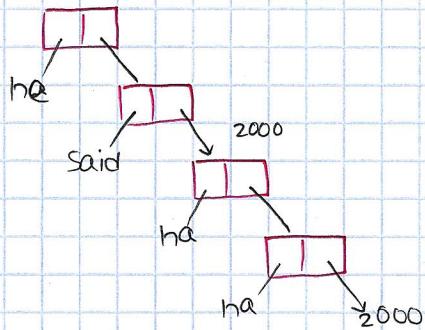
x)) '(ha ha))

:ANSWER \*

הכוונה  
ב00

(he said : . #1 = (ha ha. #1))

## היכל נסעה נאצ'י



(const) char \*name = "Mayer";

:c nədə

(read-only) **ronly** גדרה ימינה page -> דפיים

.Segmentation Fault נתקין בטעות נספחים לאטם של היכל מילויים

...ex write, read -> sign known nice analysis, debugger ->

ויכן ה'ר'ו'ן כהנוצר במלואו יתגלו ויכן ג'לון צב נערתו צו.

int \*A = {4, 9, 6, 3, 5, 1}; (n3f3mip nwd y31, 1000) : n301j kn313 \*

illegal סנייה גורם副產品. sy Side-effectoric Scheme -D \*

## 2 $\pi f G N$

...ption in S expression to print -

הקלוטן (הקלוטן) Reader Ennemy. backtracker mic

getS expression, getSExpressions

• 11337

- (  )

- # ( \_\_\_\_\_ )

## S expressions

rparen py (20) vector

- (Single-quote) ✓

nice back-quote → גייר גיבוב נטול רשיון

- (back-quote)

.PIN Scanner C1 \*

- (unquote) )

- (unquote-sp0icy) , @

File → tokens

String → tokens

(String → tokens)

(String → tokens "# \n space")

: space in

\* "(4 9 6 3 5 1)"



:NICKNAME

(tokens → sexpressions (string → token (

)

((4 9 6 3 5 1))

(4 9 6 3 5 1)

: (הנימוקים הינם יתרכזים בפונקציית פיקוח)

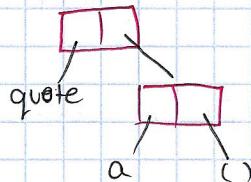
\* "(a.b)"

↓

(a.b) ...μρpn

## Reader

tokens  
" 'a'"  
((sq) (sym a))  
'a'



"(quote a)"  
((lp) (sym quote) (sym a) (rparen))  
'a'  
sexp.

unquote -> backquote      אן דען לילך זיכר

- ; - regular expressions

: comments

(זיהוי scanner → מחר). Scanner -> מחר. Scanner → מחר כו C

- # ; <Sexpr>

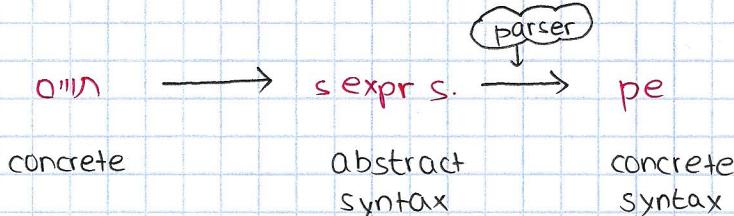
הnic מילג שמיינס כי Reader -> מחר כו קומטט. \*

קומטט עיגן פלט, (comment) Sexpression-> מחר גב

.icon Sexpression -f

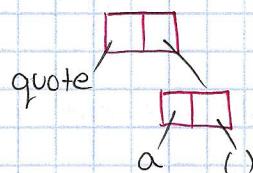
> (if)  
error

-1 Reader icl logic sic expression-הן תחוי icl icl מילויים הוראות הנקה.



• נחקק תעלוי (רכינה) Scheme  $\in$  וולטן הינה הוא מתקיימת נ-expr. וולטן על parser.

\* "a" → ((sq) (symbol a)) → 'a' → (const a)  
 ↗ tokens reader parser



\* "(if)" → ((Ip)(sym if )(rp)) → (if ) → error

1

כ) פ' עיך גולמי ד'

(if test dif) ①

(if test dit dif ) ②

\* "if" → ((sq)(sym if)) → 'if' → (const if)

\* "if" → ((sym if)) → if → error

\* if הינו יונק כנראה עכו ב- parser יעה לא ימתקין

גנרטור כו� נייח שאנוכה.

\* "(quote if)" → ((l0p)(sym quote)(sym if)(r p)) → 'if → (const if)

void func:

(set! x 3) (def x 4)

## : Side-effects

11

#<void> : נינז'ו

( ୦୩୩୧୮ ୯୮ )

> (if #f 'moshe)  
> void נונצ'ר כטבנ'ר. (ס' 10)

> (def v (if #f 'moshe))

> (list v)

(#<void>)

void חישוב\_טבלה(ויליאם כהן נסמן) \*  
הנוסחה היא:

void print () {cout << "הענ" ;}

\* void חישובים יישר וירטואליים #<void>

•inic ini0si void יי יу

30.10.2019: 13:00

①  $(\lambda (a b c) e)$  : נציג גורם כפלה כפונקציונלי

② ( $\lambda$  (a b c.r) e) :  $\lambda$  y<sub>1</sub>y<sub>2</sub>y<sub>3</sub>y<sub>4</sub> y<sub>1</sub>(y<sub>2</sub> y<sub>3</sub> y<sub>4</sub>.r) e

③ (x r e)

① ( $\lambda$ -simple  $(a\ b\ c)$   $\llbracket e \rrbracket$ ) ווריאנטה נסיעה  
parser ↗

② ( $\lambda$ -opt  $(a\ b\ c)$   $r$   $\llbracket e \rrbracket$ )

③ ( $\lambda$ -var  $r$   $\llbracket e \rrbracket$ )

(def list

( ✓ ✓ ✓ ))

list 171331701 \*

$\begin{pmatrix} & & \\ & & \end{pmatrix} \begin{pmatrix} 0 & 1 & 2 & 3 \\ a & b & z & 3 \end{pmatrix}$

- pair?  $O(1)$

- list?  $O(n)$

: rest -> ըլքուն ուշ լու համար օպտ-լամբա հակ  
symbol /improper list /proper-list

(def foo

(> (e ret-pro ret-imp ret-symbol)

(cond (pair? e)

✓ (FOO (cdr e)

ret-pro

( $\lambda$  (s a))

(ret-imp (cons (car e) s) a)) c

(> ())

(ret-imp (list (car e))(cdr e))) ))

((null? e) (ret-pro))

((symbol? e) (ret-sym))

((else (error))))

(cond \_\_\_\_\_

: FOO - Տ ԱԿԻՊԵ ԱՅՐՈՒՅՆ

\_\_\_\_\_  
\_\_\_\_\_  
((cond (pair? e)

(eq? (car e) 'λ)

(pair? (cdr e))

(list? (caddr e)))

(FOO (cadr e))

( $\lambda$  () `(&-simple, (cadr e), (parse (caddr e))))

( $\lambda$  (sa)`(&-opt,s,a,(parse (caddr e))))

( $\lambda$  () `(&-var , (cadr e), (parse \_))))

(a b c . d)

⇓

$s \equiv (a \ b \ c) \quad a \neq d$

( $\lambda$  <arg>  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
)

(cond ((test? \_\_)  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_) )

\* כנראה תמיינן הינה  $\lambda$  begin ו'ן מילויים נזקניים  
מצביעים על הינה בפונקציית begin יתנו לנו גורם אחד

(def beginify

:begin)

( $\lambda$  (es)

(cond ((null? es) void-object) +

((null? (cadr(es)) (car es)

(else `(begin ,@(es) )))))

בנין גוף  
void-object

:parse - נגזר פואט לארט שפה מילויים נזקניים ←

(parse (beginify (caddr e)))

# לניכר נ-פנס - Parser

and

(and)  $\Rightarrow \#t$

- : original of and copy.

(and 4)  $\Rightarrow$  4

- Dynamic  $\perp$  pd and rig.

(and A B)  $\Rightarrow$  (if A B #f)

macro expansion

- Biogenic 2 pf and 7pf.

(and A B C)  $\Rightarrow$  (if A (if B C #f) #f) : o((UN)C 3 pf) and .

לכט הילפה (להרכין נאחו ו-1 מילון).

$$[(\text{or } \psi_1 \ \psi_2 \ \dots \ \psi_n)] = (\text{or} \ (\ [\psi_1] \ \dots \ [\psi_n]))$$

Or

(or A B)

: macro expansion יי' או - ר תווים קבצים יי' גו

~~(if A #t  
(if B #t #f))~~

: 10.11-1

(if A A  
      (if B B #f))

folg - 2:

( let ((x A)  
      (y (λ() B)))  
  (if x x (y))))

$$\left. \begin{array}{l} \\ \end{array} \right\} \Rightarrow \boxed{(\lambda(x.y) \text{ if } x.x(y))}$$

: 3 π'ογ

## Cond 4

מבחן כתוב ופונקציית `else` מוגדרת ב-

(cond (T<sub>1</sub> E<sub>11</sub> ....)  
       (T<sub>2</sub> E<sub>21</sub> ....)  
       :  
       (else E<sub>1</sub> ....))

: Cond. נור נור

לעומת `if` : macro expansion  $\neq$  cond - func

(if T<sub>1</sub> (begin E<sub>M</sub> ....)  
      (if T<sub>2</sub> (begin E<sub>21</sub> ....)

(if T<sub>n</sub> (begin E<sub>1</sub> ....)  
          (begin E<sub>1</sub> E<sub>2</sub> ....)) ....)

## quasiquote 4

ריבר 2 ביר בירן - נסלה מיל' ג'ר - ספלסן עג'ז: עג'ז:

• Unquote-splicing - ספlicing עוקץ לא-טכני --US?

```
(define quotify
  (lambda (e)
    (if (or (null? e) (symbol? e))
        (list 'quote e)
        e)))
```

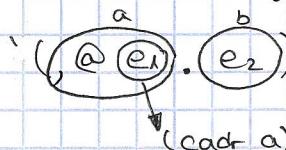
```

(define qq
  (lambda (e)
    (cond ((ug? e) (cadr e))
          ((us? e) (error "—"))
          ((pair? e)
           (let ((a (car e))
                 (b (cdr e)))
             (cond ((us? a) `(append , (cadr a) , (qq b))))
                  ((us? b) `(cons , (qq a) , (cadr b)))
                  (else `(cons , (qq a) , (qq b)))))))
          ((vector? e) `(list->vector , (qq (vector->list e))))
          (else (quotify e))))))

```

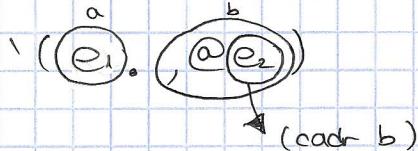
## תקנות גאנדי והתקב:

$a = \text{@e}_1$  # תרמו א' כבש ערך (א? נס. a) (\*\*)



נַחֲלָה בְּנֵי קָרְבָּן

$b = \text{@ } e_2$  הינו נושא של  $b$  נושא של  $e_2$  (אנו?  $b$ ) (\*\*)



נו הנקראת נאורה ב':

הנה הינה נושא מילוי שיכלנו לתרגם (\*\*\*)

עפיהו ה-16 נקבעו סדרי גיבובם על פי אופן הפעלה.

: backquote      ז"ק גוונא מ"ר והקווינט פאר ? Parser השאלה מ"ר מ"מ

(( quote ? e ) .... ))

(( backquote? e) (parse (qq (cadr e))))

תְּמִימָנָה

letrec ↗

לכדי גזירות פולינומיאליות נקבע  $\text{let-}\alpha$  בפונקציית  $\text{let}$ .

$$\begin{array}{l}
 (\text{let } ((\text{abs } (\lambda(x) \\
 \quad (\text{if } (\text{negative? } x) \\
 \quad (-x) \\
 \quad x)))) \\
 \\ 
 (+ (\text{abs } x) (\text{abs } y)))
 \end{array}
 \quad \Rightarrow \quad
 \begin{array}{l}
 ((\lambda(\text{abs}) \\
 \quad (+ (\text{abs } x) (\text{abs } y))) \\
 \quad (\lambda(x) \\
 \quad (\text{if } (\text{negative? } x) \\
 \quad (-x) \\
 \quad x))))
 \end{array}$$

כִּי בְּעֵת גַּם־בָּרֶךְ כָּל־עֲמָדָה וְעַמְּדָה

לפיכך:  $\int_{-N}^N f(x) dx = \lim_{n \rightarrow \infty} \sum_{k=1}^n f(x_k) \Delta x$

הוכיח נסרכותה רקורסיבית מנגן ופיזיiolוגית כORTHOGONALITY

```
(define fact  
  (lambda (n)  
    (if (zero? n)  
        1  
        (* n (fact (- n 1)))))))
```

תמחוץ (ונហוק יותר) fact פְּרִזְמִינָה (פְּרִזְמֵן) קיימן fact גוף תקין fact.

```
F = (\lambda (fact)
      (if (zero? n)
          1
          (* n (fact (- n 1))))))
```

הנה קווות כיוון שמדובר בfact) ? fact of F nic מונחים בנוסף

$$((F \text{ fact}) \emptyset) = 1$$

$$((F \text{ fact}) \ n) = (* \ n \underbrace{(\text{fact} \ (-n \ 1)))}_{(n-1)!}) = n!$$

...now 'ij' as fact  $\Leftarrow$  (F fact) = fact - e if so  $\Rightarrow$

fact - ת责任编辑 הינה יפה נושא לדיון. מינכם:

פונקציה  $f$  היא נטוי נרחבת אם  $f(D) \subseteq N(f)$

( $\text{f(x) = 0}$   $\text{for } x \in \text{B} \setminus \text{A}$ )  $f \subseteq D \times R$

$$\forall g : \underbrace{(Fg) = g}_{\text{מתקיים } F \subseteq g} \rightarrow \underbrace{\text{fact} \subseteq g}_{\begin{array}{l} \text{מתקיים fact} \\ \text{בנוסף } \text{fact} \subseteq g \\ \text{ומתקיים } g \subseteq g \end{array}}$$

(fact  $\in$   $\{ \text{fact}, \text{zero?}, \times, - \}$ )  $\Leftarrow$  מערך גורם  $F$  מוגדרת (פונקציית פאקט)

$$(g = \text{fact} \cup \{<\text{zero?}, \times, ->\}) \rightarrow g \in \{ \text{func} \}$$

$$((F g) \emptyset) = 1$$

הוכיחemos  $g$  מוגדרת נכון

$$((F g) n) = (* n (\underbrace{g \ (- n \ 1)}_{\text{העתקת הערך} \ n \ \text{ל} \ (- n \ 1)})) = n!$$

הוכיחemos  $g$  מוגדרת נכון  
 $((F g) k) = k!$   
 $k=0, \dots, n-1$   $\forall k$

:  $G$  מערך גורם  $G$  מוגדר נכון

$$\begin{aligned} G &\equiv (\lambda (\text{fact} \ n)) \\ &(\text{if } (\text{zero?} \ n) \\ &\quad | \\ &\quad (* \ n \ (\text{fact} \ \text{fact} \ (- \ n \ 1)))) \end{aligned}$$

$$((G G) \emptyset) = 1$$

הוכיחemos  $G$  מוגדר נכון

$$(G G n) = (* \ n \ (\underbrace{G \ G \ (- n \ 1)}_{\text{העתקת הערך} \ n \ \text{ל} \ (- n \ 1)})) = n!$$

:  $G$  מערך גורם  $G$  מוגדר נכון

```
int F (void* g, int n) {
    if (n == 0) return 1;
    else return n * ((int (*) (void*, int)) g)(g, n-1);
}
```

הוכיחemos  $F$  מערך גורם  $F$  מוגדר נכון (פונקציית פאקט)

:  $H$  מערך גורם  $H$  מוגדר נכון

$$\begin{aligned} H &\equiv (\lambda (\text{fact})) \\ &(\lambda (n)) \\ &(\text{if } (\text{zero?} \ n) \\ &\quad | \\ &\quad (* \ n \ (\boxed{(\lambda (\text{fact} \ \text{fact})) \ (- n \ 1)}))) \end{aligned}$$

:  $F$  מערך גורם  $H$  מערך גורם  $H$  מוגדר נכון (פונקציית פאקט)

$$((H H) \emptyset) = 1$$

הוכיחemos  $H$  מוגדר נכון

$$((H H) n) = n!$$

כבר רציתנו גורם גנרי בפונקציית הולכה; נזכיר כי גורם גנרי הוא:

$$\begin{aligned} (***) \quad & \text{fact} \stackrel{?}{=} (\lambda \text{ H} \text{ H}) \\ & = ((\lambda(x) (F(x x))) \\ & \quad (\lambda(x) (F(x x)))) \\ & = (F (F (F \dots)) \end{aligned}$$

(א) הוכיחו  $x$  מ- $\text{H}$  מ- $\text{G}$ . גורם גנרי הוא גורם גנרי.

$$(\lambda(a) ((x x) a))$$

נוכיח  $\text{H} \vdash (***) \rightarrow (x x)$  מ- $\text{G}$ .

$$\text{הוכחה של } \text{G} \vdash (\lambda(x) ((x x)))$$

$$\text{הוכחה של } \text{G} \vdash (\lambda(a) ((x x) a))$$

$$\text{הוכחה של } \text{G} \vdash (\lambda(a_1 a_2) ((x x) a_1 a_2))$$

⋮

$$(***) \quad \text{הוכחה של } \text{G} \vdash (\lambda(s) (\text{apply} (x x) s))$$

נוכיח  $\text{H} \vdash (***) \rightarrow (\lambda(x) (\text{apply} (x x) x))$ , כלומר,  $\text{H} \vdash (***) \rightarrow (\lambda(x) (\text{apply} (x x) x))$ .

נוכיח  $\text{H} \vdash (\lambda(x) (\text{apply} (x x) x)) \rightarrow (\lambda(x) (\text{apply} (x x) x))$ .

נוכיח  $\text{H} \vdash (\lambda(x) (\text{apply} (x x) x))$ .

