

מבחן סוף סטטיסטיקה – מועד א'

טור 1

מרצה אחראי: ד"ר הילה פלאג

מתרגלים: הילה לוי, תומר כהן, אנדרייבו ביבין, תומר ביתן

: הוראות

- .1 בטופס המבחן 14 עמודים, מתוכם 5 דפי נוסחאות. בדקו שכל העמודים ברשותכם.
- .2 משך המבחן שלוש שעות (180 דקות).
- .3 כל חומר עזר חיוני אסור לשימוש.
- .4 בשאלות הפתוחות, ניתן לצוין לגבי סעיף או שאלה "לא ידוע/ת". תשובה זו תזכה ב-20% מהניקוד.
- .5 תשובה שגויה לא ייכוי בניקוד.
- .6 קראו את כל המבחן לפני שאתם מתחילה לענות על השאלות.
- .7 אין צורך להגיש את טופס מבחנים זה בתום הבחינה.
- .8 את התשובות לשאלות הסגורות יש לסמן בטופס הטורי הנפרד בלבד ואת התשובות לשאלות הפתוחות יש לכתוב בריבועים הריקים לכל שאלה בטופס התשובות הנפרד.
- .9 ודאו כי אתם מгиשים טופס תשובות וטופס טורי בלבד (את מחברת הטיווה ניתן לשמר אצלם).

בצלחה!

חלק א' - שאלות סגורות (45 נק')

שלבי קומפיילציה (27 נק')

נתונה התוכנית הבאה בשפת C :

```
1. int x = 3;
2. int y = 5;
3. int m = x - y;
4. int n = x * y;
5.
6. int k = 0;
7. while (k * k < n) {
8.     k = k + n / m;
9. }
10.
11. print("Result:");
12. printi(k);
```

בסעיפים הבאים (שאלות 1 עד 5) מוצגים שינויים (בלתי תלויים) לקוד של התוכנית. עבור כל שינוי ציינו את השלב המוקדם ביותר שבו נגלה את השגיאה :

שאלה מס' 1:

int y = -5;

(3 נק') מחליפים את שורה 2 ב-

- א. שגיאה בזמן ריצה
- ב. שגיאה בניתוח סמנטי
- ג. אין שגיאה
- ד. שגיאה בייצור קוד
- ה. שגיאה בניתוח לקסיקלי
- ו.  שגיאה בניתוח תחבירי

שאלה מס' 2:

while (k ^ 2 < n)

(3 נק') מחליפים את שורה 7 ב-

- א. שגיאה בניתוח הסמנטי
- ב. אין שגיאה
- ג. שגיאה בייצור קוד
- ד.  שגיאה בניתוח לקסיקלי
- ה. שגיאה בזמן ריצה
- ו. שגיאה בניתוח תחבירי

שאלה מס' 3:

int y = 3;

(3 נק') מחליפים את שורה 2 ב-

- א. אין שגיאה
- ב.** שגיאה בזמן ריצה
- ג. שגיאה בניתוח תחביבי
- ד. שגיאה בייצור קוד
- ה. שגיאה בניתוח לקסיקלי
- ו. שגיאה בניתוח סמנטי

שאלה מס' 4:

(3 נק') מחליפים את שורה 11 ב-

- א.** שגיאה בניתוח תחביבי
- ב. שגיאה בייצור קוד
- ג. שגיאה בניתוח סמנטי
- ד. שגיאה בזמן ריצה
- ה. אין שגיאה
- ו. שגיאה בניתוח לקסיקלי

שאלה מס' 5:

implicit cast

byte k = 0b;

(3 נק') מחליפים את שורה 6 ב-

- א. שגיאה בניתוח לקסיקלי
- ב. שגיאה בזמן ריצה
- ג. שגיאה בייצור קוד
- ד. שגיאה בניתוח תחביבי
- ה. אין שגיאה
- ו.** שגיאה בניתוח סמנטי

שינויים ב-FanC

שאלה מס' 6 :

(6 נקי) נרצה להוציאFanC postfix increment משבטFanC postfix increment כמו `++` של C, אבל הכמות בה המשתנה מוגדל מוגדרת לפי מספר הפלוסים המופיעים ברצף.
למשל:

`x++;`

יגדל את x ב-1. 1 :

`y++++;`

יגדל את y ב-3.

