

# Compiler Construction

## Winter 2020

### Recitation 13: Exam Rehearsal

Yotam Feldman

Based on slides by Guy Golan-Gueta

## Question – MiniJava Extension

רוצים להוסיף לשפת MiniJava פעולה שמקצה אובייקטים על המחסנית (במקום ב-heap). תארו את ההשפעה על כל אחד משלבי הקומפילציה של התוספת לשפה של ביטוי (expression) שמקצה אובייקט על המחסנית, למשל כמו בקטע הקוד הבא:

```
class A { ... }
```

```
...
```

```
A x = newS A();
```

```
...
```

# Question – MiniJava Extension

## Lexical analyzer

- Add a new token “NEWS”
- "newS" { return token(sym.NEWS); }

## Parser

terminal NEWS;

...

Expr ::= NEWS ID:name LPAREN RPAREN

{: RESULT = new NewSObjectExpr(name); :}

...

# Question – MiniJava Extension

## Semantic analysis

- Scope analysis: “name” is refers to a defined class
- Type analysis: newS expression has the reference type of name (same as “new”)
- Stack object never escapes

- Can't be returned

```
A foo() {  
    A a = newS A();  
    ....  
    return a;  
}
```

- Can't be pointed to from a heap-allocated object or array
- What if it's passed as a parameter?

# Question – MiniJava Extension

## Intermediate representation (LLVM)

- Translate `newS` exactly as `new` while using stack memory
- Use `alloca` to allocate an array of bytes on the stack
  - `%_0 = alloca [nBytes x i8]`  
where `nBytes` is the number of bytes the compiler calculates for the binary representation of the object (vtable pointer + fields)
- Initialize the array to zeros
  - `store` in a loop, or generate individual instructions per byte
- Continue the translation as usual (assigning the vtable pointer)

# Question – MiniJava Extension

Assembly generation

## שאלה (2010 א')

נתונה רשימת מאורעות. לכל אחד מהמאורעות צייני בקצרה (מספיקות 1-3 שורות הסבר לכל פריט):

האם הוא מתרחש בזמן ריצה, בזמן קומפילציה, או בזמן בניית הקומפיילר; אם בחרת בזמן קומפילציה, מהו השלב המסוים בקומפיילר בו מתרחש המאורע;

במידה ויש מספר תשובות נכונות, הסבירי את כולן.

• בשפת C כתובת משתנה ה external נקבע ל 1500.

• למשתנה x נקבע 40 offset ברשומת ההפעלה.

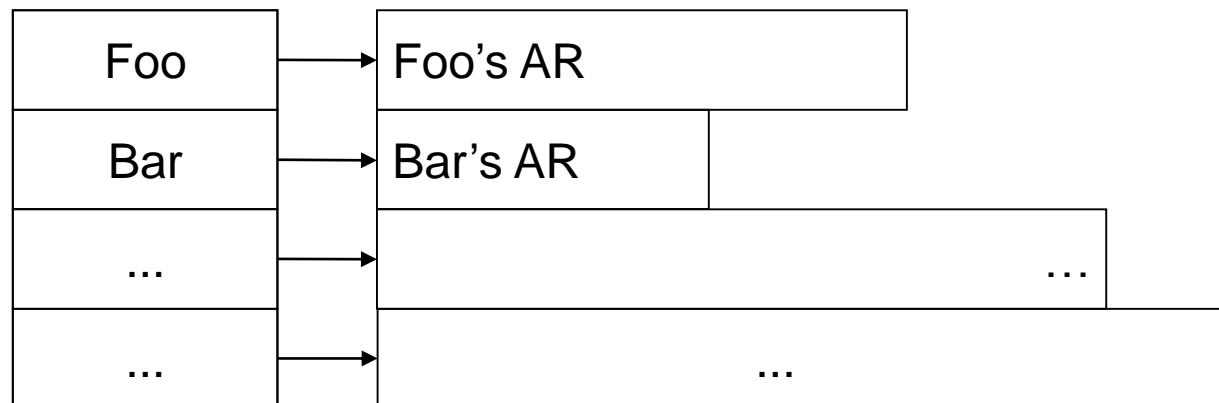
• בכל המסלולים המגיעים לנקודה מסוימת בקוד מובטח שערכו של משתנה p אינו NULL.

• משתנה x הוא frame-resident-variable.