(define Y

$$(\lambda(F) ((\lambda(x) (F (\lambda(s) (\text{apply} (x x) s)))) \\ (\lambda(x) (F (\lambda(s) (\text{apply} (x x) s)))))))$$

(define fib

$$(Y (\lambda(f) (\lambda(n) (\text{if} (< n 2) \\ n \\ (+ (f (- n 1)) (f (- n 2))))))))$$

ב-פונקציית גנרי

הוכחה של  $\text{H} \vdash (\lambda(x) (\text{apply} (x x) x))$  :

even - 1 odd - פונקציית גנרי

: (প্রিন্ট করে আবারো লেকচার মেমুন-মিস্টি) letrec - ফর্মা

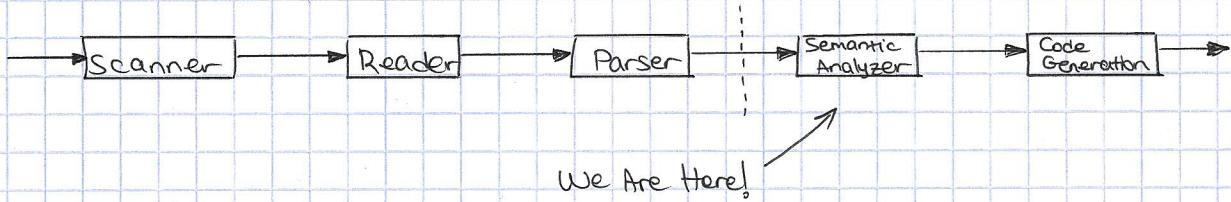
(letrec (( $f_1 (\lambda \quad)$ )  
( $f_2 (\lambda \quad)$ )  
 $\vdots$   
( $f_n (\lambda \quad)$ ))  
exp<sub>1</sub>  
exp<sub>2</sub>  
:  
)

$$F_n = (\lambda (f_0 \dots f_n) f_n)$$

$$F_0 = (\lambda (f_0 \dots f_n) \text{ exp}_1 \text{ exp}_2 \dots)$$

$$f_1 = (\lambda(n) \# f)$$

$$f_1 \equiv (\lambda(f_0 \dots f_n) (\lambda(n) \# f))$$



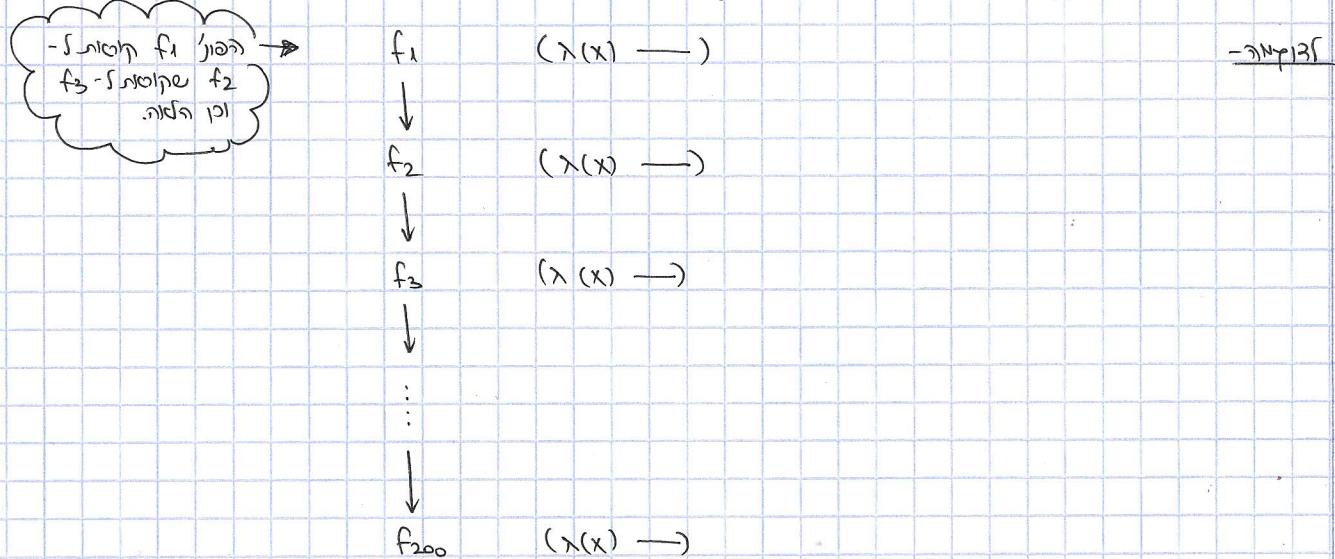
אנו גוּקָה

הכט' כפער נספחים לפרטיה - בז' ג' ינואר (push) ו-בז' ג' ינואר (pop).

הטטי. סדר גיירין זו מושלמת

የኢትዮ-ካናዳሪያውያን በተመለከተው እና የሚገኘውን ስራውን ተመልከተው

: (גִּנְעָלָה, נֶסֶת, מִזְבֵּחַ) Dynamic Scoping → (לְפָנֶיךָ) (לְמִזְבֵּחַ)



•  $\chi$  גורם ניטרואני "גלאס"  $f_{200} \sim 1$

נה יתף כלוא רותח? • הנטמן.

- (אלאן הינו יודע למד זרנוקה) (אלאן מילא יפה נושא הנושא בראן ראה).

```
(define foo  
  (lambda (n)  
    (if (zero? n)  
        x  
        (+ 1 (foo (- n 1)))))))
```

```
(let ((x 5)
      (foo 1000000)))
```

$$\begin{array}{l} n = 0 \\ \vdots \\ n = 10^6 - 1 \\ n = 10^6 \\ x = 5 \end{array}$$

Dynamic Scoping  $\rightarrow$  Se příkaz vydává - užívání názvu

$\Leftarrow$  כתוב במאמר גנרטיבי ומי-הומינין

(יש לנו שאלות, אך לא מושגנו מלהזכיר אותן).

"...deep binding" = Deep Binding = DB

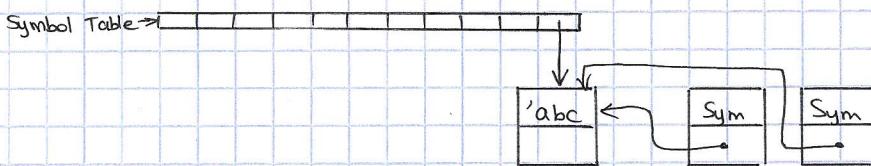
← כrho Ci נינוח מה בוויז'ן (ויז'ן) יי'. כיבז רון פטרו מעת?

סמלים - Symbol Table - אינטראקציית Scheme ->

(Symbol).סימן ערךם ① SECTION 2- SYMBOLS Symbol Table

جواب

## השלמה נוספת (2)

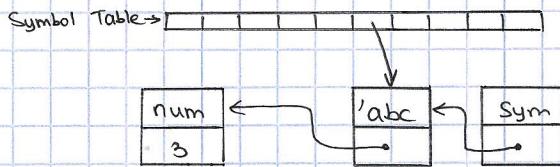


## 3. סמלים (Symbol Table) ו-פונקציית גיבוב

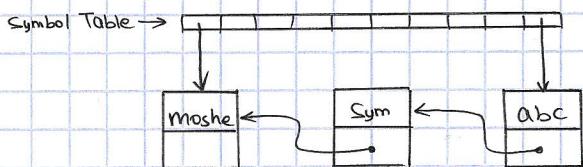
(define abc 3)

'abc  $\Rightarrow$  (const abc)

$\text{abc} \Rightarrow (\text{Var } \text{abc})$

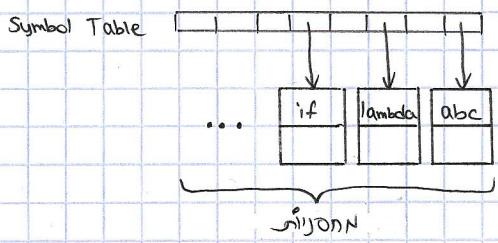


:31c (define abc 'moshe) 31c - מושה נולן



Scanner  $\rightarrow$  ?<sup>311</sup> Symbol  $\rightarrow$  UN (\*)

Symbol Table -> מילון (..., define, if, λ) המכיל מילים, פונקציות ותתי-פונקציות (ב \*)



: Deep Binding → מילוי NC נמוך

## Symbol סמל נושא

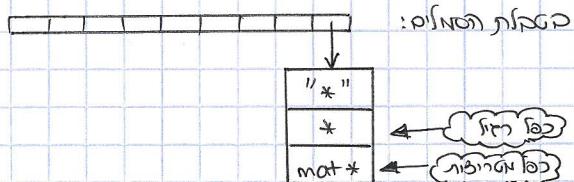
← כב, הברית יג נמלטה וו נטול גוף

המפתחות המפתחים מלחץ? כמו נחנויות חומרי גלם?

(\*) ו) ב) דינמי (dynamic) - Dynamic Scoping ויק מושך וווק הפורמה.

```
(define fact
  (lambda (n)
    (if (zero? n)
        1
        (* n (fact (- n 1)))))))
```

א. שורה וריאנט  
ב. מילוי ניקוד  
ג. פסוקים



(\*) נס בהתהווית הינה נאהו (תמיון מילון), אם בלהו  $\Leftarrow$  דמיון, כמו וטיקן גור.

- Shallow Scoping : Dynamic Scoping - סקופינג עמוק ↗

```
(define map
  (lambda (f s)
    (if (null? s)
        '()
        (cons (f (car s))
              (map f (cdr s)))))))
```

בגנום

כמבל (לפי ציון map) מפה של רשת הרכבת הלאומית ו (33) תחנות תחבורה רפואת הרכבת H.Rail, כונראה כי תחנת הרכבת:

```
> (length  
  (map (lambda (x)  
           (set! s '((a a a a a))  
           'moshe))  
        '(1 2 3 4)))
```

◀ **מיפוי פוליטי** (politic map) – מפה שמשתמשה בצבעים וסימני נקודות כדי לסייע לאדם להבין מטרת המפה.

כואל של גנטומאיה (גנום קולקטיבי). (מג'ור, וטראבאס צוירנו; תרשים Ci הול במאגר ו-SC נקבעו על ידי גנטומאיה)

5- C- 1) የዕለታዊ ሪፖርት በስራውን አገልግሎት የሚያስፈልግ ይችላል

גמיהו גומינית רדוקטיבית ← Dynamic Scoping → גומינית מילויים רדוקטיבית

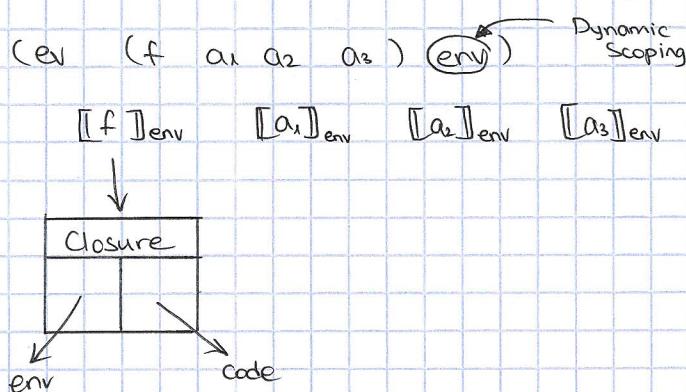
(2) *Ni* *icly* *jeferut* *jaflur* *ciw* *enaburc*.

(java, C#) Dynamic Scoping (\*)

Deep Scoping ① : Dynamic Scoping - [ Python 2 ] ↵

## Shallow Scoping ②

## PPL -> Dynamic Scoping



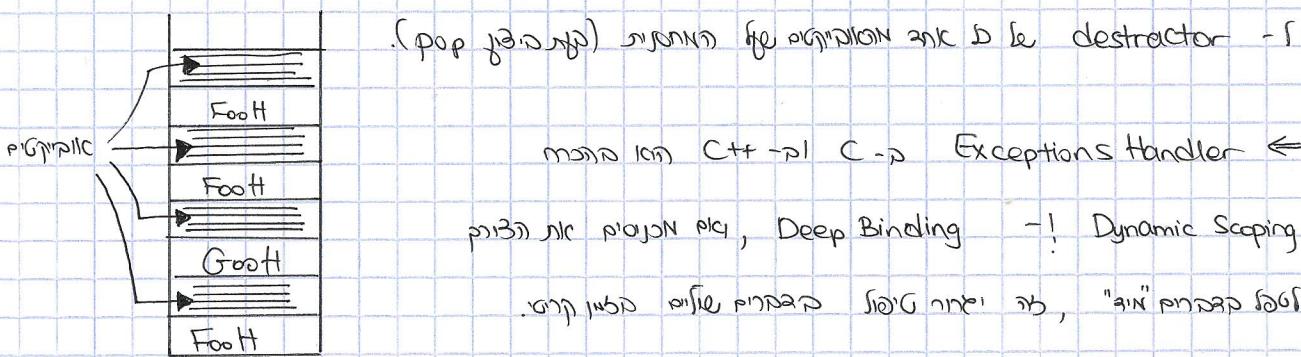
Exceptions Handler  $\Rightarrow$  ? java  $\Rightarrow$  Dynamic Scoping e/p? (\*)

ב)  $\text{func} \text{ pi- } \text{Float} \rightarrow \text{of } \text{pop} \quad \text{היפוך בrc, Goo - } \leftarrow \text{היפוך Foo} \quad \text{רכ: גונדר}$   
 $\text{Goo - } \leftarrow \text{היפוך foo - } \leftarrow \text{היפוך FooH} \quad \text{היפוך FooH - } \leftarrow \text{היפוך GooH}$

עיבוד, Deep Dynamic Scoping עיבוד דינמי עמוק, Java ->

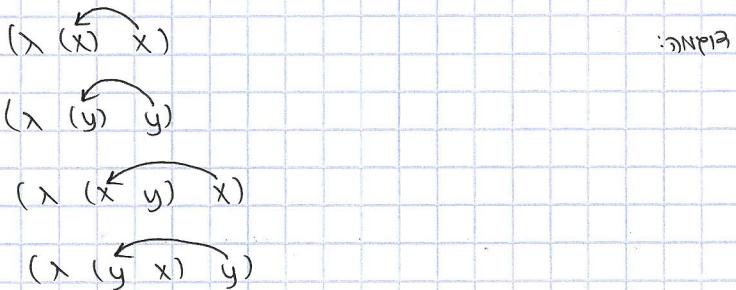
ר' Exceptions Handler for גישת NullifyNilValue בפונקציית

foot  
foot  
foot  
Goutt  
foot



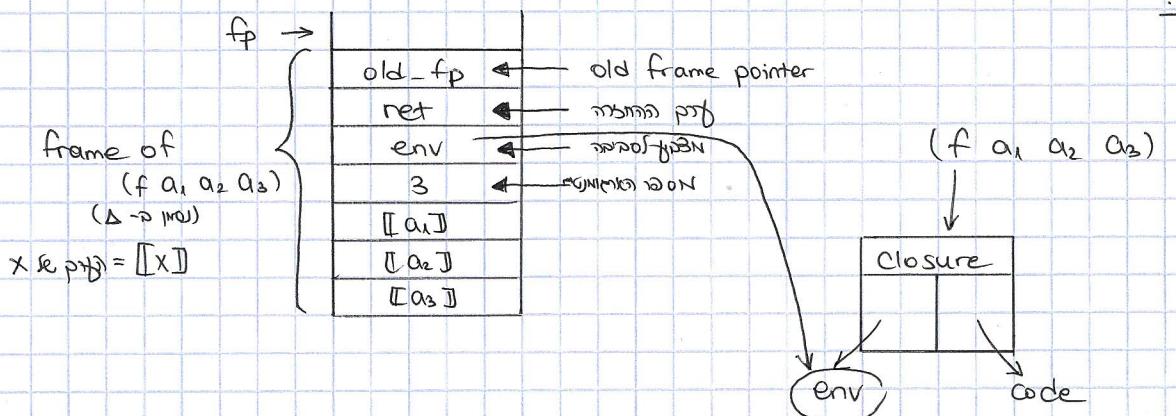
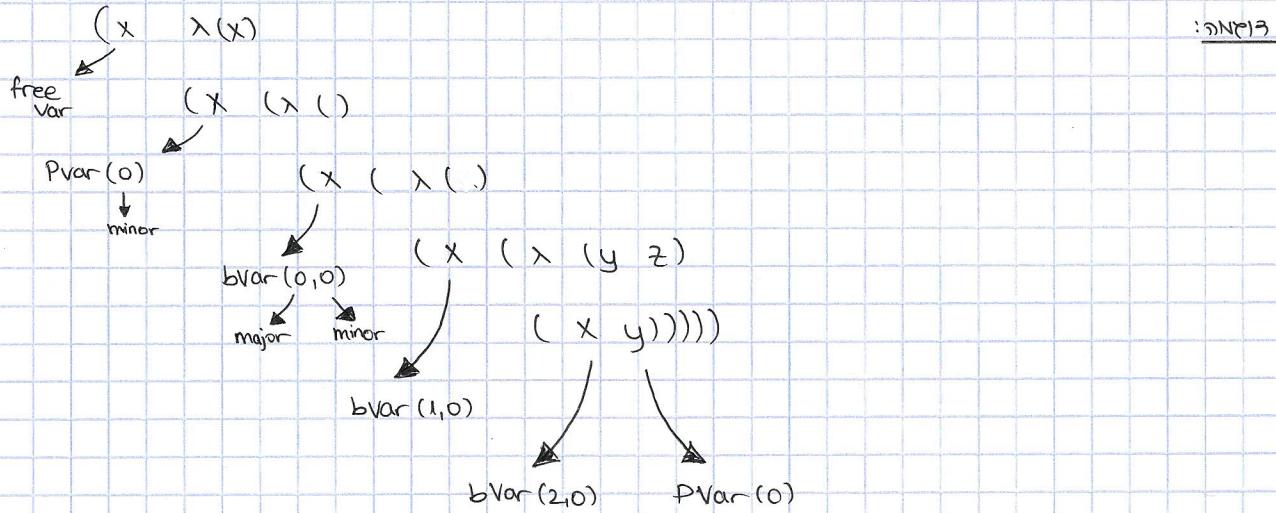
## Lexical Scoping - zusammenfassung

כטבנש נספּה י-א נספּה גָּדוֹלָה נספּה גָּדוֹלָה



לפניהם - free var - FVar • פה 3 יפה

וְנִזְמָנָה תַּחֲנוֹן וְנִזְמָנָה תַּחֲנוֹן , רֵא גַּם הַיְשִׁיר אֶת־הַזְּבָדָה - binding var - bVar  
. וְנִזְמָנָה תַּחֲנוֹן וְנִזְמָנָה תַּחֲנוֹן



(מינר, מילר וברון) : מושב מנהלי נבחר בראובון (תאום).

$$PVar(j) = f_p - \Delta - j$$

$$bVar(i, j) \equiv env[i][j]$$

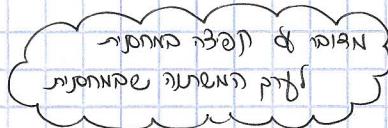
(\*) כנה נס סבב גאנז אונטראיאו ?

האומתים או תחתם נסמכים בClosure (בדרך כלל בתכלית).

לעומת ה-<sup>132</sup> (fact 1000) נספחים 4000 → ~~ב-~~ (נרטווים אוניברסיטאי)

כבר נקבעו היחסים – מאר כפרהוויא או ויטנאמית נסlein (ב-א), וולגוריין או ויטנאמית קראטונג'יאן, גיאו-

כלי (וגם) מיל נס הרכזות הנטקאות



• תקינה ↗ נאבק הוציא, וכך מכך הולך וגדל.

major •

minor •

- 2 මුදල සිංහල ප්‍රාග්ධන නිවැරදි තුළ ප්‍රාග්ධන නිවැරදි - මුදල නෙයා ගියෙන

pVar / fVar / bVar

בנוכחותם מתקיימת הדרישה למסרים מודפסים.