מהו שלב הקומpileציה המקודם ביותר שצריך לשנות על מנת לתמוך במשפט החדש?

- א. ייצור קוד
- ב. ניתוח סמנטי
- ג. ניתוח תחבירי
- ד. לא נדרש לשנות אף אחד מהשלבים
- ה.  ניתוח לקסיקלי

שאלה מס' 7 :

(6 נקי) לאחר שמיימנו את המשפט החדש, נרצה להפוך את האופרטור משבט לביטוי.
מהו שלב הקומpileציה המאוחר ביותר שלא צריך לשנות מפתרונו השאלה הקודמת?

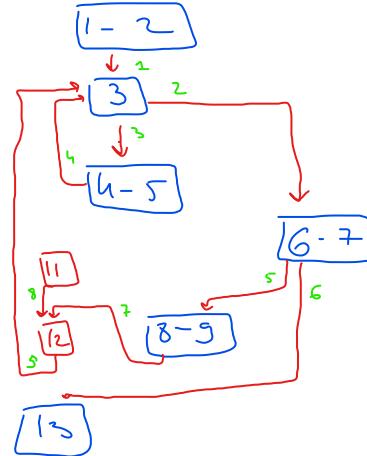
- א. לא נדרש לשנות אף אחד מהשלבים
- ב. ניתוח סמנטי
- ג.  ניתוח לקסיקלי
- ד. ניתוח תחבירי
- ה. ייצור קוד

אופטימיזציות (18 נק')

שאלה מס' 8:

נתונה התוכנית הבאה בשפת הרביעיות:

- 1. $a = 1$
- 2. $x = a$
- 3. if $x > 5$ goto 6
- 4. $x = x + 1$
- 5. goto 3
- 6. $c = 3$
- 7. if $x > 10$ goto 13
- 8. $x = x + 1$
- 9. goto ~~10~~ 12
- 10. ~~goto 12~~
- 11. $x = x - 1$
- 12. goto 3
- 13. print c



(6 נק') הריצו על המתוודה את האופטימיזציה עד להתכנסות, כמה בלוקים וכמה קשתות יש בגרף?

- 7 בלוקים, 7 קשתות
- 7 בלוקים, 8 קשתות
- 8 בלוקים, 9 קשתות
- 9 בלוקים, 9 קשתות
- 8 בלוקים, 8 קשתות

שאלה מס' 9:

(6 נק') נתונה גרסה אינטראפרט (mprsh) של C FanC. מתי משתמש בגרסה זו על פני הגרסה של C מתרגלי ביתם שלכם? ענו את התשובה הנכונה ביותר.

- עדיף להשתמש בקומפיילר אם הולכים להריץ את התוכנית יותר מפעם אחת.
- תמיד עדיף להשתמש בגרסת אינטראפרט.
- תמיד עדיף להשתמש בקומפיילר.
- עדיף להשתמש באינטראפרט אם הולכים להריץ את התוכנית יותר מפעם אחת.

שאלה מס' 10:

(6 נק') נוסף לקומפיילר אופטימיזציות ולאינטראפרט (mprsh) JIT.

נתונה תוכנית אשר אנחנו מתוכננים להריץ פעם אחת בלבד.

מתי עדיף להשתמש באינטראפרט החדש ולא בקומפיילר החדש על התוכנית הנתונה?

- הריצה אחת תמיד עדיף להשתמש בקומפיילר.
- עדיף להשתמש בגרסת האינטראפרט כשהתוכנית גודלה מאוד ות្រוץ בזמן רב.
- אף פעם אי אפשר להניח שהתוכנית תרוץ רק פעם אחת.
- הריצה אחת תמיד עדיף להשתמש בגרסת אינטראפרט.
- עדיף להשתמש באינטראפרט כשהתוכנית קטנה וכוללת קטע קוד שירוץ מספר גדול מאוד של פעמים.