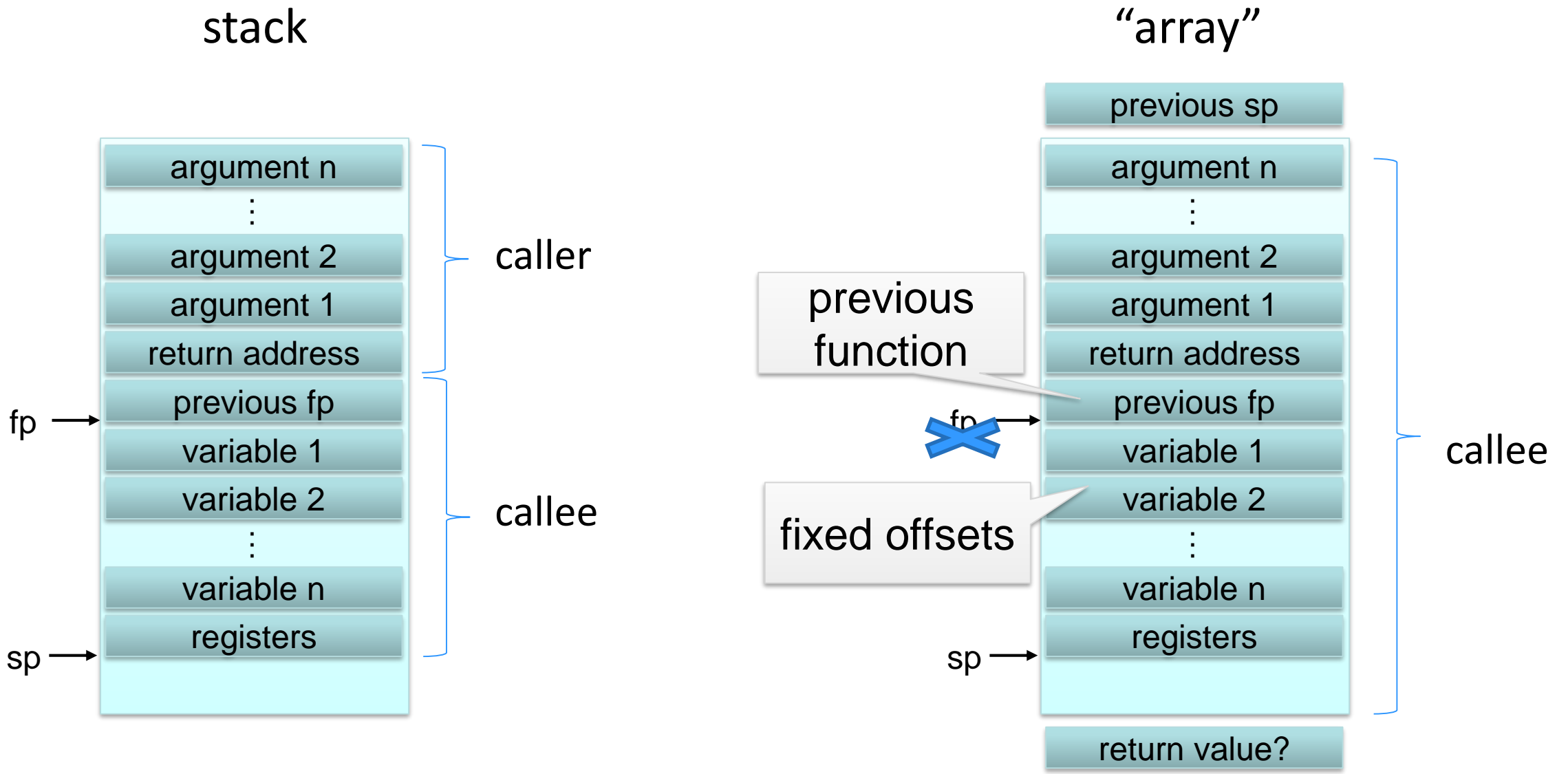
## שאלה (2008 א')

אנו מעוניינים כי רשומות ההפעלה ינוהלו במערך ולא במחסנית כמתואר  
בציור הבא. במערך זה קיימת כניסה יחידה לכל פונקציה בתכנית. בכל כניסה  
במערך ניתן לנהל רשומת הפעלה אחת בלבד. שים לב שגודלה של רשומת  
הפעלה עלול להשתנות במהלך ריצת התוכנית. הנח כי כל קוד התכנית זמין  
בקובץ יחיד.

א. (12 נק') הצע דרך למימוש משטר מערך מעין זה. לאילו מבני נתונים  
נוספים תזדקק. התייחס לכל רכיבי רשומת ההפעלה – מה ישתנה במימוש  
עבורם ומה יותר ללא שינוי בהשוואה למימוש משטר מחסנית.