## lexical Addressing ①

የኢትዮጵያ ከተማ የሚከተሉበት ስራዎች

בנין נספח מילון גLOSSARY

מזה 3, מזיה 3 - מיון מילים 4

. (LR(k) ഫോറം, തന്റെ നിർമ്മാണം) . പ്രാഥമിക തഹര ഫോറം - 1 ഫോറം

(ຕອງການ) የሚገኘውን በግብር - ንዑስ ተከታታለ

(תבן ויהי בתק גתון קב- ז' ו כ' ו ז' ו ז' ו ז' )

מגנום גיגס

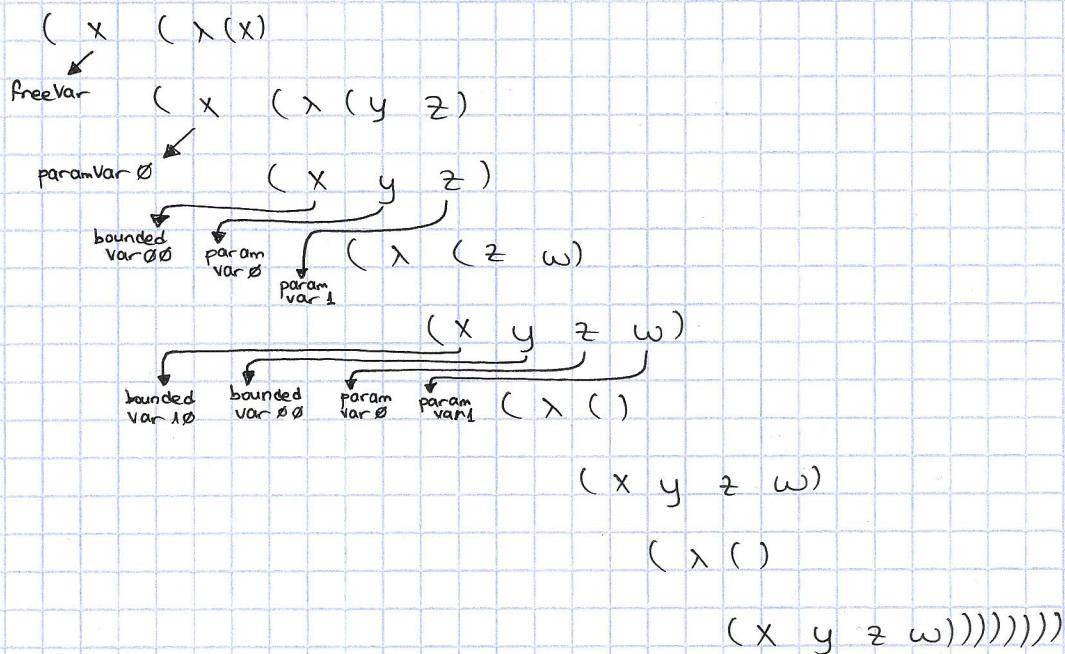
לכון. מטרות ניראיות מילויים.

11/12/2012

## דוקומנטציה - הרצאה 9:

תפקידים פוטנציאליים

לעומת - מוכרים:

פונקציית תבניות פוטנציאלית

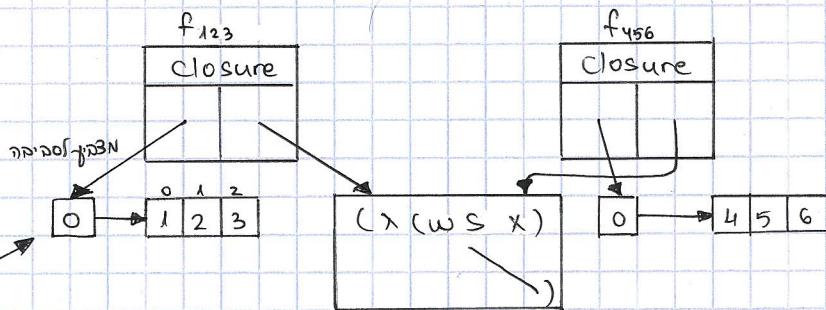
(הו גוף)

```
(define f
  (\lambda (q a z)
    (\lambda (w s x)
      (\lambda (e d c)
        (\lambda (r f v)
          (\lambda (+ g b)
            (\lambda (y h n)
              (\lambda (u j m)
                )))))))))
```

))))))))

```
(define f123 (f 1 2 3))
(define f456 (f 4 5 6))
(define f456789 (f456 7 8 9))
(define f456101112 (f456 10 11 12))
```

(הו גוף f) הגדלת ערך



מי. ודר כריסטוף  
דעת הדריך נזון או  
שלאן סטטוס.

לעומת הדוגמה הקודמת, נסמן את ה-*instance* השני כ-*f123*, כלומר  $f_{123}$ .

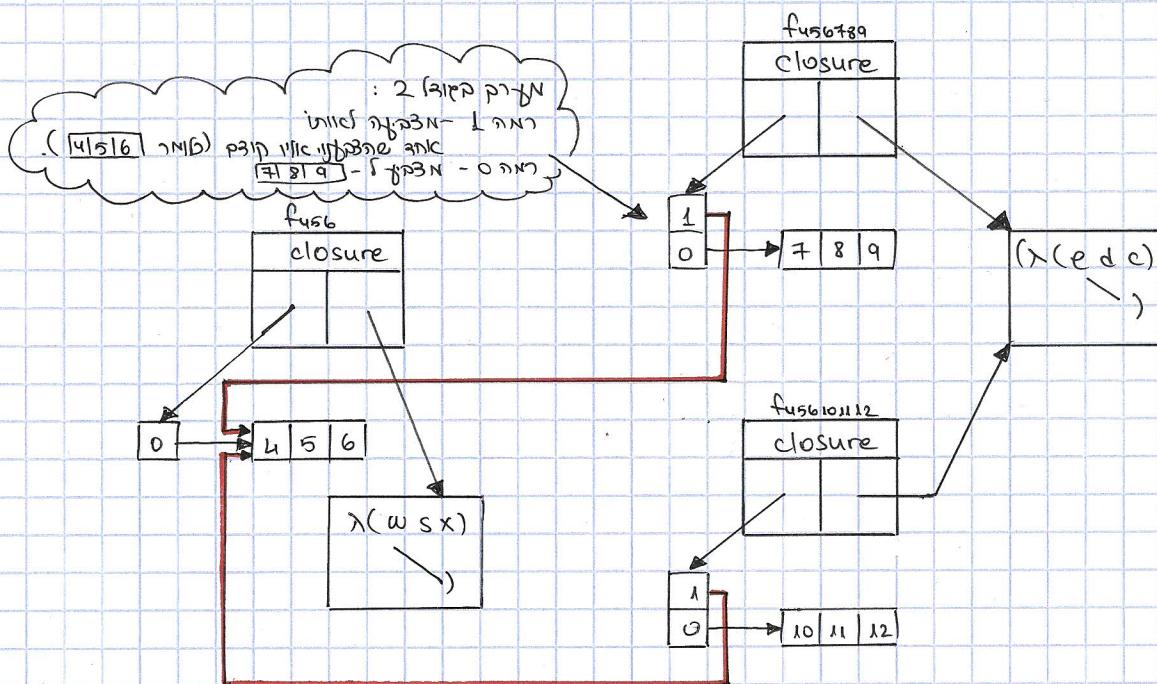
closure → le

constructor -  $\lambda$   $S \rightarrow N \leftarrow$  closure  $\lambda x y p$  (\*)

• *גַּדְעָן* *מִזְבֵּחַ* *לְפָנָיו* ← closure *לְפָנָיו*

$f_{156} = \{ \text{sim1, ref, f123} \rightarrow q_1, a_1, z \}$  se ijie (\*)

(define f456789 (f456 7 8 9))  
(define f456101112 (f456 10 11 12))



:java נספחה ל' זו נ' ←

```
x1 = new C1() {
    int q, a, z;
    void m1() {
        x2 = new C2() {
            int w, s, x;
            void m2() {
                x3 = new C3() {
                    int e, d, c;
                    void m3() { ... } ... }
```

$$\begin{aligned}y_1 &= x_1 \cdot m_1(); \\y_2 &= x_2 \cdot m_1(); \\y_3 &= x_1 \cdot m_1(); \\y_{31} &= y_3 \cdot m_2(); \\y_{32} &= y_3 \cdot m_2();\end{aligned}$$

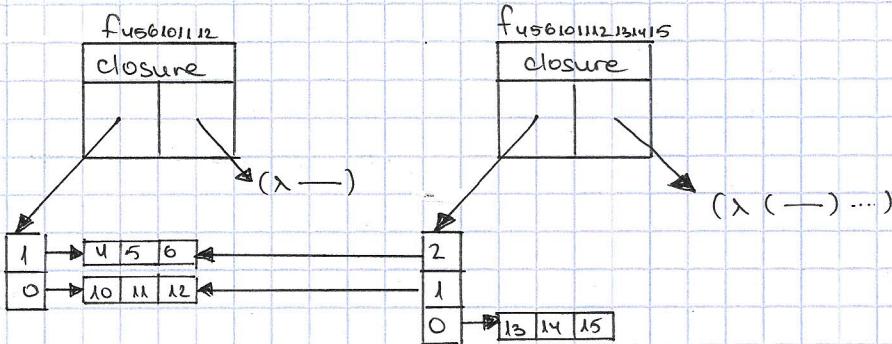
variable instance (y<sub>1</sub>, y<sub>2</sub>, y<sub>3</sub>)

then  $q_1 q_2 \rightarrow$   
 $w, s, x \rightarrow pp \rightarrow$   
 $ed, c \rightarrow plc$  then

↙ מה קיון נס ספה  
נוירוט פאנט

(f<sub>456101112</sub> 13 14 15)

רחלר פונקציה כלהות (בפ' מילויים) נספחים ←



\* נספחים קיימים בפ' מילויים. אם הולך ו增多ת נשים נשים יתנו לנו אוסף של סימני רוחה.

סימני רוחה יי' רכזם ג-ה, ג-ה bounded var → , ג-ה רוחה

הנחתה (ה-ה) ← NO גדרה גדרה שאנטירוק ג-ה צוואר, גדרה גדרה סדרה סדרה.

new C1 () {

הצגה:

```

(***) C2m1 (int a) {
    → x2 = new C2() {
        C2m2 (int b) {
            :
            :
            :
            return x2;
        }
    }
}

```

• כה כתוב ואו יולץ ב' (ה-ה) ? אם כן מילויים ?

• ה-ה של C2 יוכן ב- heap הרהה קא-קע-ס-ען C2 מילויים.

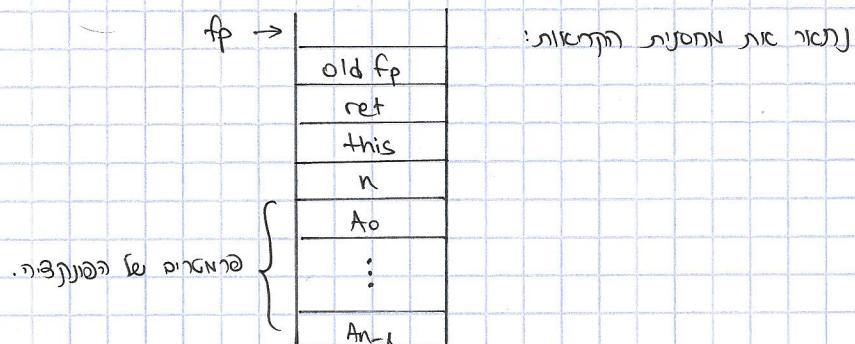
• ה-ה (\*\*) : נזכיר טו, ה-ה של a היה קא-קע-ס-ען (ה-ה) פורה טו;

• heap -> מאריך גודל פורה טו, ה-ה של a פורה טו;

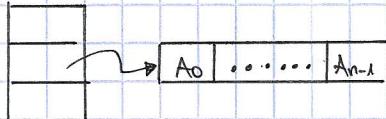
• פוןkt מילויים ס-ען קא-קע-ס-ען ←

• C2m1 (final int a) {...} : מילויים, final -> a -> a

? final -> a -> a מילויים מילויים גודל גודל מילויים מילויים .



לפחות קייפי גדרה (boundary), לרבות נקעה הינה שטח שוכן על גדרה.



כ"כ מפערת סדרת גאנדרהינר - ני נקנין פאר וווען ד-2 (גאנדרהינר?) דיאר, לא אמיט ערנאי!

java p5, ( heap -> ppl stack -> pp k3njev qmpic . e. reis s3b1j p3r0 n3y )

. final will give a generic info

• ביצה כפופה ליריחו - מירוח יבש או יבש ומים

Carry (final int [] a) { ... } : ပြန်လည်

נ' מיליכם נורא  
ל' כוות מיליכם נורא לך ת'

`class`, `int` ו`final` גורמים ארכיטקטוניים (ארכיטקטוניים) נקראים `final`, `int` ו`final` גורם גורם (ארכיטקטוני).

heap-Stack ניגן מחרוזת של גיבובים ייחודיים, scheme-בhid יסודו, גיבובים ניגן מחרוזת של גיבובים ייחודיים.

לעומת Scheme, ב-**C/C++** מושג זה נקרא **pointer** (הנמצא ב-**heap**) ו-**set box** (הנמצא ב-**stack**). (Set -> פונקציית קבוצה שמייצגת ב-**ptr**)

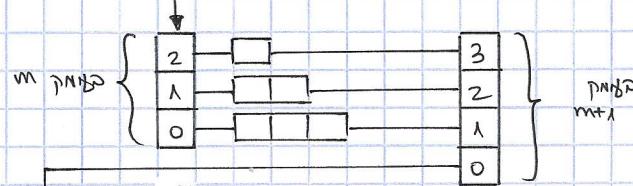
## نحویں و جملوں

env יפ'רכ

## הארון והלטינה

env  
(Expand env)

## פיחקת טק'קה



ויקוּרִות: new • נָרְגִית סַנְוֹקָה.

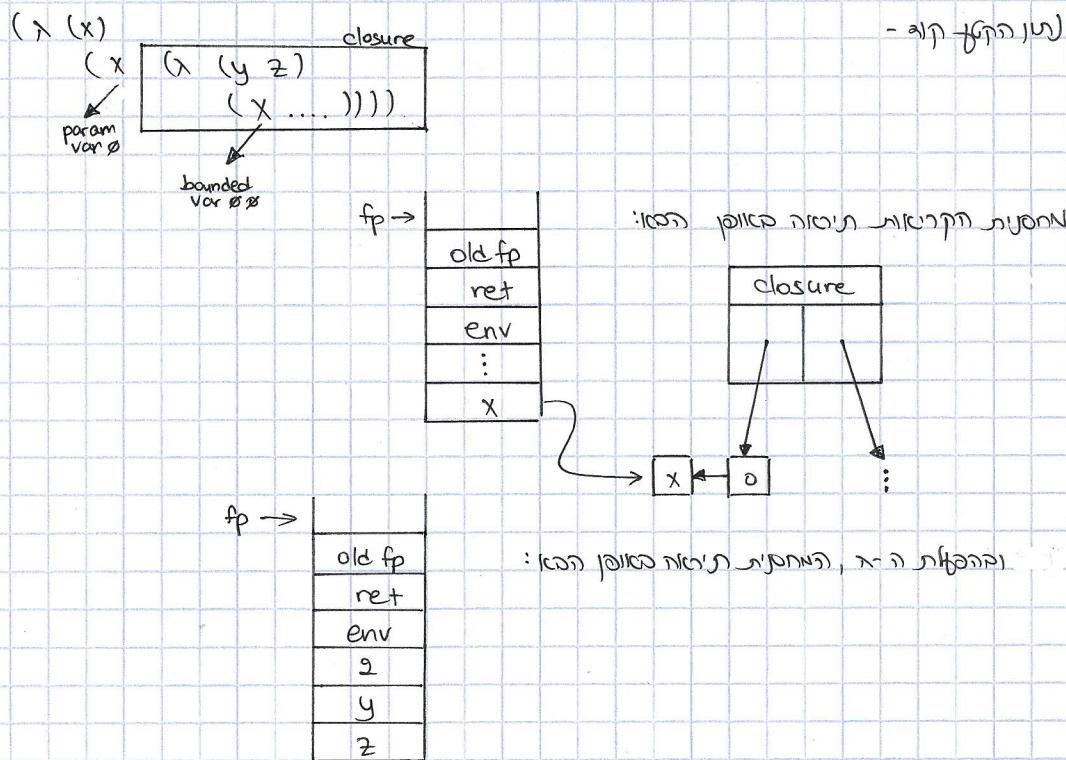
• כרא גרי נורו און אונד נורו, בזיד ג'יגאנט און פְּרִינְגְּרָט.

לפיכך  $\lim_{n \rightarrow \infty} f_n(x) = f(x)$  ב.DOM(f).

מתי התחילו סנקת נ- ppl ↗

למי שפתקה נטה: למי שפתקה נטה

בז'נַעֲמָן



: (ב) הדריך פיראטי ויליאם

```
(define goo
  (λ (n)
    (letrec ((→)
            (→))))
```

```
(define goo  
  (letrec (—)  
    (→ (n))
```

← (ההמם מתקין ויכל לחייב אותו).

:3 ହାତ

(fvar x) ①: הינהו גורם ב-Ν ונקה Variable (בגין ס פונקציית ב-Ν) (\*)  
(pvar x mi) ②  
(bvar x ma mi) ③

(C) (1) የግብር በኋላ እንደሚሸጠው ይመለከት ይችላል

וְכַא כֵּן, שֶׁבָּרְךָ יְהֹוָה אֱלֹהִים עֲלֵיכָם כָּל־בְּנֵי־יִשְׂרָאֵל (\*).

## תורת-הגדרות כנתנתן

return ε' מוגדר בפונקציית הדרישה "זרם" בין-פונקצייתם של פונקציות

int foo (int x) {

בזאת:

⋮  
return (f)(g(h(x)));

*לעומת קבוצת פונקציות*

int foo (int x) {

בזאת:

return (x < y)? (f)(g(h(x))):

*לעומת קבוצת פונקציות* ↗ (g) (h(f(x)));

}

frame → גדרה גנטית של גדרה גנטית דקה - מתחילה ב-

frame ⇒ גדרה פנימית דקה שוקעת ב- כוון (ב-)

$$\text{Ack}(\emptyset, b) = b+1$$

בזאת - פונקציית זרימת:

$$\text{Ack}(a+1, \emptyset) = \text{Ack}(a, 1)$$

$$\text{Ack}(a+1, b+1) = \text{Ack}(a, \text{Ack}(a+1, b))$$

פונקציית גדרה גנטית מוגדרת ב-  $\text{Ack}(a, b)$  כ-  $\text{Ack}(\text{Ack}(a-1, b), 1)$ . גדרה גנטית מוגדרת ב-  $\text{Ack}(a, 1)$  כ-  $\text{Ack}(a, \text{Ack}(a-1, 0))$ .

פונקציית גדרה גנטית מוגדרת ב-  $\text{Ack}(a-1, b)$  כ-  $\text{Ack}(\text{Ack}(a-1, b-1), 1)$ .

int Ack (int a, int b) {

*גדרה גנטית*

if ( $a == \emptyset$ ) return  $b+1$ ;  
else if ( $b == \emptyset$ ) return  $\text{Ack}(a-1, 1);$   
else return  $\text{Ack}(a-1, \text{Ack}(a, b-1));$

}

int Ack (int a, int b) {

*פונקציית גדרה גנטית*

table → L: if ( $a == \emptyset$ ) return  $b+1$ ;

else if ( $b == \emptyset$ ) {  $b=1$ ;  $--a$ ; goto L; }

else {  $b = \text{Ack}(a, b-1)$ ;  $--a$ ; goto L; }

*- frame → מתחילה ב- b ו- a הם אפסים*

*ב- b-1 הערך נקבע ב- 1*

*ב- a-1 הערך נקבע ב- 1*  
ב- 1 מוחלט ב- 1 - ימוך  
לפניהם מוחלט. frame ←  
מתחילה ב- מתחילה ב- 1

! 2-ה שאלת הולכת ו回来了 נס - מ' הולכת ו回来了 נס ←

body N ← body, goto -! label → עתה גורם לכך שbody כביכול יתבצע כbody N (ב)

לעתה body ← פירעון body נס פירעון body ←

שאלה: איך מודים בScheme אם הולכת ו回来了 נס מפערת?