חלק ב' - שאלות פתוחות (55 נק')

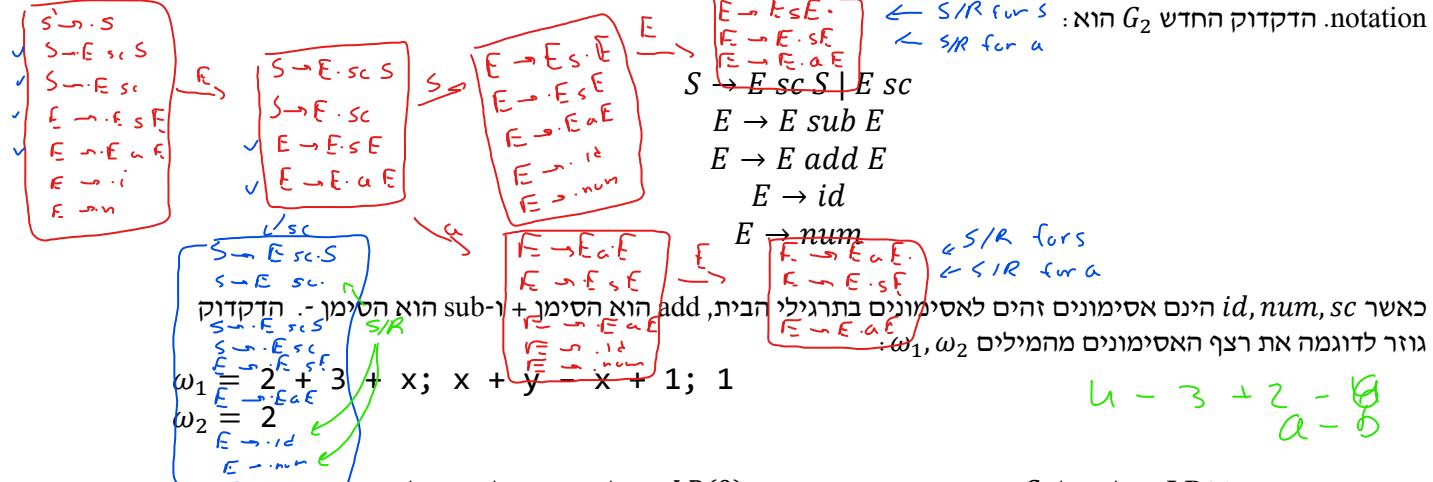
שאלה 1: דקדוקים (25 נק')

א. (8 נק') נתנו הדקדוק הבא : G_1

$$E \rightarrow lpar\ Binop\ rpar\ | id\ | num$$

$$Binop \rightarrow add\ E\ E\ | sub\ E\ E$$

ב. (8 נק') הוחלט להרחב את השפה מעט, ולשם כך גם הוחלפו הפעולות הבינאריות מ-prefix notation ל-infix notation. הדקדוק החדש G_2 הוא :



ציררו את אוטומט $LR(0)$ המלא של G_2 , והראו שהדקדוק אינו ב- $LR(0)$. יש לסמן את כל הקונפליקטים.

ג. (2 נק') ניתן לפתור את הקונפליקטים על ידי הפעלת מגנון פתרון הקונפליקטים של $bison$ על אוטומט $LR(0)$. הניחו שכרעג כל האסימונים מוגדרים $\%token$, וציינו אילו סדר ואסוציאטיביות יפתרו את הקונפליקטים – והסבירו כיצד.

ד. (7 נק') נגידר על סדרת ביטויים בשפה של הדקדוק G_2 את התוכנה "מתגלח לאיתו": רצף ביטויים מתגלח לאיתו אם בכל ביטוי יש בדיקות את כל המשתנים מהיבטי משמאלו ועד משתנה אחד חדש.

לדוגמה :

$$\begin{aligned}\omega_3 &= x + y; 1; 3 + y; \\ \omega_4 &= x; x + y; x - 2 + y - z; \\ \omega_5 &= x + y; x - y + x - z; x + y - z + w; \\ \omega_6 &= 2; 1 - x;\end{aligned}$$

ω_3 לא מתגלח לאיתו, משום שבביטוי השני לא מופיעים משתנים כלל, אך לפי ההגדירה גם x , גם y וגם משתנה נוסף היו צריכים להופיע בו. לעומת זאת, $\omega_4, \omega_5, \omega_6$ כולם מוגבלים לאיתם, משום שככל המשתנים מהיבטי הראשו חזרים על עצםם בביטוי השני בתוספת עד משתנה חדש, וכך הלאה.

כל רצף בעל ביטוי אחד בלבד (ω_2) מוגלה לאיתו באופן ריק, ללא תלות באילו משתנים מופיעים בו.