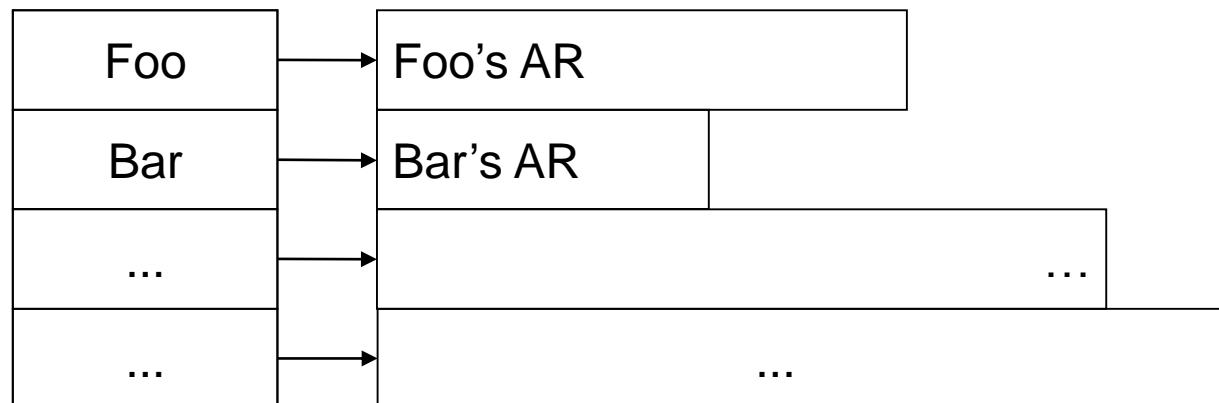




## שאלה (2008 א')

אנו מעוניינים כי רשומות ההפעלה ינוהלו במערך ולא במחסנית כמתואר  
בציור הבא. במערך זה קיימת כניסה יחידה לכל פונקציה בתכנית. בכל כניסה  
במערך ניתן לנהל רשומת הפעלה אחת בלבד. שים לב שגודלה של רשומת  
הפעלה עלול להשתנות במהלך ריצת התוכנית. הנח כי כל קוד התכנית זמין  
בקובץ יחיד.

- א. (12 נק') הצע דרך למימוש משטר מערך מעין זה. לאילו מבני נתונים  
נוספים תזדקק. התייחס לכל רכיבי רשומת ההפעלה – מה ישתנה במימוש  
עבורם ומה יותר ללא שינוי בהשוואה למימוש משטר מחסנית.
- ב. (3 נק') לאילו שינויים תזדקק אם קוד התכנית מחולק למספר קבצים.
- ג. (5 נק') האם ניתן להריץ כל תכנית במשטר מערך. הבחן בין 3 מקרים  
אפשריים.



## שאלה (2010 ב')

א. (14 נק') שפת IC תומכת בהערות שמתחילות בזוג התווים /\* ומסתיימות בזוג התווים /\*.

הנך מתבקשת להוסיף לשפת IC תמיכה בקינון של הערות אלו. משמעות הקינון היא שכאשר נתקלים ב /\*, מתייחסים לתווים העוקבים כהערה (כולל תווי newline) עד אשר מספר המופעים של /\* שווה למספר המופעים של /\*.

להלן דוגמא להערה שמקוננות בה שתי הערות נוספות:

**/\* /\*1\*/\*2\*/\***

הסבירי במילים כיצד היית משנה את שלב ה lexical analysis, על מנת להוסיף תמיכה זו.

# NestedComments with Start States

```
static int nestedCount =0;
%state comment
%%
<YYINITIAL>\^*      {nestedCount++;
                      YYBEGIN(comment);}
                      ;
<comment>\*/        { if (--nestedCount==0) {
                      YYBEGIN(YYINITIAL);
                      }
                      }
<comment>.[[ \t\n]  ;
```