רשותנו לחשוב בפה.

### : Scheme → while מילוי

```
(define while
  (λ (test body)
    (if (test)
        (begin
          (body)
          (while test body)))))
```

לעומת הולכת ו回来了 נס.

(let ((n 5)) ?)

(while

```
(λ() (< ∅ n))
(λ()
  (set! n (- n 1))
  (display "la "))
))
```

### : Scheme → for מילוי

```
(define for
  (λ (from to body)
    (if (< from to)
        (begin
          (body from)
          (for (+ from 1) to body))))
```

(for ∅ 5 ?)

```
(λ(i)
  (display
    (format "Hello ~a ! ~%" i)))
```

שורה גוזרת ← N %

תנאי: כרך N או הנטה נט פירעון ← N a

N נציג N ← N N

> (format "a" "abc") ← באה:

"abc" ← וודאי!

"abc" ← נטען נטען נטען

## קומפלטיה - הרצאה 7:

## אנו לוט

→ 'מְגַנֵּן' (applic — ✓) נושא הפעלה מילכית? run עליה מילכית נושא הפעלה מילכית (applic — )

(define run

(x) (pe tp?)

(cond ((pe-const? pe) pe))

(( pe - var? pe ) pe )

(( pe - if ? pe ))

(with pe

(λ (test dif dif))

(\*) -> IMO

, (run test #f)

1

, (run dit tp?)

, (run dif tp?))

10

(define with

: with નિગ્રિય (\*)

(\lambda (s f))

(apply f s)))

`(not tp?) , tp? , #f , #t : boolean -> boolean -> nBp@ + w' (*)`

110f 002111 7N, (tp? :ic) #t 1087) (\*) PIPNP PIC ↪

? >(+ 1 (if (+ 2 3) (+ 4 5) (+ 6 7)))

ויליאם ג'טראן ↗  
גראניטס

. 5 ගුණී පියා, සැලුගු frame නි නිස උච්චා තියි (+ 2 3) ඇතුළු, if -o, p/1

נִגְמָנָה (not tp?) עַדְעָן (\*) מִתְּנִינָה מִכְּנָה ↗

? > (+ 1 (+ 2 (if (+ 3 4) (+ 5 6) (+ 7 8)))))

The diagram consists of three separate arrows pointing upwards from the left side of the page. Each arrow has a handwritten label below it:

- The first arrow is labeled "ללא מיניהם" (labeled with a double dagger symbol).
- The second arrow is labeled "ללא מיניהם" (labeled with a double dagger symbol).
- The third arrow is labeled "ללא מיניהם" (labeled with a double dagger symbol).

A dashed line extends from the right side of the page, sloping downwards towards the center. The text "not of the kind" is written above this dashed line.

(+ 2 ....) என்று frame-இ NIC தாங் போல, (+ 3 4) கூடியது, சம்பந்தமாக

13. א. ב. ג. ד. ה. ו. ז. ח.

$> (+1 (+2 (\text{if } 'moshe (+5 6) (+7 8)))))$ ,  $\text{bc}$ , ( $\text{not tp?}$ )  $\text{aren}$ )  $(*) \text{ NPNP PIC} \Leftarrow$

↑  
#t      ↑  
#f      ↑  
p1, p2, p3, p4  
frame 012 123

? >(+ 1 (+ 2 (if 3 (+ 4 5) 6))) ➔ 6

א  
גראניט פולני  
ס.ק.

(+ 2 —) അതു frame -> നില പജ് യെല്ലാർ ദൈവകൾ എന്നും (+ 4 5) കൂടാൻ

.10 : ପରାମାନ୍ତର, pfi

$>(+1(+2(\text{if } 3 (+45) [6])))$  נגזרת נסיעה,  $\text{tp}^2$  ו' (\*\*\*) פ'IC

• תרגום

מקרה 1-2: גורם גנטיאלי (ג'ן קלאבטי) אם  $\text{פ}(\text{הטיה}) \leq \text{פ}(\text{הטיה})$  ב- $\text{פ}(\text{הטיה})$ .

(if ( ... )

∴ 150,000

כאמ'?

: Seq.

$$[\![(\text{seq } (e_1 \dots e_n))]\!]^{+P?} = (\text{seq}([\![e_1]\!]^{\#f} \dots [\![e_n]\!]^{\#+}))$$

רשות חוקינה נקבעה בזאת:

କ୍ରମିକ ନାମଙ୍କଳି ହେଉଥିଲା ୫ ଦିନୀରେ ପରିବାର (ପିତା, ମାତା) (+ ୧ ...) ରେ ଅନ୍ତର୍ଭାବରେ frame ରେ , (୧ ୨ ୩) ଯେବେଳେ ଏହା

• ג' תרגומן ימ' מילויים ביחס לחיות גוף # פס וט גאות

ps1 ←

$$[[\text{seq } (e_1 \dots e_n)]]^{tp?} = (\text{seq } ([e_1]^{tp?} \dots [e_n]^{tp?}))$$

: or נתקדש

$$[\text{Or } (e_1 \dots e_n)]^{+p?} = \text{Or } ([e_1]^{+p?} \dots [e_n]^{+p?})$$

(מגניטית)  $\text{Co}$  ו- $\text{Fe}$  נאומני ו- $\text{O}_2$  מילוקו מ- $\text{Fe}_2\text{O}_3$ .

כיצד ניתן להזכיר מושגים?

- if we find the same word in the sentence, it is (or -) the same word in the sentence (\*).

> (or (car '(#f #t))  
      2  
      3)  
: #t

(or -o תְּפִלָּה מִתְּבָרֵךְ אֱלֹהִים בְּשָׁמָן) וְיַחֲנֵן 2, מילון - מילון

לידן הנקה נס

$$[\text{I} (\text{or} (e_1 \dots e_n))]^{tp?} = (\text{or} ([e_1]^{tp?} \dots [e_n]^{tp?}))$$

הנתקן נושא הטענה ? tp? offence העשוי בזאת ?

• תְּמִימָה נִכְנָתָה גַּם־בְּרֵךְ, מִגְּבָרָה

: NUS NIPERATION pic, or -n 50% RNA Seq -n 50% : RNA (A)

(or ( $[e_1]^{#f} \dots [e_n]^{p?}$ ))  $\rightsquigarrow$  (seq ( $[e_1]^{#f} \dots [e_n]^{p?}$ ))

(and (f<sub>1</sub> x)  
(f<sub>2</sub> x)  
(f<sub>3</sub> x)  
(f<sub>4</sub> x))

בז'אנר גלוי סימול: מושג זה מתייחס ל-?

? או -\\$ inc if -\\$ macro-expansion מושגים אלו?

: define ↗ ↘

$$[\llbracket (\text{define } (\text{var } x) \text{ PE}) \rrbracket]^{tp?} = (\text{define } (\text{var } x) [\llbracket \text{PE} \rrbracket]^{\#^f})$$

: apply ↗ ↘

$$[\llbracket (\text{applyc proc (e<sub>1</sub> ... e<sub>n</sub>)}) \rrbracket]^{\#^f} = (\text{applyc } [\llbracket \text{proc} \rrbracket]^{\#^f} ([\llbracket e_1 \rrbracket]^{\#^f} \dots [\llbracket e_n \rrbracket]^{\#^f}))$$

$$[\llbracket (\text{applyc proc (e<sub>1</sub> ... e<sub>n</sub>)}) \rrbracket]^{\#t} = (\text{tc-applyc } [\llbracket \text{proc} \rrbracket]^{\#^f} ([\llbracket e_1 \rrbracket]^{\#^f} \dots [\llbracket e_n \rrbracket]^{\#^f}))$$

גאומטריה: מושג זה מתייחס ל-  
 $\uparrow$  proc  
 $\cdot > ([\llbracket (\text{car } '(\text{, car }, \text{cdr})) \rrbracket]'(a b c))$

proc, #t -> proc f<sub>f</sub> מתייחס ל-  
 $\cdot$  (defn proc name (body))  $\cdot$  proc proc, a מתייחס ל-  
 $\cdot$  (defn proc name (body))

. #<procedure car> מתייחס ל-  
 $\cdot$  (lambda (x) x) מתייחס ל-  
 $\cdot$  (lambda (x) x) מתייחס ל-  
 $\cdot$  (lambda (x) x) מתייחס ל-

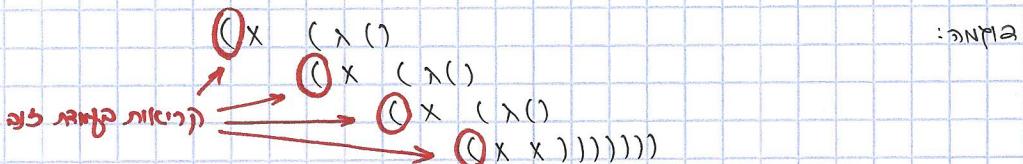
: lambda ↗ ↘

. λ-Simple → λ-Optical → λ-Variadic → λ-Opt

$$[\llbracket (\lambda\text{-simple } (abc) e) \rrbracket]^{tp?} = (\lambda\text{-simple } (abc) [\llbracket e \rrbracket]^{\#t})$$

הינתן: פונקציית return מושג, מושג frame מושג, מושג λ-Optical מושג.

{ ה-λ-optical פונקציית return }



? מושג? C, גאנטיאט כירע C, return x (→, ומיוני ההפניות הילוקטור, דילוקטור return)

דילוקטור return, ε

frame → λ-optical, call frame → λ-optical return ↗ ↘

בז'אנר גלוי סימול: מושג זה מתייחס ל-  
macro-expansion ↗ ↘



גוף פונקצייתית  
 call by name

```

real foo ("cbn real a, cbn int i) {
    real S = 0.0;
    for (i=0; i<MAX; i++) {
        S += a;
    }
    return S/MAX;
}
  
```

תוצאות הקריאה: סכום כל איבר

סכום כל איבר = סכום כל איבר + איבר הנוכחי

סכום כל איבר נאנו מודפס ב-call by name

סכום כל איבר נאנו מודפס ב-call by need

סכום כל איבר נאנו מודפס ב-memoization

סכום כל איבר נאנו מודפס ב-call by value

סכום כל איבר נאנו מודפס ב-call by reference

הסדרת ה执着: Order-evaluation מוגדרת כ-normal order evaluation (\*)

הסדרת ה执着 מוגדרת כ-call by name (ה执着 מוגדרת כ-call by value)

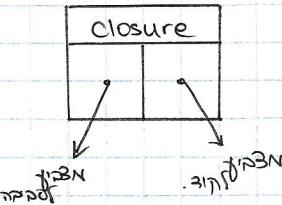
הסדרת ה执着 מוגדרת כ-call by need. (ה执着 מוגדרת כ-call by reference)

הסדרת ה执着 מוגדרת כ-memoization, normal order evaluation -> סדרת ה执着

call by name -> סדרת ה执着, normal order evaluation סדרת ה执着 מוגדרת כ-call by name (\*)

סדרת ה执着 מוגדרת כ-call by value

## לומדים - הרצאה 8:

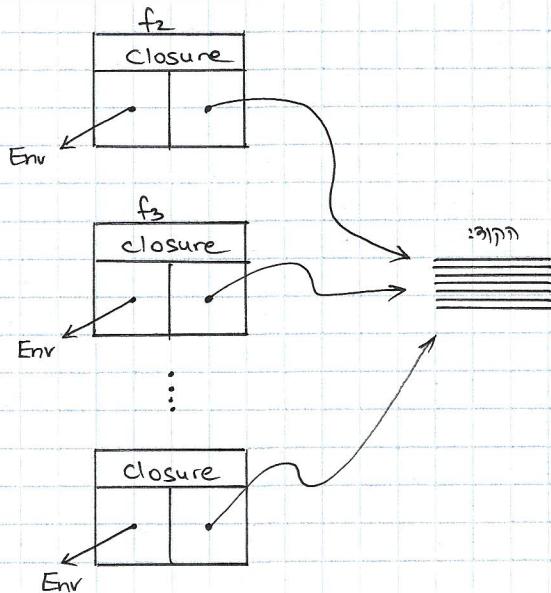
Closures vs. Methods

נקודות על closure:

$$f \equiv (\lambda(a) (\lambda(b)))$$

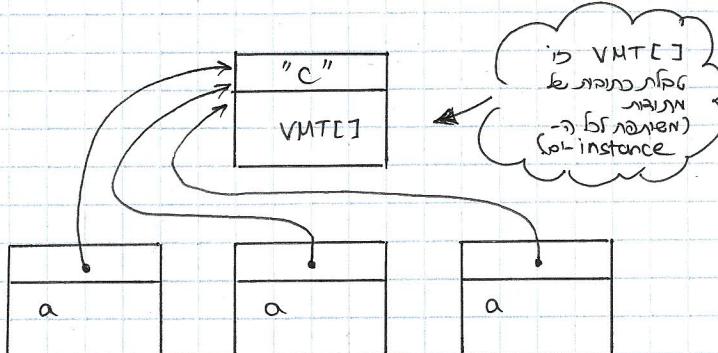
```
(define f2 (f 2))
(define f3 (f 3))
⋮
```

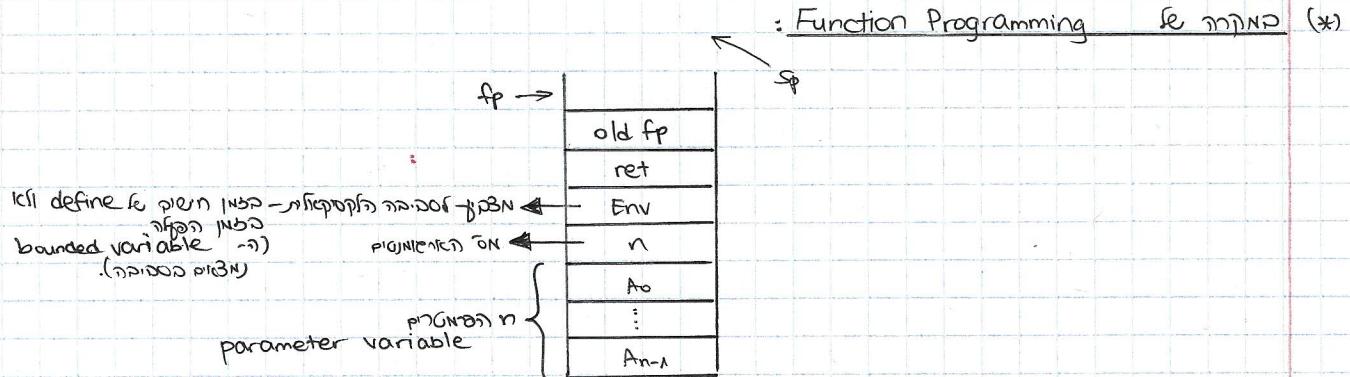
בכל closure יש לנו גורם עולמי  
הנמצא בClosure עצמו  
ולא בclosure של פונקצייתו  
או נספחים לו.



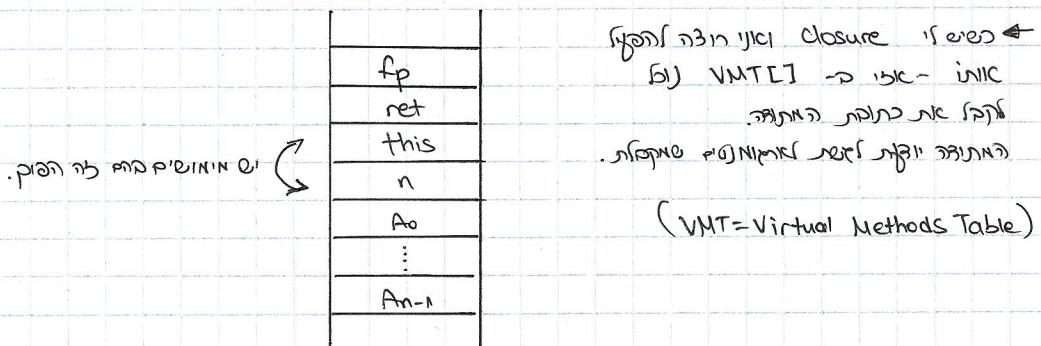
אנו יכולים לחשוב על closure כמו על גורם עולמי שקיים בכל מקום.

closure → הינה מושג ① ו-closure → הינה מושג (\*)  
או מושג ②

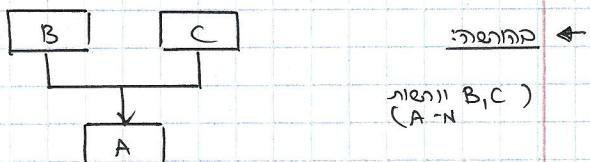




: (OOP ) Object Oriented Programming Se import (xt)



When we pass this list,  $f(x, a_1, a_2, a_3)$  will  $x$ .  $f(a_1, a_2, a_3)$  be returned (\*)



תקבץ גנרטור מילולאי יי' פונטיות תומאסון מוגדר כת-טיפוס כת-פונטיות, שאותו מוגדרת כ-Sub-class של קבוצת כת-פונטיות.

( foo  $\gg$  (0N) & offset  $\rightarrow$  ) = Foo )

Variable X → register → VMT[FOO] : pc ←

מינימליסטי - מילויים מינימליים. foo שמיינטן מינימליים. foo שמיינטן מינימליים.

.VMT[] → p3 foo



```
foo (Rect r) {  
    r.draw();
```

ପାଇଁ ନାହିଁ ।

on window)  
(Rect ->

3

ב-**virtual** מודול, ניתן לשלוח ביצועים מודולריים, ו-

• Obj pointer पर जिनसे प्रैक्टिक बना, r. class. VMT[DRAW](r) ~ 5

הארה כו, ב"ג עז.

(virtual inherited (mp))

interface      SOB {  
                  Scheme Object  
                  ||

:map interface -> port

SOB apply (final SOB[] v);

3

Obj1 = new SOB();

CN1 d:

SOB apply (final SOB[]) v<sub>1</sub>) {

```
int a,b,c;
```

```
obj2 = new SOB() {
```

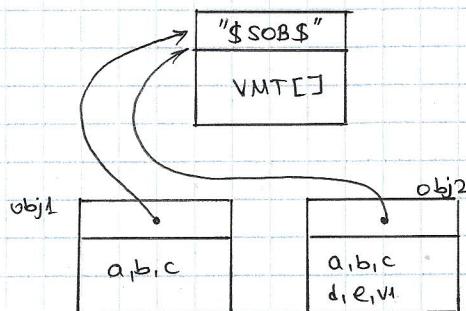
```
int d,e;
```

SOB apply (final SOB[J] v2) {

100

3

3



a,b,c →  
.PƏNEN

o

$a, b, c$  ? obj 1 be state  $\rightarrow$   $\mathcal{N}^0$ .

3 obj 2 (6, state  $\rightarrow$  2N •

(\*) סדרת מילים יפה (obj2 we m3bi'ne) (obj3 יפה גוף גוף) מילויים

final int[] obj = new int[5]; // heap -> obj

$\forall_1 \exists_1 \forall_1 \exists_1$  final ci ה-psi וילך נארולין.

## Code Generation

.pipeline → מנגנון ייצור קוד Code Generation

סימנטית סינטקטית. מילוי מילים נאות מילים + מילוי מילים נאות מילים

• From a parse tree / AST → to code generation (\*)

← פונקציית `next`, מפליצה ליה `next` קידומת מוגבלת נספחת בפונקציית `parse`.