כתבו הגדירה מונחית תחביר עבור G_2 המשמשת בתוכנות **גזרות בלבץ** שתזودה כי המילה מתארת רצף ביטויים מתוגלים לאיתו.

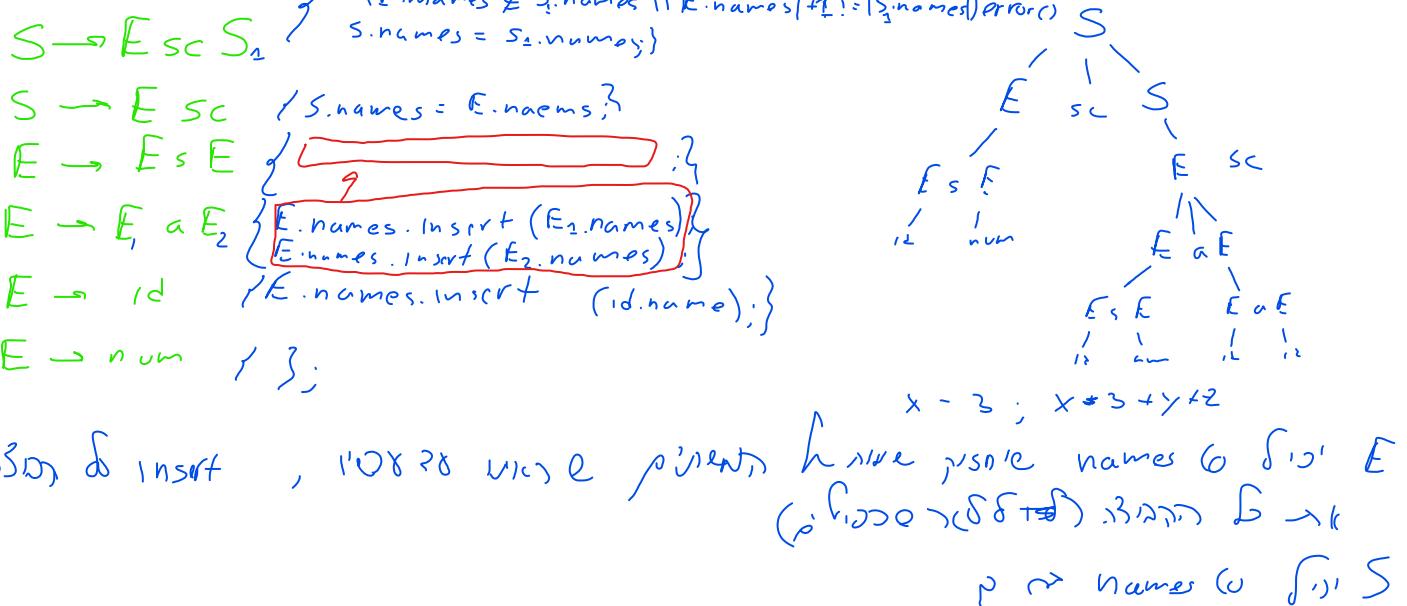
הוראות:

- מותר להשתמש בפעולות על קבוצות כחלק מהקובד שלכם על מנת לתקן.
 - אם המילה תקינה אין לבצע דבר, ואם המילה אינה תקינה יש לקרוא לפונקציה () error.
 - אין לשנות את הדקדוק! יש לכתוב פעולות סמנטיות רק במקומות המסומנים לכם.
 - אין להוציא טרמינלים תכונות סמנטיות. טרמינל id תכוונה בשם name המכילה את הלקסמה שלו.
 - טרמינל num תכוונה בשם value המכילה את הערך המספרי של הקבוע.

ציננו ב謄וף איזה תוכנות סטטיות הגדירות לכל אחד מהמשתנים **ונמקו היטב!! את משמעות התכונה.**

מלאו את מימוש הבדיקה שלכם עבור כל כלי הדקוק.

יש למלא את ההגדרות במקום המתאים לכך במחברת התשובות.