## המשך:

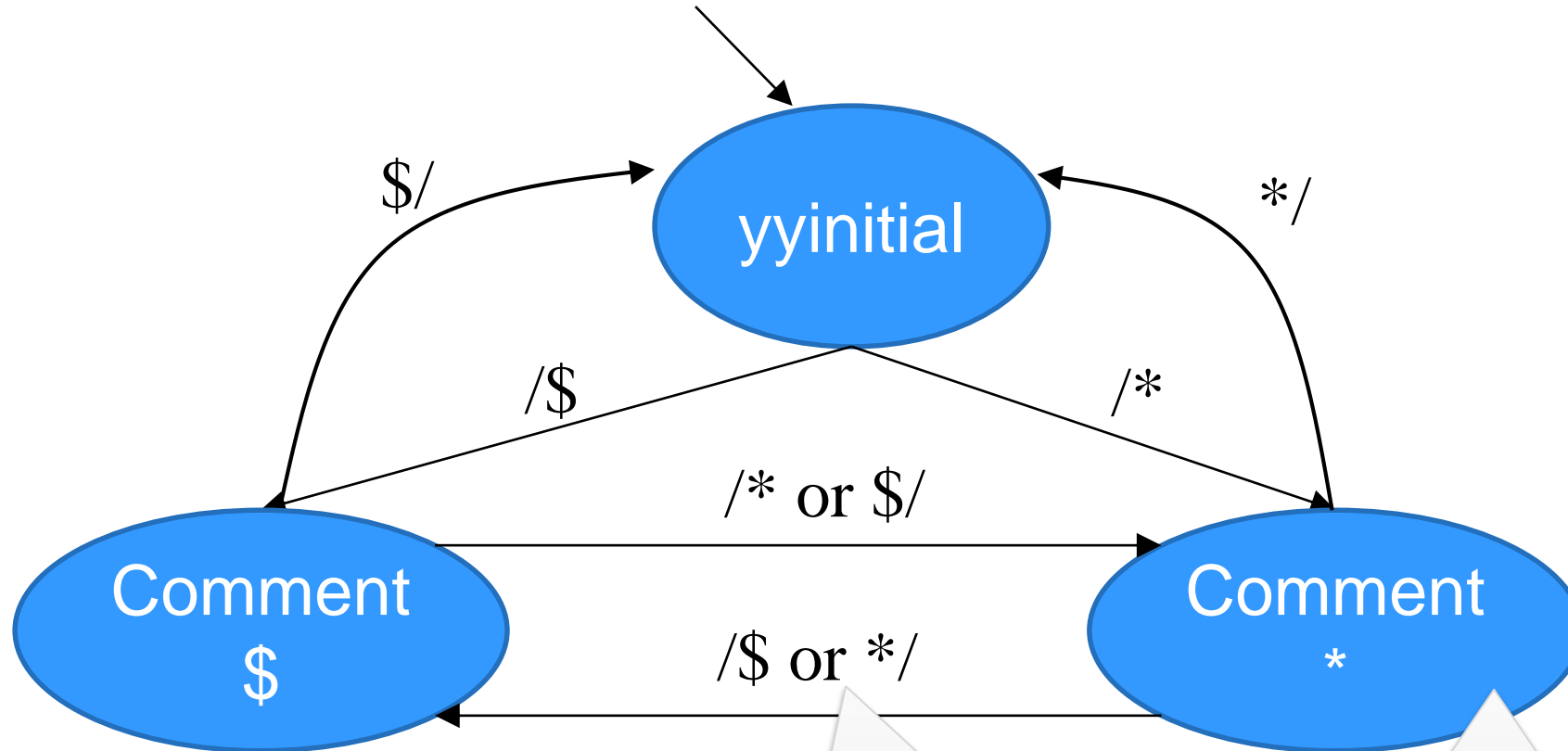
ב. (14 נק') נניח שרוצים להוסיף סוג דומה של הערות שמתחילות ב  $\$/$  ומסתיימות ב  $\$/$  (בדומה ל  $/*$  ו  $/*$ ). ורוצים לאפשר קינון בין שני הסוגים של ההערות.  
נאמר שהערה היא חוקית כאשר יש התאמה בין הסוגרים שהיא כוללת.

להלן שתי דוגמאות להערות חוקיות:  $\$/ /* /* \$/ \$/ /* /* \$/$   
 $/* /*1*/ /*2\$/ /*$

להלן שתי דוגמאות להערות לא חוקיות:  $/* 1 \$/$   
 $/* \$/ /* \$/$

הסבירי במילים כיצד היית משנה את שלב ה  $\text{lexical analysis}$ , על מנת להוסיף תמיכה כזו. הקומפיילר צריך להודיע על שגיאה במקרה וקיימת הערה לא חוקית.

```
int nestedStarCount;  
int nestedDollarCount;
```



If other counter > 0  
(or, split to more states)

Throw exception on \$/

שאלה 6 (2010 א')

נתון הדקדוק הבא

**Goal** → **ClockNoise** \$  
**ClockNoise** → **ClockNoise** *tick tock*  
**ClockNoise** → *tick tock*

ב. (7 נק') האם הדקדוק ניתן לזיהוי בשיטת Top-Down ? אם לא, כתבי דקדוק שמקבל אותה שפה וניתן לזיהוי בשיטת Top-Down.