לזה מוכנה קבוצה : Cisco מילויים

הנתק חאייה, חיטור, כמ' וכמו פס' מ-הנתקים (ב(זב) - 86, נתקן קטע כמ' יב, הילע קלאינר נסמן).

. (תרכז גיאוגרפיה 5)

## E of Code Generation (ஈவு, நில)

אנו מודים לך על תרומותך ותומך בCode Generation וCode Refactoring!

\* נייחות ריגור אובייקטיב נס סטנדרטי (set) \*  
\* היגיינה גנטית (genetic hygiene) - מושג Black box \*

י. הַכְּרִי

?Seq סדרה של איברים : Sequence

$$[[\text{seq } (E_1 \dots E_n)]] = [[E_1]]$$

$[E_2]$

卷之三

[E<sub>n</sub>]

. Side effect מושפע מהתוצאות של פעולה אחרת, side effects

לעתים קוצרה בשם side-effects

הפעלה מושפעת מהתוצאות של פעולה אחרת, side effects

(begin 1 2 3)

בdziig:

2. מושפע מ"הפעלה", 1. מושפע מ"הפעלה"  
כיצד? (display →) "הפעלה"  
3. מושפע מ"הפעלה", side effect  
כ"ה" מושפע מ"הפעלה".

(begin 1

2

(display "moshe!")

3)

בdziig:

(\*) אם מושפע מ"הפעלה" פירושו כי הפעלה מושפעת מהתוצאות של פעולה אחרת?

נכונות כורט פונקטן - R0

(begin (display "moshe!")  
3)

(begin 1  
2  
(display "moshe!")  
3)

נכונות תקינה,

(display →) R0 ← void : קיון נוירא, תוצאות סיגטן מושפעות.  
R0 ← 3

הפעלה מושפעת מהתוצאות של פעולה אחרת, side effect מושפע מהתוצאות של פעולה אחרת.

הצהרה זו לא נכונה כי R0 מושפע מהתוצאות של פעולה אחרת.

ולכן, side-effect מושפע מהתוצאות של פעולה אחרת.

(\*) גלויה - פונקציית CIC בפונקציית "mul". בdziig:

גלויה, ראה כי לint יש יוניק "הפעלה", אך לא כערך.

הנימוק כפוף לנקודות כתובות על ידי הדריך, גלויה, גלויה.

(לעתים edx -> km Side-effect =>, כלומר ערך edx מושפע מהתוצאות של edx -> km)

כותרת גלויה ? OR גלויה ? :

$\llbracket (\text{or } (E_1 \dots E_n)) \rrbracket = \llbracket E_1 \rrbracket$

CMP R0, SOB\_FALSE }

JUMP\_NE L\_EXIT\_ }

$\llbracket E_2 \rrbracket$

CMP R0, SOB\_FALSE }

JUMP\_NE L\_EXIT\_ }

:

$\llbracket E_n \rrbracket$

L\_EXIT\_ :

הפעלה מושפעת מהתוצאות של פעולה אחרת, side effect מושפע מהתוצאות של פעולה אחרת.

(\*) הינה: מלה איתן ולו, הינה פונקציית שמיון אם יתגלה if, או אם יתגלה else.

אלא, ברגע שיתגלה if.

שינה: (בז' ) - if נטה, בז' one\_label\_label - if יתגלה פונקציית else.

( L\_EXIT\_12 : גזירה )

ונז' נטה פונקציית else. מכאן, בז' one\_label\_label - if יתגלה פונקציית else. ←

? if "הנה קה ר' ז" : IF

$\llbracket (\text{if test dit dif}) \rrbracket = \llbracket \text{test} \rrbracket$

CMP R0, S0B\_FALSE

JUMP\_EQ L\_ELSE\_39

$\llbracket \text{dit} \rrbracket$

JUMP L\_IF\_EXIT\_39

L\_ELSE\_39 :  $\llbracket \text{dif} \rrbracket$

L\_IF\_EXIT\_39 :

(var+ const ) 00000000 - כה היכתה מפונקציית else - כה היכתה מפונקציית if. ←

#1 הרצאה 3, פסק

: CPS of 1 הרצאה 1

```
(define search-in-rib
  (λ (a s ret-min ret-nf)
    (cond ((null? s) (ret-nf))
          ((eq? (car s) a) (ret-min 0))
          (else (search-in-rib a (cdr s)
                                (λ (min) (ret-min (+ 1 min)))
                                ret-nf)))))
```

חישוב נושאן הפלט  
ב-הארת הפלט  
הפלט מוחזק

```
(define search-in-ribs
  (λ (a env ret-maj+min ret-nf)
    (if (null? env)
        (ret-nf)
        (search-in-rib a (car env)
                      (λ (min) (ret-maj+min 0 min)))
                      (λ () (search-in-ribs a (cdr env)
                                             (λ (maj min)
                                               (ret-maj+min (+ 1 maj) min)))
                                             ret-nf))))))
```

חישוב נושאן להרין  
הפלט

search-in-ribs → search-in-rib → וירטואלי ←

```
(define pe→lex-pe
  (letrec ((run
            (λ (pe)
              (run pe '() '())))))
```

גדרת הפלט  
נקייה או run  
run מוגדר run  
PE = parsed expression  
params = params  
env = env

## : run fls פלט בירן

(run (λ (pe params env)  
 (cond  
 :

Cloud 1:  $\lambda$  ו-pe נאטור  
 $\rightarrow (\text{eq?} (\text{car pe}) 'var)$   
 (with pe  
 $\lambda \_ v$ )

Cloud 2: גלגול אוניברסלי נאטור (כלומר נאטור)  
 $\rightarrow (\text{search-in-rib} v \text{ params})$   
 Cloud 3:  $\lambda$  נאטור, נאטור ID פק  
 $\rightarrow (\lambda (\min) '(\text{pvar} , v , \min))$   
 $\rightarrow (\lambda ()$

(search-in-rib v env  
 $\lambda (\text{maj min})$   
 $'(\text{bvar} , v , \text{maj} , \text{min})$   
 $\lambda ()$   
 $'(\text{fvar} , v))))))))$

Cloud 4:  $\lambda$ -Simple  
 (באותה מילויים  
 פולימורפיים או הולמים).  
 $\rightarrow (\text{eq?} (\text{car pe}) '\lambda\text{-simple})$   
 (with pe  
 $\lambda (\_ \text{argl body})$   
 $'(\lambda\text{-Simple} , \text{argl} , (\text{run body argl}))))$

Cloud 5:  $\lambda$ -opt  
 (באותה מילויים  
 מושגים או הולמים).  
 $\rightarrow (\text{eq?} (\text{car pe}) '\lambda\text{-opt})$   
 (with pe  
 $\lambda (\_ \text{argl opt body})$   
 $(\text{let} ((\text{argl} '(\_ @argl , \text{opt})))$   
 $'(\lambda\text{-opt} , \text{argl} , (\text{run body argl}))))$

:

- הוראות הנטה - (cons params env) (\*)

## : annotate-tc (בננות פלט, הוראות הנטה)

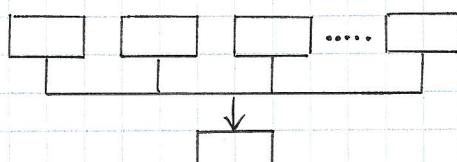
יש לנו פלט מילויים מילויים run (בלי), run pe plc, annotate-tc  
 פלט מילויים מילויים run (בלי), run pe plc, annotate-tc

הנארה ציפורן.

פלט מילויים מילויים run (בלי), run pe plc, annotate-tc  
 פלט מילויים מילויים run (בלי), run pe plc, annotate-tc

## Design Patterns ←

. מילויים מילויים run (בלי), run pe plc, annotate-tc  
 פלט מילויים מילויים run (בלי), run pe plc, annotate-tc



זה הוראות הנטה נעלמת

ונעלמת גזירה נעלמת

. מילויים מילויים run (בלי), run pe plc, annotate-tc  
 פלט מילויים מילויים run (בלי), run pe plc, annotate-tc

## Code Generation

### pvar מנגנון

המערכת בפונקציית הפעלה, סדרת הפעולות בפונקציה. רצף של הפעולות בפונקציה נקרא code generation ופונקציית הפעלה מוגדרת כפונקציה.

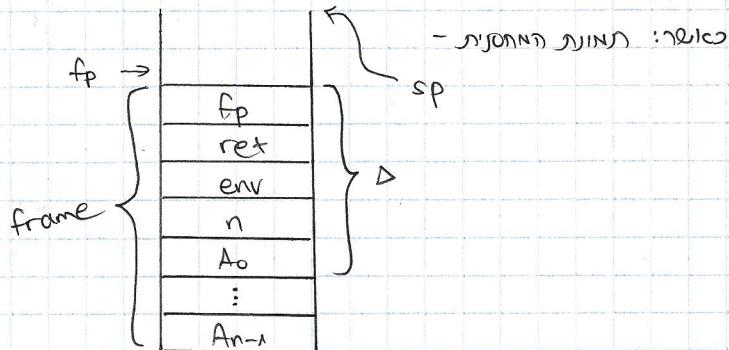
$R\emptyset \rightarrow \text{ויה} \ L(pvar \vee min)$

$L(pvar \vee min) =$

$R\emptyset \leftarrow \text{FRARG } (\Delta + min)$

{frame pointer argument}

$R\emptyset \leftarrow \text{FRARG } (\Delta + min)$



המזהה פוןksi, גיבובוןיג, נתקדש פונksi ופונksi רג'  $\Delta, min$  מזהה (\*)

הזהה פוןksi מיל' נתקדש הצעה הזהה פוןksi מיל'

### bvar מנגנון

$L(bvar \vee maj \ min) =$

$R\emptyset = \text{FPARG } (3)$

{ג'ר, ג'ר}

$R\emptyset \leftarrow R\emptyset [maj]$

INDD ( $R\emptyset, maj$ )

$R\emptyset \leftarrow R\emptyset [min]$

Env [maj] [min] מיל' נתקדש  $R\emptyset$ , הצעה הזהה פוןksi מיל'

$$\text{INDD}(p, n) \equiv * (p + n) = p[n]$$

הזהה פוןksi מיל' (\*)

## எல்லாம் கீர் applic

$$[(\text{applic } \text{ proc } (B_1 \dots B_m))] = [[B_m]]$$

push (R $\emptyset$ )

[B<sub>m-1</sub>]

push (R $\emptyset$ )

[B.]

push ( $R\emptyset$ )

push(m)

[proc]

CMP (SOB-TYPE)

JUMP\_NE L\_error\_not\_a\_clos

PUSH INDD(R0,1)

Call (INDD(RØ,2))

$$Sp = 2 + \left( \frac{m - \pi}{m + \pi} \right) \text{exp}(-\pi)$$

## הניער החקלאי (בגיה)

A diagram illustrating a stack frame structure. On the left, the label "fp →" is written above a vertical line. To the right of this line is a bracketed list of items, each preceded by a short horizontal line:

- fp
- ret
- env
- n
- Ao
- :
- An-1

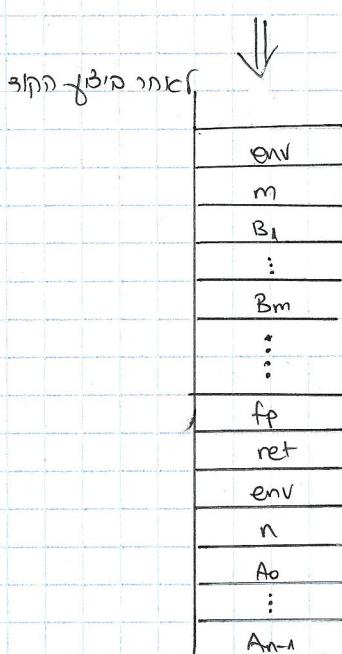
The bracket on the right side of the list is labeled "frame (\*)".

(+/-) motion frame if 0.1, the next frame is 0.0

$\pi_1(B \rightarrow \gamma_B)$  Poly frame slips?

• గుణాల ప్రాంగణాలకి ఆచ పెన్స్ విభజన ది రేస (\*)

( ජ්‍යෝගාරිත්මක සඟන් Dr. Racket)



לען, נסיבותן. הרכה יוג'ה וטף, גנום כל אחד (Nelson)

(\*) קבוצה של יישובים Scheme-object

וְאַתָּה תִּשְׁלַח בְּנֵי יִשְׂרָאֵל מִן-פָּנֶיךָ וְיַעֲמֹד בְּנֵי יִשְׂרָאֵל בְּבָרָאֵל כִּי-כֵן

IND (R $\emptyset$ )  $\vdash$  *unic after m, minic*

በዚህ የትምህር ንግድ ገዢ ተስፋ ተስፋ ተስፋ ተስፋ ተስፋ ተስፋ ተስፋ

(\*) זכיינן מ-2 מילויים frame → זכיינן מ-2 מילויים

① (proc -> body ->) הינה פונקציית Proc

$\lambda$ -variadic +  $\lambda$ -opt  $\rightarrow$  ②

• 8.000 ፩ በመጀመሪያ የሚከተሉት ማዕከራዊ  $\lambda$ -variadic

ret  
env  
1000  
1  
:  
1000

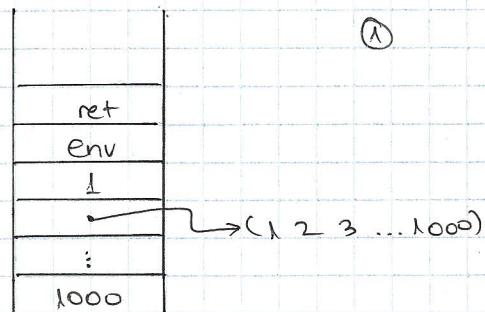
82. גודל גזען מיקון גנומית (בז'ר גאנטהו.)

? הלו pointer-ה מילא לפני הולך - כמה גורמים

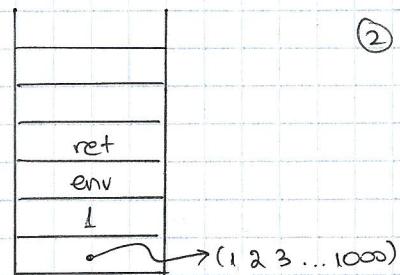
באנדרה פוט-ו-גראן נטולות על אקלים הנבדוקות

"ביק" כוונת המילה בילוי, או שטחן גוף הבהיר בילוי

נומינטור נטורה אוניברסיטה

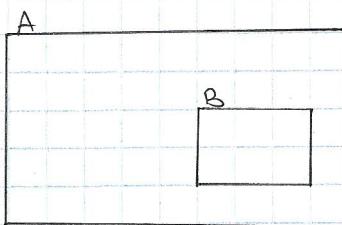


הנורווגיה לה - בירק ג'ונסן מילר (טומאס ג'ונסן) אונז'ה.



(\*) Configuration 3 ו' DS, ביד יכין מפלס נו' תרמוניות דתנו'ן

← ביציך גזרת צורה נסגרה (וכן נאסרן) אם ה- frame-ם מושך ווילג נסגר.



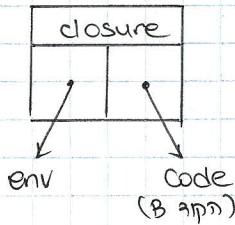
## $\lambda$ -Simple

Closure  $\rightarrow$  self-referential - A

. closure  $\rightarrow$  ફે બોડી  $\rightarrow$  ફે એજ નિચન -B

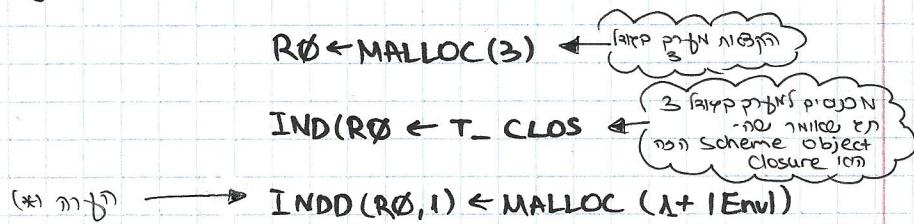
in A - closure pipe of D ←

In B - closure presence of C



: closure  $\lambda x_1 \dots x_n . A$  (\*)

$[(\lambda\text{-simple argl body})] =$



{  
for ( $i=0, j=1 ; i < 1\text{Env} ; ++i, ++j$ ) {  
     $R0[i][j] = \text{Param};$   
}

$R0[1][0] = \text{MALLOC } (\text{iparams})$

{  
for ( → ) {  
    —  
}

$R0[2] = \&L\_CLOS\_CODE\_139$

JUMP L\_CLOS\_EXIT\_139

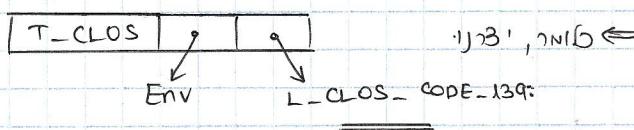
L\_CLOS\_CODE\_139 :

PUSH fp	B
$fp \leftarrow sp$	
[body]	
pop fp	
ret	

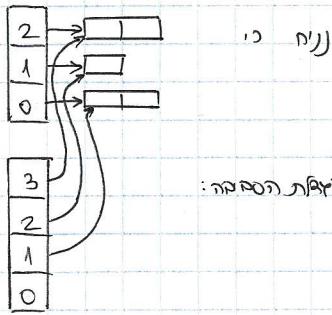
(\*\* נספ) →

L\_CLOS\_EXIT\_139 :

CLOSURE  
ptr R0 ->  
num closure



הנתקה נ-1 וטבילה טבילה (\*) : ↗



בנוסף לארון התנ"ה:

!הענין נגזר מהתפקיד כי מה שפועם פועל יכלה לאפשרו רשות. (כיוון שטהור עוזר, גלויה)

עוזי הפטיניה).

body-הו Le code generation-הו גדרה מילויים (filling): נציג (\*) ↪

בכל פונקציית body נקבעו מנגנון אבטחה:

כינ' גראן נספַתְהָ בְּרִיאָהָ וְעִזָּהָ בְּרִיאָהָ.

כונת:

```
(define fact
  (\lambda(n)
    (if (zero? n)
        1
        (* n (fact (- n 1)))))))
```

RF  $\Rightarrow$  implicit ( $\lambda(n) \rightarrow$ ) closure  $\rightarrow$  nice thing for A, fns define nice problem.

לענין גזירות ב- B (fact 5) ניתן לרשום

(we find A) we, in B is (fact 10) also the answer

(define f  
  (λ(a)  
    (λ(b)  
      (+ a b))))

(define f<sub>1</sub> (f 1))      (define f<sub>2</sub> (f 2))      (define f<sub>3</sub> (f 3))

$$(f_3 \quad 5) \Rightarrow 8$$

## #1 nice - 3 גזירות

## : λ-opt (לכון פטור) (\*)

```
((eq? (car pe) 'λ-opt)
  (with pe
    (λ (— argl opt body)
      `(λ-opt ,argl ,opt
              ,(run body `',(,@ argl ,opt) (cons params env)))))))
```

(תורה נבואה ותורת הרים נורדיים בתרבות היהודית)

(כ"ל עז' נסומנערם כראמיים יונק כראמיים)

æGJAGOE ryN fjl mænic fyls bjc- fvar / bvar / þvar ÞjND var pf 2 nÍKE

15. וְמוֹנָתָה תִּמְצֵא - וְבַיִתָּהּ מִזְבְּחָה תִּמְצֵא 2. וְלֹא תִּשְׁתַּחַטֵּא כְּבָשָׂן כְּבָשָׂן תִּמְצֵא

הנורא - קיון הטעון (3).