שאלה 2: אגלויזה סטטיטית (30 נק')

מוסיף לשפת הריביעיות את הפקודה call המבצעת קריאה לפונקציה. על מנת לאפשר העברת פרמטרים וערך החזרה, נוספות גם שתי הפקודות:

- x param דוחפת את הערך של x לממחסנית כפרמטר
- x retval קוראת את ערך החזרה של הפונקציה מהמחסנית לתוך x

נתון הקוד הבא לפונקציה `me_analyze` :
שים לו לב כי x* היא פניה זיירון בכתובות x, כפי שראיתם בהרצאה.

```

1. analyze_me: //params: b, r
2.   if r == 8 goto end
3.   i = 0
4.   loop:
5.     if i >= 8 goto loop_end
6.     param b
7.     param r
8.     call isSafe
9.     retval t1
10.    if t1 == 0 goto loop_post
11.    t1 = r * 4
12.    t1 = t1 + b
13.    *t1 = i //write to memory address
14.    t2 = t1 + 1
15.    param b
16.    param t2
17.    call analyze_me
18.    *t1 = -1 //write to memory address
19.    loop_post:
20.      i = i + 1
21.      goto loop
22.    loop_end:
23.  end:
24.  nop

```

א. (5 נק') כאשר תרוויחו את ייצור קוד האסמבלי משפת הריביעיות לטפל בפקודה call, איזה טיפול עליהם להוסיפה עבור רегистרים? הסבירו:

1. היכן בקוד יידרש טיפול? הסבירו במלואות ווגם צייניא את מספרי השורות בפונקציה `analyze_me` בהן הטיפול ייתושך. אומנם מטרת התרגיל היא לתרגם את הקוד לאסמבלי, אך מטרת התרגיל היא לתרגם את הקוד לאסמבלי.
2. מדוע הטיפול נדרש?

אחרי הרצת אלגוריתם register allocation, התקבלו ההקצאות הבאות :

מספרה	רегистר
r5	b
r8	i
r9	r
r17	t1
r24	t2

בטיוד ה-ABI שהקומPILEר שלכם משתמש בו נתונה لكم החלוקה הבאה בין הרגיסטרים:

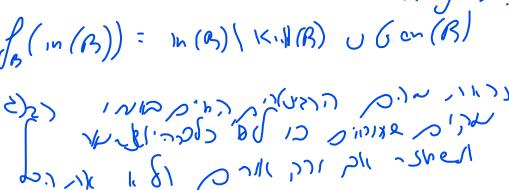
Caller-save	Callee-save
r8-r15	r1-r7
r24-r25	r16-r23

ב. (4 נק') מהו הטיפול הטוב ביותר ברגיסטרים שניתן לבצע לא כל מידע נוסף? הסבירו בקצרה את החיסרון של הטיפול שהצעתם.

ג. (6 נק') הגדרו אנליזה שתאפשר לכם לשפר את הטיפול. ניתן להשתמש באנליזה שלמדתם בכיתה אך יהיה צריך להרחיב אותה לפחות החדשות, או להגדיר מחדש.

1. ציינו את הדומין של האנליזה, את יחס הסדר ופעולות ה-move, ואת פונקציית המעברים.
2. הסבירו כיצד תשתמשו בתוצאת האנליזה כדי לשפר את הטיפול.

ד. (10 נק') ציירו את ה-CFG של הפונקציה והריצו את האנליזה. ✓



ה. (5 נק') לפי תוצאות האנליזה שלכם, הסבירו כיצד טיפול הקומPILEר ברגיסטרים בנקודות שהדרטם. ✓

נוסחאות ו알גוריתמים

כל ההגדרות מתייחסות לדקוק $G = (V, T, P, S)$

Top Down

$$\text{first}(\alpha) = \{ t \in T \mid \alpha \Rightarrow^* t\beta \wedge \beta \in (V \cup T)^* \}$$

$$\text{follow}(A) = \{ t \in T \cup \{\$\} \mid S\$ \Rightarrow^* \alpha A t \beta \wedge \alpha \in (V \cup T)^* \wedge \beta \in (V \cup T)^*(\epsilon | \$) \}$$

$$\text{select}(A \rightarrow \alpha) = \begin{cases} \text{first}(\alpha) \cup \text{follow}(A) & \alpha \Rightarrow^* \epsilon \\ \text{first}(\alpha) & \text{otherwise} \end{cases}$$