ג. (7 נק') כתבי את הפונקציות הרקורסיביות המתאימות לזיהוי שפת הדקדוק בשיטת Top-Down.

# Example Predictive Parser

```
<S> → id := <E>  
<S> → if (<E>) <S> else <S>  
<E> → <T> <EP>  
<T> → id | (<E>)  
<EP> → ε | + <E>
```

```
<S> → <S> ; <S>
```



```
def parse_S():  
    if id(input):  
        match(input, id)  
        match(input, assign)  
        parse_E()  
    elif if_tok(input):  
        match(input, if_tok)  
        match(input, lp)  
        parse_E()  
        match(input, rp)  
        parse_S()  
        match(input, else_tok)  
        parse_S()  
    else:  
        syntax_error()
```

```
def parse_E():  
    parse_T()  
    parse_EP()  
  
def parse_T():  
    if id(input):  
        match(input, id)  
    elif lp(input):  
        match(input, lp)  
        parse_E()  
        match(input, rp)  
    else:  
        syntax_error()
```

```
def parse_EP():  
    if plus(input):  
        match(input, plus)  
        parse_E()  
    elif  
        rp(input) or  
        else_tok(input) or  
        eof(input):  
        return // ε  
    else:  
        syntax_error()
```

```
def main:  
    parse_S;  
    if not match(input,  
EOF) ...
```



**Goal** → **ClockNoise** \$  
**ClockNoise** → *tick tock* **Rest**  
**Rest** →  $\epsilon$  / *tick tock* **ClockNoise**

```
def parse_goal():
    parse_clock_noise()
    match(input, dollar)

def parse_clock_noise():
    match(input, tick)
    match(input, tock)
    parse_rest()

def parse_rest():
    if tick(input):
        match(input, tick)
        match(input, tock)
        parse_clock_noise()
    if dollar(input):
        pass
else:
    syntax_error()
```

שאלה 6 (2010 א')

נתון הדקדוק הבא

**Goal** → **ClockNoise** \$  
**ClockNoise** → **ClockNoise** *tick tock*  
**ClockNoise** → *tick tock*

א. (7 נק') האם הדקדוק הוא LR(0)? הצדיקי את תשובתך.

## שאלה (2007 ב')

היזכרי באלגוריתם הקצאת האוגרים הלוקאלי של Sethi-Ullman לעצי ביטויים, המבוסס על תכנות דינאמי, ועני על השאלות הבאות. נמקי את הפיתרונות בצורה מדויקת ככל האפשר, ע"י התייחסות לאלגוריתם.

- א. (5 נק') מהו ה-worst case של האלגוריתם? כלומר, מבחינה אסימפטוטית, מהו היחס הגבוה ביותר בין מספר האוגרים לגודל עץ הביטוי?  
רמז: מהו היחס בין מספר האוגרים לגובה עץ הביטוי?
- ב. (5 נק') מהו ה-best case של האלגוריתם?
- ג. (5 נק') מתי היחס בין מספר האוגרים הניתן ע"י הקצאת אוגרים נאיבית (במעבר מלמעלה למטה, מצד שמאל לצד ימין) לזה הניתן ע"י האלגוריתם הוא הגבוה ביותר?