## Code Generation

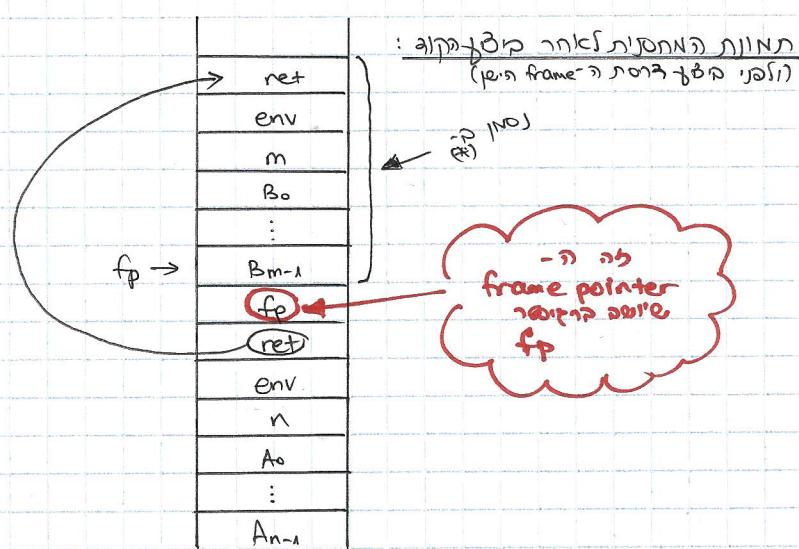
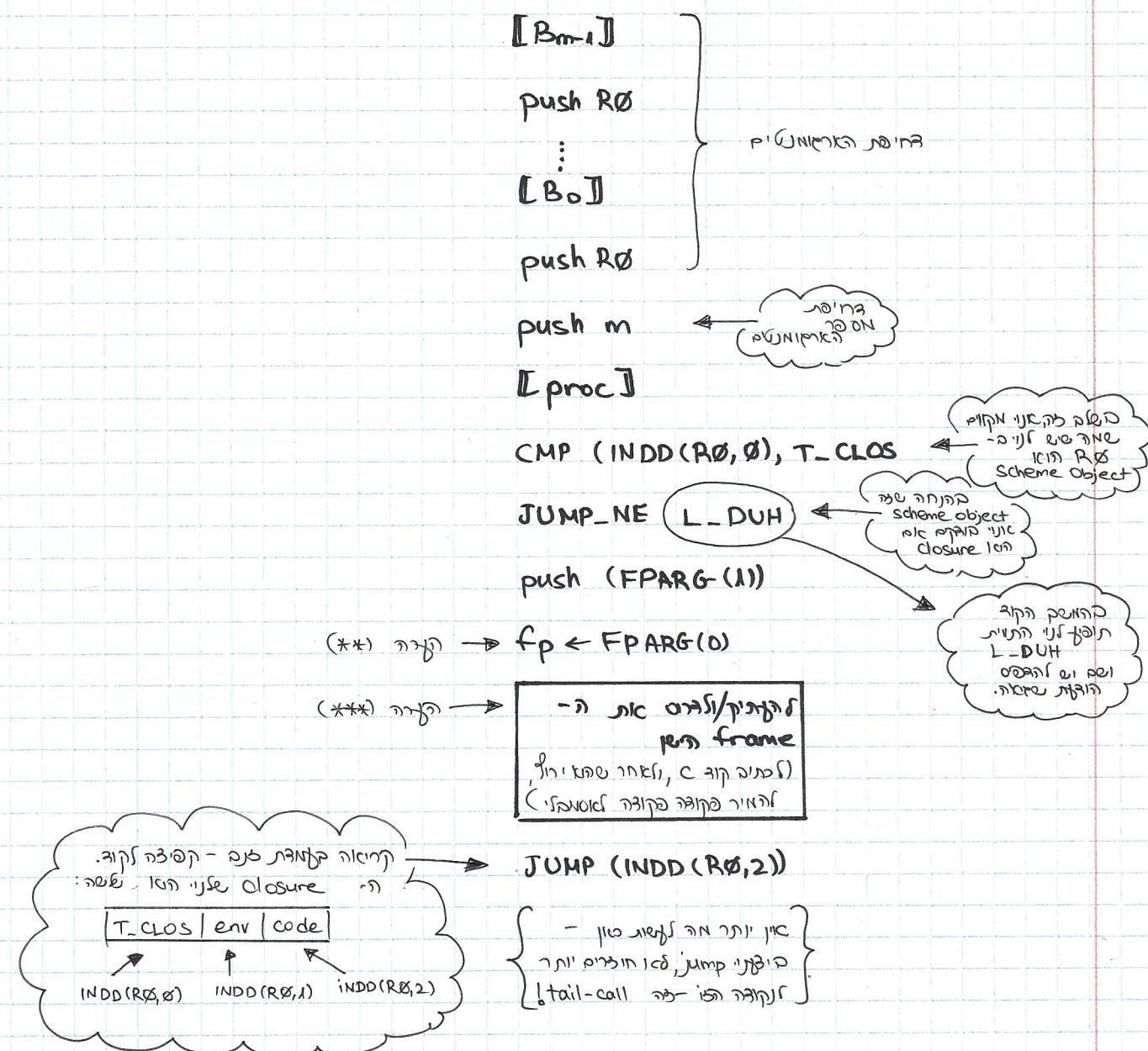
## סמליקת קבאות לבן

( $\text{tc-applic}$  proc ( $B_0 \dots B_{m-1}$ ))

$fp \rightarrow$	invariant region ( $*$ )
fp	
ret	
env	
n	
Ao	
:	
An-1	

הו? מילוי החלטה מילוי?

$\llbracket (\text{tc-applic proc } (B_0 \dots B_{m-1})) \rrbracket =$



ויקרא (\*\*): כו"ה נאכלת ה' כה, גבריה ורשותה, מתקין.

לפניהם, מילוי הפקה push fp יתבצע בראמה תחתית (בפער).

1130 INTON frame pointer-> reg fp גורוּפְרָטְרָגְרָם נֶתֶן וְגַם, pr

`F P A R G (0)`      "ה' נספ' מ'ת'ן, frame pointer -> ה' נספ' פיר'ם <

מacro מוגדר באמצעות פונקציית `macro` ב-`FP ARG(i)`

• ((p<sub>1</sub>) fp → ncp<sub>1</sub> f<sub>p</sub> →

frame-הנוקם ב-

• כיתה (\*\*\*): כה הראת כי מלהקם נר (\*)

!frame  $\rightarrow$  נחנוך בפונקציית  $\text{frame}$

בנוסף ל- $\text{p-frame}$  ~ה- $\text{ON-NO}$ -ה- $\text{frame}$  ~ה- $\text{N-NO}$ -ה- $\text{frame}$ . ה- $\text{N-NO}$ -ה- $\text{frame}$  ~ה- $\text{N-NO}$ -ה- $\text{frame}$

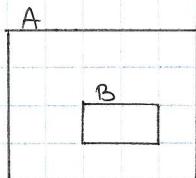
כיד מילוי פוליאם ג'לטס ג'לטס זיר (\*) ? כיד נ-ה א-ב ג'לטס

(בתק גיילרין)  $m+n = \Delta$   $n+m = \Delta$  (בתרום, כדי גורם נקיין)

# MAGIC

(יירן רונן נירן)

## : λ-Opt

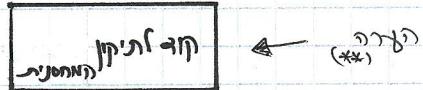


תרכות: פאר, ניידן ו-simple מתייכת לפא:

$\llbracket (\lambda\text{-opt } (a\ b\ c) \ r\ \text{body} ) \rrbracket =$

L-CLOS-B-37: push fp

$f_p \leftarrow s_p$



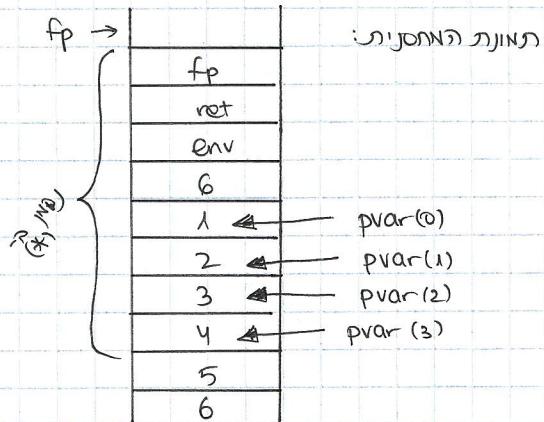
[body]

$f_p \leftarrow \text{pop}()$

return.

$f: (\rightarrow (a \ b \ c \ . \ r) \longrightarrow)$  : (\*\*\*) ???)

$$f(1 \ 2 \ 3 \ 4 \ 5 \ 6)$$



၅ : የዚህ ቀን እና ስምምነት በመሆኑ የሚያስፈልግ ይችላል (፫፭፻) ዓ.ም.

מי'ם צוותה? נו'תראן צו'ר ה'ליאו'ר ט'ג? ז' ב'כ'א:



•. ගැඹුම් සැස - 5 පෙරිය - ගැනීම හෝ ගැනීමේ මූලික පිළිබඳ නොවා.

מ-Const מוגדר ב-*ריבוע* כפונקציית *lambda* עם גורם אחד בלבד, λ-variadic פוג'ר מ-*ריבוע* "תולב" ימ-Const

• גְּדוֹלָה כָּלִיל גְּדוֹלָה תְּבוּרָה (לֹא גְּנוּמָה) קַדְשָׁה :

לעת 20.00 גנריי נס הנקודות - 91%

כ"ב "תירב" ז"ה (\*) נאותג'.

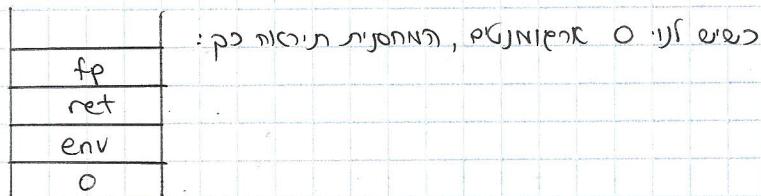
בכד מה יוכן גוונת גוף frame יי' נתקה באנטנה, אך הנטה וענין נקייה

כטבנגי, הנו ביך גומי רק 4 - 500kr.

# MAGIC - הַבָּשָׂר

Object Scheme Scheme Object - ה- נספחים MAGIC יגדיר

((> v —)) : ip ennej 'n



: ps . prar(Φ) - גורן מוגן מפני אם ↪

\* מינימום 1 - גובהו נמוך מהגובה של קאנון גוף צבוי.

מג'יק 2 מג'יק, MAGIC - 2 מג'יק.

$fp \rightarrow$	
	$fp$
	$ret$
	$env$
	$n$
	$to$
	$\vdots$
	$An-1$
	MAGIC

כoping 2, MAGIC אקלזיות ונהננות ←

## நிதி நெடுஞ்செலுத்தும் பார்வை

... nice fit fit

הנובע מהתבניות שבסדרת הנקודות נקרא **Cycle**.

יכוּן גְּדוּלָה מִכֶּם נַעֲמָה.

# וְיַעֲשֵׂה יְהוָה כָּלֵב אֶת-מִצְרָיִם

## (Unacademy) Loop Fusion - #1 നബ്ലിനഗൾ

```
for (i=0; i<100; ++i) {  
    A[i] = 5;  
}
```

## በኢትዮጵያ 2 የገዢ

```
for (i=0 ; i<100 ; ++i){  
    B[i] = 5;  
}
```

```
⇒ for (i=0; i<100; ++i) {  
    A[i]=5;  
    B[i]=5;  
}
```

ה-ארכיאולוגיה ה-הנ"ל גראן ג'יימס:

(בוחן): בוחן רגולרי של מילוי (גיאר, עיבודים) או update 100, or test 100 (בז'ן)

לפניהם הינה גוף הגוף (body) ומכאן שפה יפה מילויים.

(ניגן-גיאר עם ביצועה 2 ו- pipeline 2)

ב-3 סעיפים גודל הפלט מ-3 ל-6 (א)

program pointer יתבצע ב- Fetch .

- מה הפעולה? ואילו היא מושגת? נסמן ב- Decode .

. Fetch - Execute .

	$i$	$i+1$	$i+2$	$i+3$	...
Fetch					
Decode					
Execute					

לעומת פלט 3 סעיפים גודל הפלט מ-3 ל-6 (א)

כגון נזכר, מטרתו של ה- NTF היא:

	$i$	$i+1$	$i+2$	$i+3$
Fetch				
Decode				
Execute				
Retry				

3 סעיפים גודל הפלט מ-3 ל-6 (א)  
. (IFN pipeline).  
לעומת פלט 3 סעיפים גודל הפלט מ-3 ל-6 (א)  
. Retry פלט

כגון נזכר, מטרתו של ה- NTF היא:

לעומת פלט 3 סעיפים גודל הפלט מ-3 ל-6 (א) מטרתו של ה- NTF היא:

[INT] [INT] [INT] [INT] : סעיפים

לעומת פלט 3 סעיפים גודל הפלט מ-3 ל-6 (א) מטרתו של ה- NTF היא:

ולעתה נזכיר את ה- MMX .

ולעתה נזכיר את ה- MMX .


כגון נזכר, מטרתו של ה- NTF היא:

לעומת פלט 3 סעיפים גודל הפלט מ-3 ל-6 (א)

for (i=0; i<100; ++i){  
    A[i]=5;  
    B[i]=5;  
}

לעומת פלט 3 סעיפים גודל הפלט מ-3 ל-6 (א)

integer unit  
לעומת פלט 3 סעיפים גודל הפלט מ-3 ל-6 (א)

לעומת פלט 3 סעיפים גודל הפלט מ-3 ל-6 (א) מטרתו של ה- NTF היא:

for (i=0; i<100; ++i){  
    A[i]=5;

לעומת פלט 3 סעיפים גודל הפלט מ-3 ל-6 (א) מטרתו של ה- NTF היא:

for (i=0; i<100; ++i){  
    B[i]=5;

לעומת פלט 3 סעיפים גודל הפלט מ-3 ל-6 (א) מטרתו של ה- NTF היא:

}

לעומת פלט 3 סעיפים גודל הפלט מ-3 ל-6 (א) מטרתו של ה- NTF היא:

```
for (i=0; i<100; ++i) {
    A[i]=5;
}
```

לעתה נרעד במשתנה: בזאנו

```
for (i=0; i<150; ++i) {
    B[i]=5;
}
```

: 2- גודל התרומות קידם לתפקידו של B[i], לא B[150]. בזאת כי הוחיה כרך גודל התרומות.

```
for (i=0; i<150; ++i) {
    B[i]=5;
}
```

$\Rightarrow$

```
for (i=0; i<100; ++i) {
    B[i]=5;
}
for (i=100; i<150; ++i) {
    B[i]=5;
}
```

נולדו: סדרת הפעמים שבאותו רגע הפלט מופיעים.

```
for (i=0; i<128; ++i) {
    A[i]=∅;
}
```

### עונרולינג - #2 גודל גודל

בזאנו

השאלה היא: מה עשו?

. מטרתנו היא לסייע לך להבין מושג "פונקציית אינדקס".

בשאלו, פונקציה כזו נקראת, אזי כדי לרשום אותה:

```
for (i=0; i<128; ++i) {
    A[i]=∅;
}
```

$\Rightarrow$

```
for (i=0; i<128; i+=2) {
    A[i]=∅;
    A[i+1]=∅;
}
```

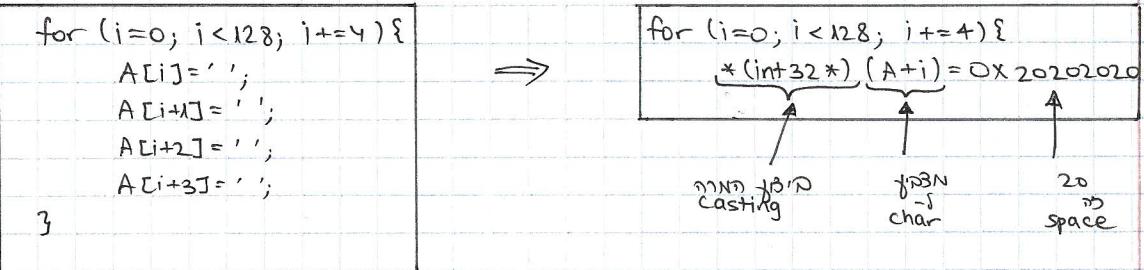
הגורוען: נניחו רציתך בזאת ש-body (הinstructions) מוגדרת כפונקציית אינדקס (בזאת ש-body מוגדרת כפונקציית אינדקס).

### - #3 גודל גודל

```
for (i=0; i<128; i+=4) {
    A[i]='';
    A[i+1]='';
    A[i+2]='';
    A[i+3]='';
}
```

לעתה פונקציית אינדקס מוגדרת ככזה:

אם תשים פונקציית אינדקס, פותח את הכתובת כפונקציה נקודה, למשל  $i=0$ , ותשים פונקציית אינדקס + 4 ותשים פונקציית אינדקס + 8. לאחר מכן התחייב ליכוח פיקחות.



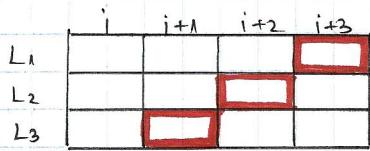
(כזה בוגר מובן gcc, נאכטן)

...הנכנסה מטרת מינימום של מילוי מemory ב-4 bytes

### : software pipeline - #4 הדריך נאכטן

for (*i*=0; *i*<M; ++*i*) {  
 L<sub>1</sub> → A[*i*] = f(*i*) \* g(*i*) - A[*i*];  
 L<sub>2</sub> → B[*i*] = A[*i*] + B[*i*];  
 L<sub>3</sub> → C[*i*] = C[*i*] \* (A[*i*] - B[*i*]);  
}

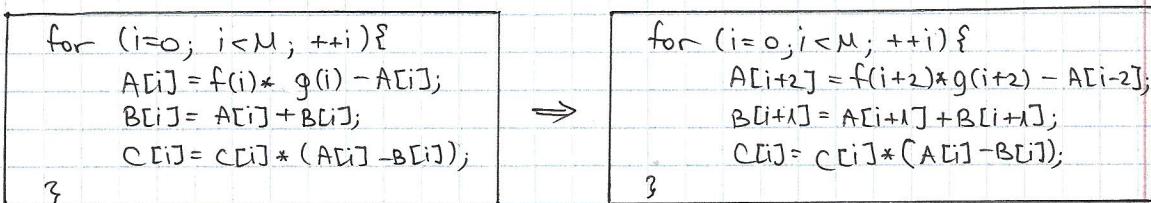
L<sub>1</sub> → L<sub>2</sub> → L<sub>3</sub> ? קוויאר מה?