הגדרה: דקוק G הוא LL(1) אם ורק אם לכל שני כלליים ב- G השווים לאותו משתנה A מתקיים:
 $\text{select}(A \rightarrow \alpha) \cap \text{select}(A \rightarrow \beta) = \emptyset$

הגדרת טבלת המעברים $M : V \times (T \cup \{\$\}) \rightarrow P \cup \{\text{error}\}$: LL(1) עבור דקוק

$$M[A, t] = \begin{cases} A \rightarrow \alpha & t \in \text{select}(A \rightarrow \alpha) \\ \text{error} & t \notin \text{select}(A \rightarrow \alpha) \text{ for all } A \rightarrow \alpha \in P \end{cases}$$

אלגוריתם מנתח :LL(1)

```

Q.push(S)
while !Q.empty() do
    X = Q.pop()
    t = next token
    if X ∈ T then
        if X = t then MATCH
        else ERROR
    else // X ∈ V
        if M[X , t] = error then ERROR
        else PREDICT(X , t)
    end if
end while
t = next token
if t = $ then ACCEPT
else ERROR

```

Bottom Up

פריט LR(0) הוא $A \rightarrow \alpha\beta \in P$ כאשר $(A \rightarrow \alpha\bullet\beta)$ מוגדר באוֹפָן אינדוקטיבי:
סגור closure על קבוצת פריטים I מוגדר באוֹפָן אינדוקטיבי:
 $\text{closure}(I) = I$ בסיס.

צעד: אם $(B \rightarrow \bullet\gamma) \in \text{closure}(I)$, אז לכל $P, \gamma \in I$, גם $(A \rightarrow \alpha\bullet B\beta) \in \text{closure}(I)$.

פונקציית המעברים של האוטומט:

$$\delta(I, X) = \bigcup \left\{ \text{closure}(A \rightarrow \alpha X \bullet \beta) \mid (A \rightarrow \alpha \bullet X \beta) \in I \right\}$$

פריט LR(1) הוא $t \in T \cup \{\$\}$, $A \rightarrow \alpha\beta \in P$ כאשר $(A \rightarrow \alpha\bullet\beta)$ מוגדר באוֹפָן אינדוקטיבי:
סגור closure על קבוצת פריטים I מוגדר באוֹפָן אינדוקטיבי:
 $\text{closure}(I) = I$ בסיס.

צעד: אם $x \in \text{first}(\beta)$, אז לכל $P, \gamma \in I$, גם $(A \rightarrow \alpha\bullet B\beta, t) \in \text{closure}(I)$ ולכל $P, \gamma \in I$, גם $(B \rightarrow \bullet\gamma, x) \in \text{closure}(I)$.

פונקציית המעברים של האוטומט:

$$\delta(I, X) = \bigcup \left\{ \text{closure}(A \rightarrow \alpha X \bullet \beta, t) \mid (A \rightarrow \alpha \bullet X \beta, t) \in I \right\}$$

הגדרת טבלת action למנת SLR :

$$\text{action}[i, t] = \begin{cases} \text{SHIFT}_j & \delta(I_i, t) = I_j \\ \text{REDUCE}_k & \text{rule } k \text{ is } A \rightarrow \alpha, (A \rightarrow \alpha\bullet) \in I_i \text{ and } t \in \text{follow}(A) \\ \text{ACCEPT} & (S' \rightarrow S\bullet) \in I_i \text{ and } t = \$ \\ \text{ERROR} & \text{otherwise} \end{cases}$$

הגדרת טבלת action למנת $\text{LR}(1)$:

$$\text{action}[i, t] = \begin{cases} \text{SHIFT}_j & \delta(I_i, t) = I_j \\ \text{REDUCE}_k & \text{rule } k \text{ is } A \rightarrow \alpha \text{ and } (A \rightarrow \alpha\bullet, t) \in I_i \\ \text{ACCEPT} & (S' \rightarrow S\bullet, \$) \in I_i \text{ and } t = \$ \\ \text{ERROR} & \text{otherwise} \end{cases}$$

הגדרת טבלת goto ו-SLR למנת $\text{LR}(1)$:

$$\text{goto}[i, X] = \begin{cases} j & \delta(I_i, X) = I_j \\ \text{error} & \text{otherwise} \end{cases}$$