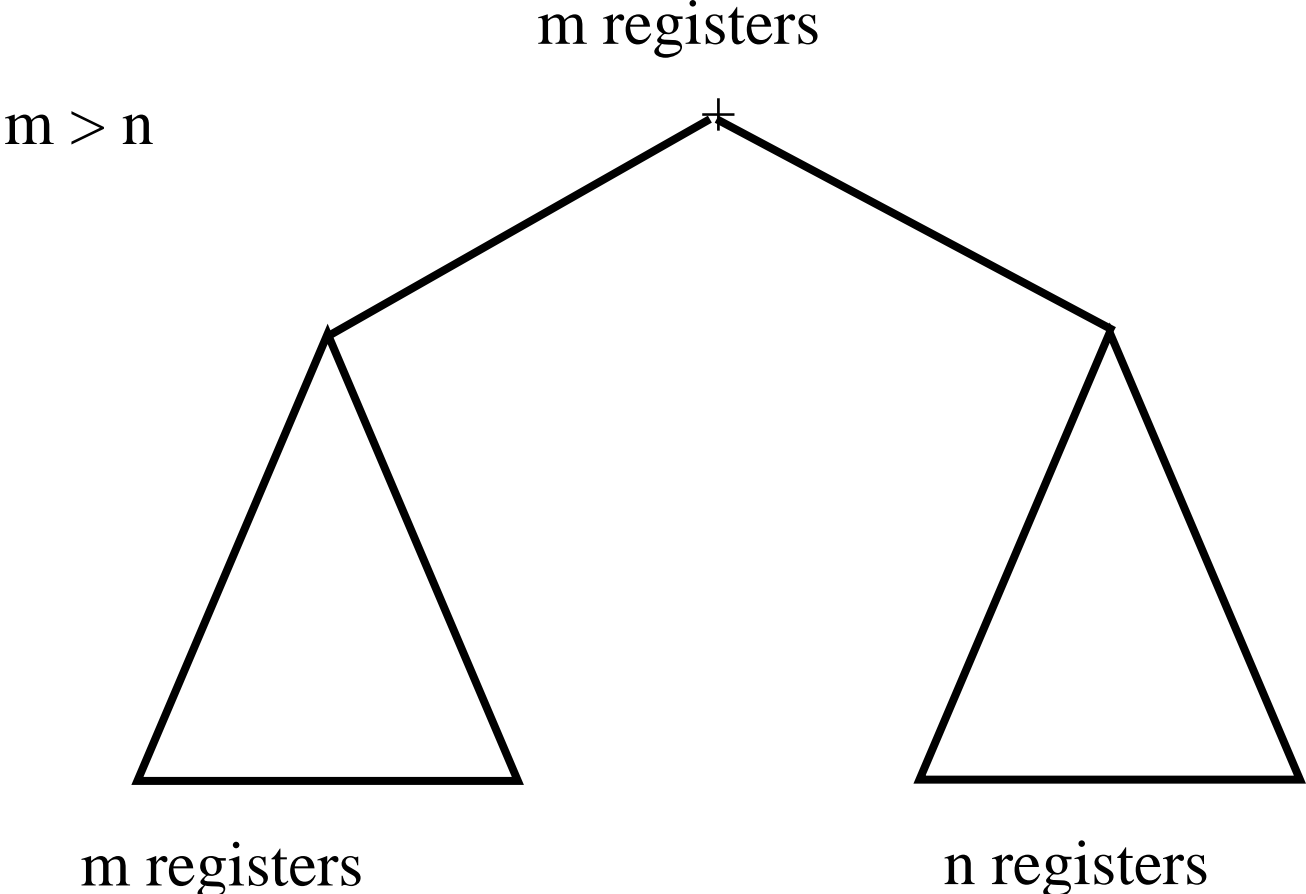
# Two Phase Solution

## Dynamic Programming

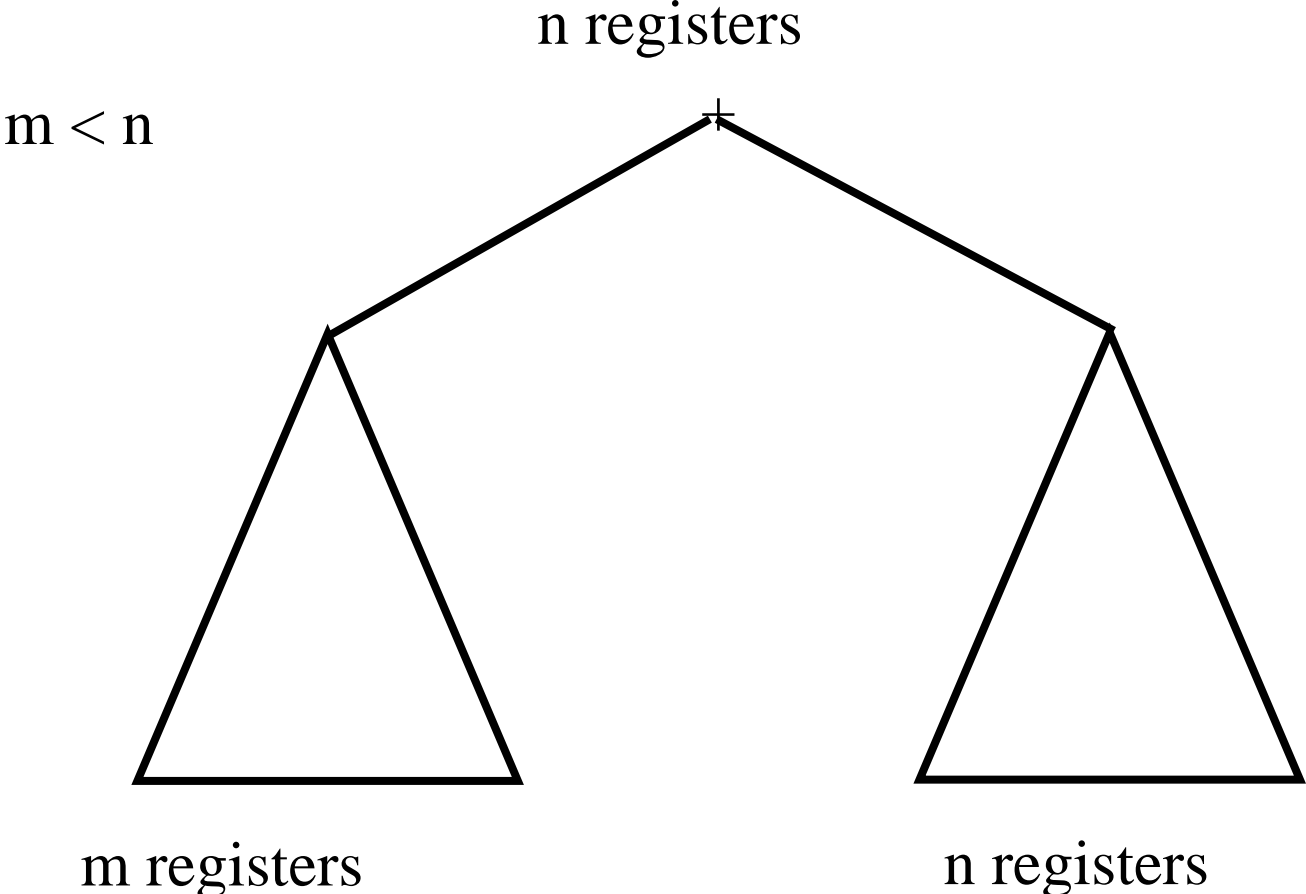
### Sethi & Ullman

- Bottom-up (labeling)
  - Compute for every subtree
    - The minimal number of registers needed
    - Weight
- Top-Down
  - Generate the code using labeling by preferring “heavier” subtrees (larger labeling)
  - Can integrate spilling