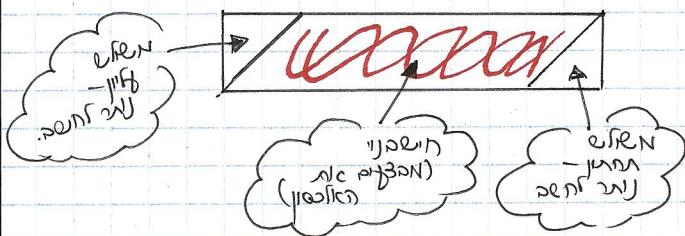
המזהה הינה גלגול תרמיות:

- גלגול תרמיות נאכטן

לז → pipeline →



...הנכנסה מטרת מינימום של מילוי מemory ב-4 bytes



...הנכנסה מטרת מינימום של מילוי מemory ב-4 bytes

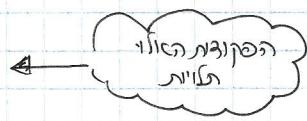
(\*) תרגיל: קודו את פונקציית body ⇒ נסמן מטרת מינימום של מילוי מemory ב-4 bytes

(\*) תרגיל: קודו את פונקציית body ⇒ נסמן מטרת מינימום של מילוי מemory ב-4 bytes

13. ריצוף גזירה (Software pipeline) גזרה מודולרית

$S = " "$ ;

```
for ( — ) {  
    S += "A" + i;  
    S += "B" + i;  
    S += "C" + i;  
}
```



(\*) הערה - סדרת הפעולות יצר נסיגת פאיבר (fibonacci sequence).

### Strength Reduction - #5 נסיגת פאיבר

int n = m \* 32;

גנרטור גזירה נסיגת פאיבר על ידי יצרן.

. . . shift  $\leq 5$  גנרטור פאיבר  $m \ll 5$  נסיגת פאיבר:

$2^5 - 1 = 31$  shift (גנרטור פאיבר)  $m \ll 5 + m = 31$  גנרטור פאיבר  $m * 31$ :

. . . cycle 56 גנרטור פאיבר - shift / plus . . .

. . . cycle 56 גנרטור פאיבר - shift / plus . . .

כך, לא נזקם.

לעתים קיימת גזרה מודולרית (Software Pipeline) כפולה (double pipeline).

בפועל פאיבר  $m = 2^n - 1$  גנרטור פאיבר - פאיבר נסיגת פאיבר.

לעתים קיימת גזרה מודולרית (Software Pipeline) כפולה (double pipeline), כאשר פאיבר נסיגת פאיבר, גזרה מודולרית (Software Pipeline) כפולה (double pipeline).

ונאכטן גזרה מודולרית (Software Pipeline) כפולה (double pipeline).

Top LevelHash Table

אנו נזכיר Hash Table כטבלה שפונקציית המAPPING מפינה בין סימולרים לVALUES.

כעת גיבים לנו פונקציית גיבוב.

פונקציית גיבוב מפה בין הכתובת הילוגית לכתובת הפיזית.

$f(x_1) = f(x_2)$  אם  $x_1 = x_2$  !!!!!!!!

גירוי פונקציית גיבוב.

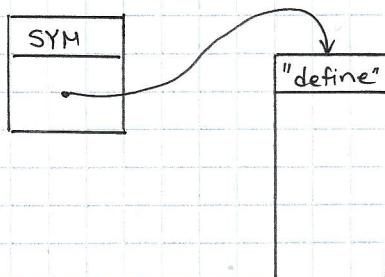
לפנינו מופיע סימולר  $\lambda$  ופונקציית גיבוב מפה בין הכתובת הילוגית לכתובת הפיזית.

( $\lambda$  יתגדר כsymbol בTABLE).

$\text{string} \rightarrow \text{symbol}$  : Symbol table  $\rightarrow$  sequence

> ( $\text{string} \rightarrow \text{symbol}$  "define")

: TABLE



hash table -> גיבוב Scheme  $\lambda$  בפונקציית define (\*)

: TABLE

> (define foo  
    ( $\lambda (x) (x \times x)$ ))

, symbol מוכן בTABLE, פון-טוקן מוכן בTABLE Scanner  $\rightarrow$

hash table -> initial state בTABLE

> (define is-in-sym-table?  
 (λ (—))

: פונקציית

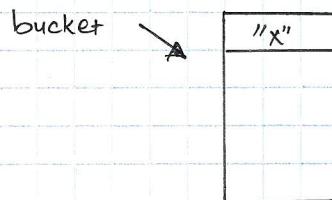
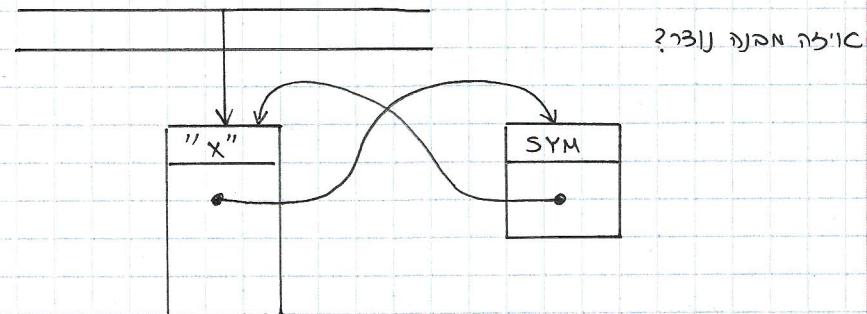
השאלה היא: אם גיבר true, מילוי true, symbol או false מילוי false?

השאלה היא: אם גיבר true, מילוי true, symbol או false מילוי false?  
 (איך, בפונקציה הטעינה תחת פונקציית Scanner?)

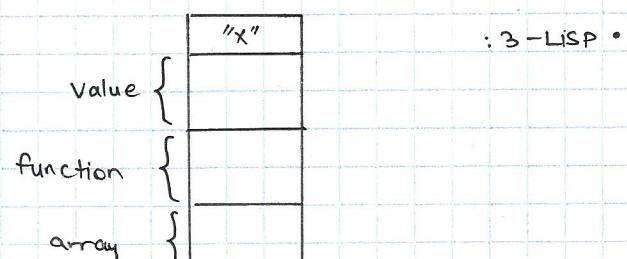
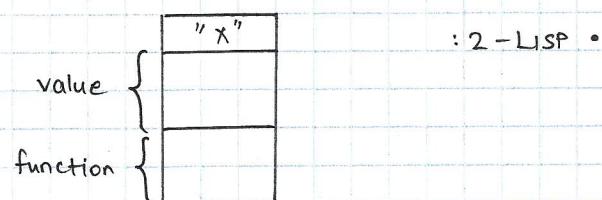
> (define x 'x) ←  
 (השאלה היא: מילוי true?)

> x

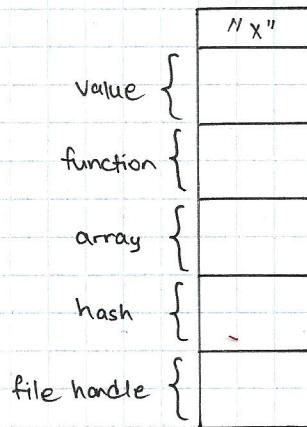
x



? bucket → true? ←



scalar	\$x
function	&x
array	@x
hash	%x
file handle	<x>



## 5 - LISP

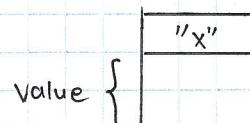
(\*) כוונת פונקציית  $\text{IF-COND}$  היא  $\text{IF-COND} \equiv (\lambda x. \text{COND } x)$ .

השאלה היא: (\*) הו שורש של  $\sqrt{3}$  הו השאלה השאלה

new 's()' → (defun x () s)



value to FON bucket  $\rightarrow$ , Scheme  $\rightarrow$  NIC  $\rightarrow$ , I-LISP ION scheme (\*)



מאריךBucket (0, 100) null, מכירBucket (100, 200), ...etc, מאריךBucket (N-1, N)

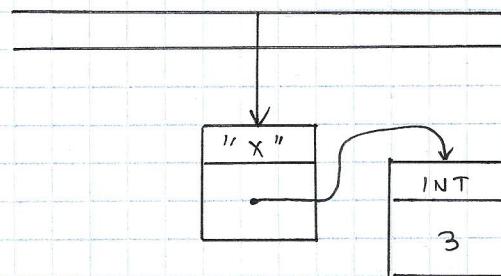
2.5. (ט) נapis pointer גורר הערך.

```
> (define x 3)
```

ט' נס

> X

3



> (define x 3)

לען 13

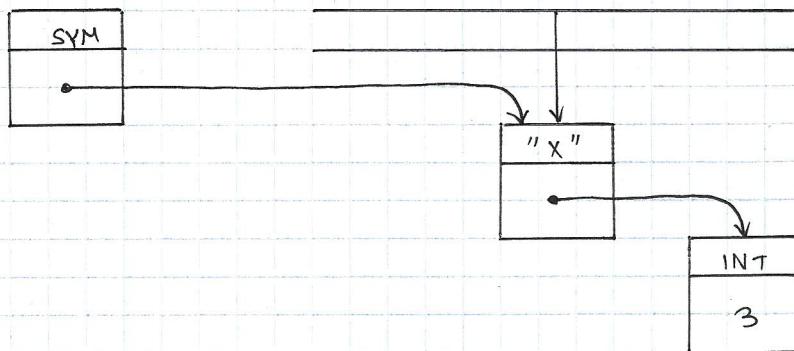
> x

הנתקה לנו כתיבי כוונת x/  
- ה- Code generation  
- נתקה כתיבי שפם נספחים -  
x נספחים bucket

3

> 'x

x



(היגון נציג נתקה כתיבי symbols שנשי נספחים - x : נהווים הטעינה טיה.  
אם נתקה כתיבי symbol הינו נספחים צו (או מילויו אפס) או הטעינה יתירה, קוליזיה פגוי מתקבז  
על רוכסן פגוי).

! חישוב: ה- symbol table קיימת בזיכרון!

> ( x () x)

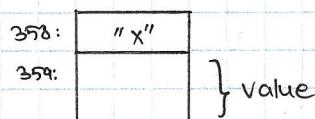
לען 14

# < CLOS >

: x רצף נתקה גואט. צו, זיך גואט גיא שיכרתו גיא רצף x :

- גיא רצף  $\leftarrow \begin{cases} \text{MOV } (\text{R0}, 359); \\ \text{MOV } (\text{R0}, \text{IND}(\text{R0})); \end{cases}$   
 $\text{R0} \leftarrow *359$

. 359 רצף גברת נתקה גיא רצף x . 358 רצף x רצף bucket  $\rightarrow$  הטעינה מ- 358



## לינק סכמי פירוט (pirrot for scheme)

$\llbracket (\text{define } x \ e) \rrbracket =$

:define פירוט

$\llbracket e \rrbracket$

MOV R1, 458

$R1[1] \leftarrow R\emptyset$

$R\emptyset \leftarrow \text{SCB\_VOID}$

פירוט הינו מערך  
value → א. ב. ג. ד. א. ב. ג. ד.  
 $\llbracket e \rrbracket$  מערך זהה

define  
void מערך

. x בפירושו הוא סימן השוואתית. ונקודות השוואתית NC מושבת לvalue hashtable → NC/sites יוציאו מערך (458) ו- x בפירושו מערך sites. פירוט הינו מערךsites, א. ב. ג. ד. א. ב. ג. ד.

$\llbracket (\text{fvar } x) \rrbracket =$

:fvar פירוט

$R\emptyset \leftarrow 458$

$R\emptyset \leftarrow R\emptyset[1]$

CMP (R $\emptyset$ ,  $\emptyset$ )

JUMP\_EQ Error\_Undefined

R $\emptyset$  → א. ב. ג. ד. א. ב. ג. ד. א. ב. ג. ד. א. ב. ג. ד.  
x בפירושו NC

$\llbracket (\text{set! } x \ e) \rrbracket =$

:set! פירוט

MOV R1, 458

CMP R1[1],  $\emptyset$

JUMP\_EQ Error\_Undefined

$\llbracket e \rrbracket$

MOV R1, 458

MOV R1[1], R $\emptyset$

MOV R $\emptyset$ , SCB\_VOID

מי יתלה פירוטה פירוט?

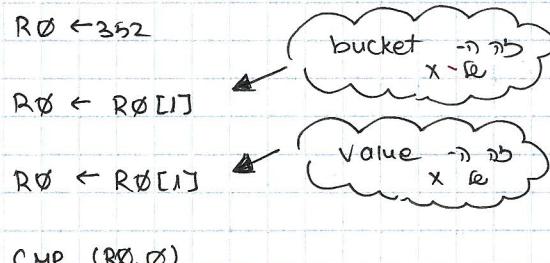
ptr pointer if it's diff to current diff in hash table → NC if it's diff to current diff in NC

## Scheme $\rightarrow$ סימולטור

T-SYM |  $\emptyset$

:352 סימולטור, הוראות

$\llbracket (\text{fvar } x) \rrbracket =$



JUMP\_EQ Error - Undefined

Code generation  $\rightarrow$  מעתה נזכיר, בפונקציית  $\text{hash\_table}$  מילוי פון גער.

352 כוונת Symbol  $\rightarrow$  סימולטור ימצא סימולטור, ויכלול סימולטור.

v ((int\*) 352[1]) = hash ("v"); יממש תבנית שפיטר.

parse tree  $\rightarrow$  פג שמיינט (המבנה הפיזי של המילים) מעתה נזכיר  $\text{Symbol} \rightarrow$  סימולטור.

- מילוי פון גער תכנתם של סימולטור, רשיון הסימולטור, ורשיון סימולטור.

MEM[0]  $\rightarrow$  pointer

(define var 352) סימולטור.

15/1/2013

## דוחף, ביציה - הרצאה 12 (המשךה!)

: מומס  $e'$ , Seq פאר, code generation  $\rightarrow$  מזהה(\*)

$$[(\text{seq } (\text{E}_1 \dots \text{E}_n))] =$$

$[\text{E}_1]$

CALL (WRITE\_SOBJ(R $\emptyset$ ));

$[\text{E}_2]$

⋮

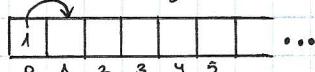
$[\text{E}_n]$

CALL (WRITE\_SOBJ(R $\emptyset$ ));

Lend : PUSH (R $\emptyset$ ); מומס  $e'$ , main ל R $\emptyset$  : מזהה(\*)  
CALL (WRITE\_SOBJ);

### מכה וטיכון:

טיכון, נטיגת אפס, נטיגת גולן, נטיגת סטטוס, start-machine  $\rightarrow$  start-machine(טיכון, טיכון).



ויהי T.

כדי להציג נקיון בפ' קאצין סימן נטיגת T (טיל שורה וריאציה).

לעת גז, יוציא גז עזנה, בולט נטיגת 10, ולו (טיכון גולן) גולן נטיגת 10, ולו (טיכון גז עזנה) גז עזנה.

Void	Bool	#f	#t
T_VOID	T_NIL	T_BOOL	0
10	11	12	13

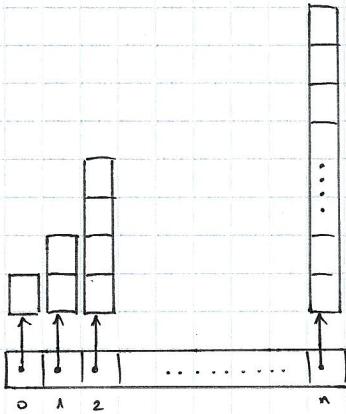
Bool	#f	#t
T_BOOL	14	15
14	15	16

MOV (R $\emptyset$ , 14) יוציא טיכון #t ותפקידו תרעה, boolean? (true), על מנת בז'  $\leftarrow$

MOV (R $\emptyset$ , 12) יוציא טיכון #f ותפקידו תרעה, boolean? (false), על מנת בז'  $\leftarrow$

## :Paging

הוּא גָּתֵן וְלִיכְמֹן גַּתְּהָם. מִזְבְּחָה רַיִשׁ כִּילְעָם בְּלִיכְמֹן גַּתְּהָם.



linked-list を使う場合: 각각의 노드는 같은 형식을 가진다.  
 예: 각각의 노드는 같은 형식을 가진다.  
 $P \cdot 2^j$  와 같은 형식을 가진다, 예:  $P_0, P_1, \dots, P_{2^j-1}$

**נהוג code generator -> קובץ יוצר קוד**

```
(define code-gen
  (lambda (pe env params)
    (cond
      ((pe-const? pe) (cg-const (cadr pe)))
      )))
```

הנחיות בפונקציית code-gen:

- env → הסביבה
- params → פרמטרים
- type-tag → סוג טג

(define cq-const

(> (c)  
(cond  
=====  
=====  
)))

כִּי תְּבַקֵּשׁ כִּי  
תְּבַקֵּשׁ כִּי  
: const  
num, symbol, boolean, ...

: Code-generator  $\rightarrow$  NL and FIN PIPE, PIPE

code-generator → Se zuse . 1

גַּתְתָּא - אֶתְנָהָר .2

Or, if, seq - ערך נאכ"ה .3

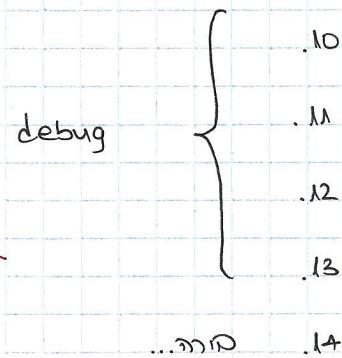
4)  $\lambda x. f(x)$  ("סבב" א, ב, כ, ד) bvar, pvar, applic,  $\lambda$ -simple- $\lambda$  UNICITY 4  
. (annotate- $\lambda$

$\lambda$ -variadic,  $\lambda$ -Opt גיאיכה ר' .5

#f (cond ((*if* (*lambda* (*x*) (*apply* (+ *x* 1) (*x*)))) (*lambda* (*x*) (*apply* (+ *x* 1) (*x*)))))) .6

define, fvar, const      (לNICII ד- .7

primitives .8



## כִּי בְּ נַעֲמָה צָרָה הַקּוֹנְגָּרָט?

- **pr Expression** -> **pr Sub-expression** -> ... -> **pr Leaf**

(define foo

(\lambda(e))

(cond ((pair? e)  
`(`, @ (foo (car e))  
,@ (foo (cdr e))  
, e))

$((\text{Vec} ? \ e) \quad \underbrace{\quad}_{\text{פונקיה של עליון}}$

(else `((,e))))))

[

(ML-כ) : ניהובים

(c, addr, [...])

10

- תרגום של הפעלה ב-  
sense (ב-  
ב-  
- 13,19 : pair - . ו- נספחים

מזהב:  $c - \frac{1}{\lambda^2}$  - Addr כיריך נסוי (הנחייה מינימלית)

- (שאלה נוספת, מילוגים נס' 6 הערך היא [...] )

[T\_INT,5] : int DB PLC, S8 NO  
[T\_BNP,1-10] : DWord DI PLC

eq?

? פילע רט פילג . "abc" , "abc" , כ-ז ליהב 2 יפער זיכר (\*)

לה נוֹאכַר גָּהָת לְנֹאכַר.

ריכוזה הימראתו). לה תוו' ר' לוי היראיה צייראקי כי זו לא.

אקס, וויליאם הרטמן, פול נוימן - ג'ון טריילו

ללא פירג 550 ₪ ומכאן ש 2 גנרטורים מילויים.

lookup ב- "abc" מחרטת קורסרים. ייצר, פונקציית  $f(x)$ ,  $x$ .

(6) **என்ன என் கிடைக்காது - ஏ கொயில் அல்ல கெள்ளும்.**

היאוואר כתקינה או נפח גזומרייה (picric acid), רותם גאנט נלען (Gant's reagent).

כ"א) סקאניה לא רצה גזע שלן וזה מוגן בתקופת

אם רציתם להציג פה 2 מחרוזות, כל אחת ככזו: נקי בפיון, ריאן גיבנסן וכו'

char = ?

$$\text{char-ci} = 2$$

string = ?

string - Ci = ?