אלגוריתם מוגן :shift/reduce

```

Q.push(0)           // where 0 is the initial state of the prefix automaton
while true do
    k = Q.top().state
    t = next token
    do action[k , t]
end while

```

קוד בינויים

```

x := y op z
x := op y
x := y
goto L
if x relop y goto L
print x

```

- סוגי פקודות בשפת הבינויים:
1. משפטי השמה עם פעולה ביןארית
 2. משפטי השמה עם פעולה אונריה
 3. משפטי העתקה
 4. קופיצה בלתי מותנה
 5. קופיצה מותנה
 6. הדפסה

Data-Flow Analysis

הגדרות מתיחסות ל-CFG $:G = (V, E)$ מהצורה

הצורה הכללית של המשוואות בחישוב סריקה קדמית:

$$\begin{aligned} \text{in}(B) &= \bigcup_{(S,B) \in E} \text{out}(S) \\ \text{out}(B) &= f_B(\text{in}(B)) \end{aligned}$$

הצורה הכללית של המשוואות בחישוב סריקה אחורי:

$$\begin{aligned} \text{in}(B) &= \bigcup_{(B,D) \in E} \text{out}(D) \\ \text{out}(B) &= f_B(\text{in}(B)) \end{aligned}$$

FanC שפת

איסימונים:

מבנה	איסימון
int	INT
byte	BYTE
b	B
bool	BOOL
and	AND
or	OR
not	NOT
true	TRUE
false	FALSE
return	RETURN
if	IF
else	ELSE
while	WHILE
break	BREAK
continue	CONTINUE
;	SC
(LPAREN
)	RPAREN
{	LBRACE
}	RBRACE
=	ASSIGN
== != < > <= >=	RELOP
+ - * /	BINOP
[a-zA-Z][a-zA-Z0-9]*	ID
0 [1-9][0-9]*	NUM
"([^\n\r\"\\] \\[rnt\"\\\"])+"	STRING

דקדוק:

1. $\text{Program} \rightarrow \text{Statements}$
2. $\text{Statements} \rightarrow \text{Statement}$
3. $\text{Statements} \rightarrow \text{Statements Statement}$
4. $\text{Statement} \rightarrow \text{LBRACE Statements RBRACE}$
5. $\text{Statement} \rightarrow \text{Type ID SC}$
6. $\text{Statement} \rightarrow \text{Type ID ASSIGN Exp SC}$
7. $\text{Statement} \rightarrow \text{ID ASSIGN Exp SC}$
8. $\text{Statement} \rightarrow \text{Call SC}$
9. $\text{Statement} \rightarrow \text{RETURN SC}$
10. $\text{Statement} \rightarrow \text{IF LPAREN Exp RPAREN Statement}$
11. $\text{Statement} \rightarrow \text{IF LPAREN Exp RPAREN Statement ELSE Statement}$
12. $\text{Statement} \rightarrow \text{WHILE LPAREN Exp RPAREN Statement}$
13. $\text{Statement} \rightarrow \text{BREAK SC}$
14. $\text{Statement} \rightarrow \text{CONTINUE SC}$
15. $\text{Call} \rightarrow \text{ID LPAREN Exp RPAREN}$
16. $\text{Type} \rightarrow \text{INT}$
17. $\text{Type} \rightarrow \text{BYTE}$
18. $\text{Type} \rightarrow \text{BOOL}$
19. $\text{Exp} \rightarrow \text{LPAREN Exp RPAREN}$
20. $\text{Exp} \rightarrow \text{Exp BINOP Exp}$
21. $\text{Exp} \rightarrow \text{ID}$
22. $\text{Exp} \rightarrow \text{Call}$
23. $\text{Exp} \rightarrow \text{NUM}$
24. $\text{Exp} \rightarrow \text{NUM B}$
25. $\text{Exp} \rightarrow \text{STRING}$
26. $\text{Exp} \rightarrow \text{TRUE}$
27. $\text{Exp} \rightarrow \text{FALSE}$
28. $\text{Exp} \rightarrow \text{NOT Exp}$
29. $\text{Exp} \rightarrow \text{Exp AND Exp}$
30. $\text{Exp} \rightarrow \text{Exp OR Exp}$
31. $\text{Exp} \rightarrow \text{Exp RELOP Exp}$
32. $\text{Exp} \rightarrow \text{LPAREN Type RPAREN Exp}$