א) ג' כב' הולונן נ-?eq - מ- פתקתי רלוטיבר אונליין פונק נסחים.

טיקון: ו. מה יא.

2. זה יכול להיות MCP1 או גראטינ (Granul

3. जैविक जल संग्रह क्षमता का निर्णय कैसे किया जाता है?

2. נגזרת מושגית:  $\lim_{x \rightarrow x_0} f(x)$  אם  $f(x) = g(x)$  עבור  $x \neq x_0$  ו- $f(x_0) = L$ .

2. ስልክና ቀን ገዢ እና የሆነት (ይህንን አገልግሎት የሚያሳይ).

# Garbage Collector

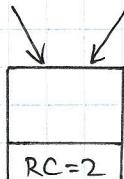
דריכים חיצוניים נאומניים מחייבים פגיעה בפערם.

הרי הם מזכירים מילויים - צער, כעס וקנאה לתוכן נט ערבה, והנאמה במאמר ישוחרר היכרין.

GENERIC  $\Sigma$ -garbage collector  $\leftarrow$

(\*) (ב) ו (ג) נסיבותן לא זריזותן של האובייקטים. מילויים לא זריזותם של האובייקטים.

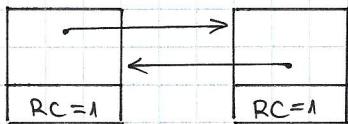
reference count : הנק סמייסטיקו



RC = reference count

כזה,  $R=0$  ו-  $C=0$  נס庭תית גליות נטולות.

. garbage → ~~pen~~



הנחתה (ולקוטר קתדרות) היה דמוקרט נאשוויט,

fishes milk eggs rice pickles onions, garbage left

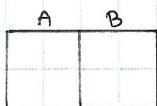
גנרטור ניומני ל.ג.

ו' 2 גזעניכן נס ציון

கால்களை வெட்ட அடிப்பாக நான் சுதாமல் - Mark & Sweep .1

9 fig 9 top level -> N. Africa, India, China, Indonesia, Australia

וְיַעֲשֵׂה יְהוָה כָּל־אָמֵן



וְעַתָּה תִּשְׁכַּח נַעֲמָד - Stop & Copy .2

כלהי הנקב בפרק חוזק A, כוואר זעג הנקב בפרק

garbage inc. income A -re הכנסה גיורו א' פירמה

הנתקים מכם. ב- ג' ינואר הודיעו הפלגון ותפקידם שלם.

כמי לאב, כמי ראן גנלאד נטיכון מעריך B, מעריכים דיכוחו אויר

```
(define zablanc
  (lambda (n x)
    (if (zero? n)
        x
        (zablanc (- n 1) (make-vector n))))))
```

בז'אנר ג'טלי

גָּרְבָּגְגַּתְּרָה וְלִפְנֵיָהּ

אם מזכיר לך garbage collector, שום מה זה נכון לך (garbage collector)

נו, פונקציית ה- garbage collector, היא הינה שמיינטן (keep), כלומר

לעתים מיותרת לזכור נסחאות, דינמי-נו, n-garbage-collector, ועוד, X, מילוי-

מיינטן, מילוי מילוי ג'טלי.

ולא רק X בז'אנר ג'טלי garbage collector (בז'אנר ג'טלי)

כלומר, מילויים יקרים יותר הם יוצרים short living object

מילויים זולים הם long living object.

בז'אנר ג'טלי ג'טלי

```
(define goo
  (letrec ((foo (lambda (n m k small large)
                  (if (zero? k)
                      (let* ((m+m (+ m m))
                             (small (make-vector 10))
                             (large (make-vector m+m)))
                        (foo (+ n 1) m+m m+m small large))
                      (foo n m (- k 1) small large)))))))
```

(lambda (n m k small large)

מילוי זול  
מילוי יקר

מילוי זול  
מילוי יקר

(let\* ((m+m (+ m m))

(small (make-vector 10))

(large (make-vector m+m))))

(foo (+ n 1) m+m m+m small large))

לצורך הרצאה  
לצורך הרצאה  
לצורך הרצאה  
לצורך הרצאה

מילוי זול  
מילוי יקר

(let ((small (make-vector 10)))

(foo n m (- k 1) small large))))))

מילוי זול  
מילוי יקר

(lambda () (foo 1 1 1 #f #f))))))

פונקציית זיכרון - פונקציית זיכרון. ככלהותן ייחודה בז'אנר.

29/11/2012

## חכפיילץיה - שיעור העשרה:

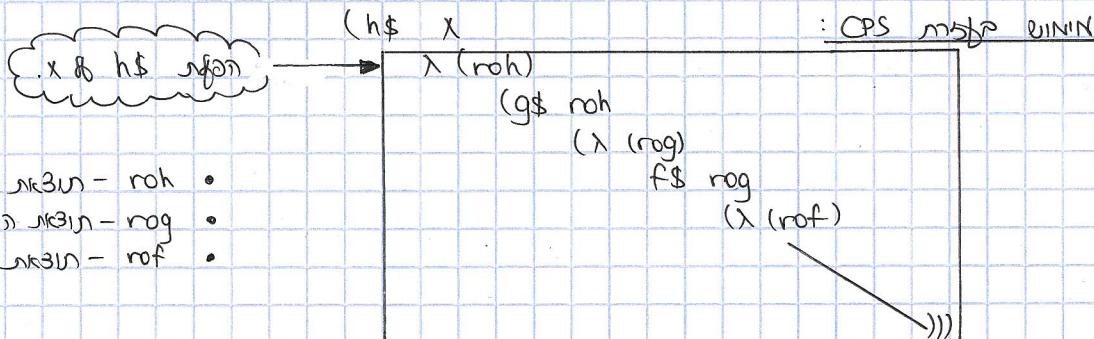
### CPS

$(f(g(h(x))))$

רמז גוף נסיעה הצעה:

$(\text{let } * ((\text{roh} (h x))$   
 $\quad (\text{rog} (g \text{ roh}))$   
 $\quad (\text{rof} (f \text{ rog}))$   
 $\quad \text{rof}))$

: CPS ~~יכל~~ עיןין



$(\text{fact } \overline{n}) \Rightarrow 120$

: בזינקן

זה תחוליר גנטאותה הולמי ?  
~~יכל~~ אם גנטה גנטה נסיעה נסיעה  
~~יכל~~  $\text{continuation}$  ⇒  $(\text{fact\$ } \overline{n} \text{ K})$

↑  
 continuation

(define fact\$  
 $\quad (\lambda (n \text{ ret-n!})$   
 $\quad \quad (\text{if } (\text{zero? } n)$   
 $\quad \quad \quad (\text{ret-n! } \perp)$   
 $\quad \quad \quad (\text{fact\$ } (- n 1))$   
 $\quad \quad \quad (\lambda (n-1!)$   
 $\quad \quad \quad \quad (\text{ret-n! } (* n n-1!)))))))$

: תעריך מוכחים

$(\text{fact\$ } \emptyset \text{ K}) \Rightarrow (\text{K } 1)$

: fact\\$ ~~יכל~~ גנט גנט

$(\text{fact\$ } n \text{ K}) \Rightarrow (\text{fact\$ } (- n 1))$

Continuation →  $\boxed{(\lambda (n-1!)$   
 $\quad (\text{ret-n! } (* n n-1!)))})$

: let  $\text{if} \text{else} \leftarrow \lambda \text{ if } \text{not} \text{ if } \text{then} \text{ else}$

$(\text{let } ((\text{n-1! } (n-1!))$   
 $\quad (\text{ret-n! } (\text{K } n \text{ n-1!))))$

Continuation → of  $\text{N-1! K}$   
 $n-1!$  if  $\text{N-1! K}$  if  $\text{N-1! K}$   
 $(\text{K } n \text{ n-1!})$  if  $\text{N-1! K}$

(define fib\$  
 (lambda (n k)  
 (if (< n 2)  
 (k n)  
 (fib\$ (- n 1)  
 (lambda (fib-n-1)  
 (fib\$ (- n 2)  
 (lambda (fib-n-2)  
 (k (+ fib-n1 fib-n2)))))))))))

continuation n < 2  
 continuation - n 1  
 continuation - n 2

$(+ (\text{fib} (- n 1)) (\text{fib} (- n 2)))$  : הרצאה על CPS ו-CC מגדנ

אם היחסון מליילו מוביל ל Side-effect או גורם לא תקין (אפקט לוואי), ניתן לתקן אותו (גירעה).

מגנטוגרפיה - IC

‘**విజ్ఞాన వైదిక శాస్త్రముల ప్రాచీనతా**’ అనే పేరును కలిగిన ప్రాచీన విజ్ఞాన శాస్త్రముల ప్రాచీనతా అనుభంగములు.

```

(fib$ (- n 2)
  (λ (fib-n-2)
    (fib$ (- n 1)
      (λ (fib-n-1)
        (k (+ fib-n-1 fib-n-2)))))))

```

הפטה כת' "כלו". מאר סאר וגיהנום.

$(\text{map } f \text{ s})$  :  $\text{map}$  מגדיר פונקציית Side-effect - פונקציית גנרטור שמשתמש ב-

((cons (f (cons))  
      (map f (cdr s))))

. side-effect → ספקה, עיון נזנץ גוף, מוגבך 2 "טס"cons נזנצה כה,

# ନୀତି କେ ପରିଚାଳିକ

רקע יער חאנז – מ وقت מלחמת העולם הראשונה ועד תחילת המאה ה-20 היה יער חאנז אחד מיערות-

◀ כביחת גרעין גרעינית בפער - רעיון גרעינית (frame) (GNC, BGC).

```

int ack (int a , int b) {
    if (a==0) return b+1;
    else if (b==0) return ack (a-1 , 1);
    else return ack (a-1, ack (a,b-1));
}

```

בז'ן:

הינתן גורם בק →  
הינתן גורם בק ←  
הינתן גורם בק  
הינתן גורם בק →  
הינתן גורם בק ←  
הינתן גורם בק

$A(a,b) = \begin{cases} b+1 & a=0 \\ A(a-1,1) & b=0 \\ A(a-1, A(a,b-1)) & \text{else} \end{cases}$

פירושה ט': (תוקן פונקציית ack):

פוך ריבוי גורם מפונקציית ack לפונקציית ack כפולה.

```

int ack (int a , int b) {
L:    if (a==0) return b+1;
    else if (b==0) {
        --a;
        b+=1;
        goto L;
    }
    else {
        b=ack (a,b-1);
        --a;
        goto L;
    }
}

```

בז'ן:

הגביה כת', ON הניתן (הנחותית) ומי' נטה. ת' מ' מ' אוניברסיטת ירושלים.

ר'continuation -> גורם בק של גורם בק של גורם בק של גורם בק ? CPS -> גורם בק

continuation -> גורם בק של גורם בק של גורם בק של גורם בק של גורם בק

```

(define ack$ 
  (λ (a b k)
    (cond ((zero? a) (k (+ b 1)))
          ((zero? b) (ack$ (- a 1) K))
          (else (ack$ a (- b 1)
                    (λ (r)
                      (ack (- a 1) r K))))))

```

continuation -> גורם בק של גורם בק של גורם בק של גורם בק

continuation -> גורם בק של גורם בק של גורם בק של גורם בק

continuation -> גורם בק של גורם בק של גורם בק של גורם בק

continuation -> גורם בק של גורם בק של גורם בק של גורם בק

CPS ↔ נציגות

לע' ו' f (g y)

: K continuation of CPS -> ON

```

  (g$ y
    (x (rog)
      (h$ x
        (x (roh)
          (f$ roh rog
            (x (nof)
              (K nof)))))))
```

(\*) → JNOJ

הנימוק:  $\lambda x. f(x)$  הוא גורם ב- $\lambda$ -תuring מודולו אפליקציה.

↳ (חטף) קבוצה נספחתה על ידי קבוצה אחרת (בהתאם לדרישות).

$$(\lambda \frac{(\text{not } k)}{k}) \Rightarrow k \quad : \text{N15D}$$

כדו, מילויו נקבע בהתאם כ- CPS.

## CPS כוכייתה מתרן גן

כזה ימיה שפנוי סוכנותי נבדקה לא. פיתח גיברלטר נס סוכנותי מער

.зкј ртн

$$\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}^n = \begin{pmatrix} F_{n+2} & F_{n+1} \\ F_{n+1} & F_n \end{pmatrix}$$

- ביצוע סעיפים: ונזץ

זאת מה אמור? כי אם זו עיון וכאן יש לנו קבוצה נוספת (ג). נתקה בפיה (ב) מעתה (ב) מעתה.

(λ(a b c d)...) 모양의 4 번째 Continuation → 어떤 결과가 있는지 알 수 있다.

$$\cdot \left( \begin{pmatrix} a & b \\ c & d \end{pmatrix} \text{ דוגמאות } \right)$$

(define fib

```

λ(n net-abcd)
  (if (< n 2)
    (net-abcd 1 1 10)
    (fib (/ n 2))
  λ(a b c d)

```

הברוחה היא לא חישובית רק מילוי.  
ולו נשים מילוי נסירה מה  
.heap -> (כל נסירה כ-NULLptr)

continuations CPS : future continuation - function lifting }

בז' נג-כינוי נס' דקון ס' ל-ל-ל-ל-ל:

$\lambda$ -Opt  $\leftarrow$  a b c.d      "pt.

$\lambda$ -simple  $\Leftarrow (a \ b)$  γρ.

Continuation - ↳ proper list

Continuation - ↳ improper list

Continuation - ↳ list

continuation

```

(define foo
  (λ (argl)
    (cond
      ((pair? argl)
       (foo (cdr argl))
       (λ (s a)
           (k-imp (cons (car argl) s)
                  a)))
      ((λ () (k-imp `(, (car argl) (cdr argl)))))))
      ((null? argl)
       (k- pro))
      (else (k-atom))))))
  
```

בונארת הרכז - פרי גומיניה: a b c d .r.

Diagram illustrating the continuation mechanism in Scheme:

- The stack frame contains the environment pointer  $r$  pointing to  $(b\ c\ d\ .\ r)$ .
- The current frame contains the expression  $(b\ c\ d)$ .
- A continuation pointer  $k\text{-imp}$  points to the lambda expression  $(\lambda(s) a)$ .
- A call to  $\text{cons}$  is shown with arguments  $((a\ b\ c\ d))$  and  $a$ .
- The continuation  $k\text{-imp}$  is passed as an argument to the lambda function, which is then applied to  $s$ .

foo - function

```
(foo arg1
  (λ () '(λ-simple , arg1 →))
  (λ (s a) '(λ-opt , s , a →))
  (λ () '(λ-var , arg1 →))
)
```

## Pattern - Matching

ଏହାର ପରିମା କେବଳ 2-୩ ମିନିଟ୍‌ରେ ଥାଇଲା ଏବଂ ଏହାର ପରିମା କିମ୍ବା ଏହାର ପରିମା କିମ୍ବା

ב-**pattern matching** (הנ' פטנ'ר) כל מחר גאנדרה (ווען) מיר פאָך.

גאומטריה:  $\Rightarrow \boxed{?} = b$

במה מנגה יתעורר צעדי נחורה כז' עלייה מתחילה (באותה תקופה, רואין יי"ח ע"ג ?)

. S-expressions  $\lambda$ -expressions (closure) סכטער  $\lambda$

2. עסוק CPS  $\Rightarrow$  מנגנון PC ② ① ו (טנה) 2. מנגנון (\*)

. ret-fail , ret-success : continuations

Nov 29, 12 16:32

[pattern-matcher-with-example.scm](#)

Page 1/2

```

1 ;;; matcher.scm
2
3 ;;; Programmer: Mayer Goldberg, 2012
4
5 (define match pattern-matcher
6   (letrec ((match e = s-expression
7             (λ (pat e ret-vals ret-fail)
8               (cond ((and (pair? pat) (pair? e))
9                      (match (car pat) (car e)
10                         (λ (vals-car)
11                           (match (cdr pat) (cdr e)
12                             (λ (vals-cdr)
13                               (ret-vals
14                                 (append vals-car vals-cdr)))
15                               ret-fail)))
16
17                 ((and (vector? pat) (vector? e)
18                       (= (vector-length pat) (vector-length e))
19                         (match (vector->list pat) (vector->list e)
20                           ret-vals ret-fail)))
21
22                 ((procedure? pat)
23                  (let ((v (ret-vals v)))
24                    (if v (ret-vals v) (ret-fail))))
25
26                 (λ (pat e ret-with ret-fail)
27                   (match pat e
28                     (λ (vals) (apply ret-with vals))
29                     ret-fail)))))))
30
31 (define ? guard
32   (λ (name . guards) no-name (*)
33     (let ((guard?
34           (λ (e)
35             (andmap
36               (λ (g?) (g? e))
37               guards))))
38       (λ (value)
39         (if (guard? value)
40             (list value) #f)
41           #f))))))
42
43 ;;; composing patterns
44
45 (define pattern-rule
46   (λ (pat handler)
47     (λ (e failure)
48       (match pat e handler failure))))
49
50 (define compose-patterns
51   (letrec ((match-nothing
52             (λ (e failure)
53               (failure)))
54             (loop
55               (λ (s)
56                 (if (null? s)
57                     match-nothing
58                     (let ((match-rest
59                           (loop (cdr s)))
60                           (match-first (car s)))
61                         (λ (e failure)
62                           (match-first e
63                             (λ ()
64                               (match-rest e failure))))))))
65             (λ patterns
66               (loop patterns))))))
67
68 ;;; Example: A tag-parser for Scheme
69
70 (define *void-object* (if #f #f))

```

Nov 29, 12 16:32

## pattern-matcher-with-example.scm

Page 2/2

```

71
72 (define simple-const?
73   (let ((preds (list boolean? char? number? string?)))
74     (λ (e)
75       (ormap (λ (p?) (p? e)) preds))))
76
77 (define var? ← Symbol תואן טוקן סימול סמל סמל)
78   (λ (e)
79     (and (symbol? e)
80           (not (reserved-word? e)))))

81
82 (define reserved-word?
83   (λ (e)
84     (ormap
85       (λ (kw) (eq? e kw))
86       *reserved-words*)))

87
88 (define *reserved-words* ← הפקודות הפקודות הפקודות)
89 '(and begin cond define do else if λ
90       let let* letrec or quasiquote
91       quote set! unquote unquote-splicing))

92
93 (define parse
94   (let ((run
95         (compose-patterns
96           (pattern-rule
97             (? 'c simple-const?)
98             (λ (c) '(const ,c)))
99           (pattern-rule
100             '(quote ,(? 'c))
101             (λ (c) '(const ,c)))
102           (pattern-rule
103             (? 'v var?)
104             (λ (v) '(var ,v)))
105           (pattern-rule
106             '(if ,(? 'test) ,(? 'dit))
107             (λ (test dit)
108               '(if ,(parse test) ,(parse dit) (const ,*void-object*))))
109           (pattern-rule
110             '(if ,(? 'test) ,(? 'dit) ,(? 'dif))
111             (λ (test dit dif)
112               '(if ,(parse test) ,(parse dit) ,(parse dif))))
113             ;; add more rules here!
114           )))
115         (λ (e)
116           (run e
117             (λ ()
118               (error 'parse
119                 (format "I can't recognize this: ~s" e)))))))
120
121

```

*הפקודות*

*Symbol* *תואן* *טוקן* *סימול* *סמל* *סמל*

*-הפקודות* *הפקודות* *הפקודות*

*patterns*