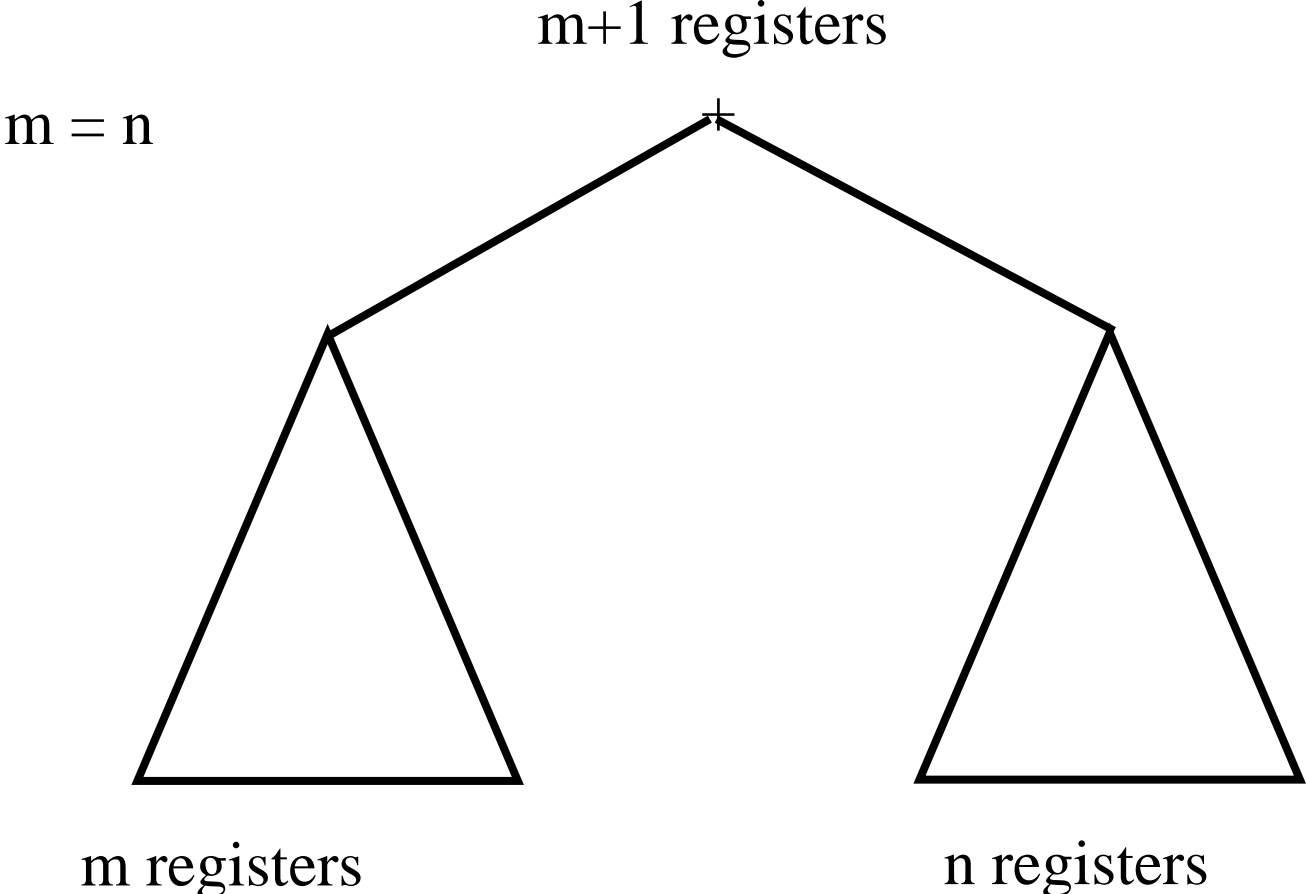
# The Labeling Principle



# The Labeling Principle



# The Labeling Principle



# The Labeling Algorithm

```
weight(Node: expression): integer {  
  switch node: {  
    case number(n: integer): return 1;  
    case localVariable(v: symbol) return 1;  
    case e1: Node + e2: Node {  
      let lw: integer = weight(e1);  
      let rw: integer = weight(e2);  
      if (lw < rw) return rw ;  
      else if (lw > rw) return lw;  
      else return lw + 1 ;  
    }  
  }  
  ...  
}
```



## שאלה (2007 ב')

היזכרי באלגוריתם הקצאת האוגרים הלוקאלי של Sethi-Ullman לעצי ביטויים, המבוסס על תכנות דינאמי, ועני על השאלות הבאות. נמקי את הפיתרונות בצורה מדויקת ככל האפשר, ע"י התייחסות לאלגוריתם.

- א. (5 נק') מהו ה-worst case של האלגוריתם? כלומר, מבחינה אסימפטוטית, מהו היחס הגבוה ביותר בין מספר האוגרים לגודל עץ הביטוי?  
רמז: מהו היחס בין מספר האוגרים לגובה עץ הביטוי?
- ב. (5 נק') מהו ה-best case של האלגוריתם?
- ג. (5 נק') מתי היחס בין מספר האוגרים הניתן ע"י הקצאת אוגרים נאיבית (במעבר מלמעלה למטה, מצד שמאל לצד ימין) לזה הניתן ע"י האלגוריתם הוא הגבוה ביותר?

שאלה (2007 ב')

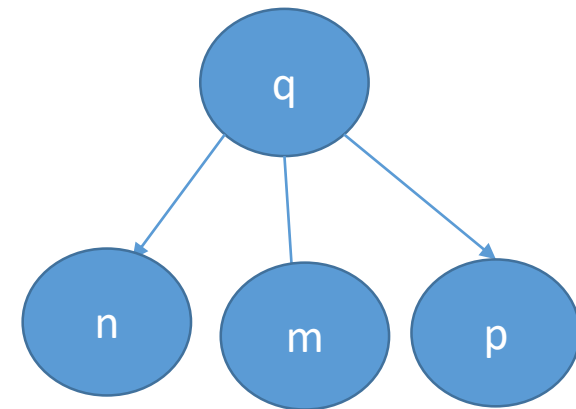
**היזכרי באלגוריתם הקצאת האוגרים הגלובאלי ע"י צביעת גרף ההפרעות.**  
ד. (5 נק') בהינתן  $k$  אוגרים, ציירי סכמטית (או תארי) את הגרף שבו יהיה  
actual spill, כלומר מספר האוגרים הנדרש יהיה גדול מ- $k$ .

# Iteratively Computing Liveness

- Construct a control flow graph of instructions
  - Every instruction **uses** a set of variables and **defines** a set of variables
    - example  $x = y+z$ 
      - use( $\{y, z\}$ )
      - def( $\{x\}$ )

$$\text{liveOut}(q) = \text{liveIn}(n) \cup \text{liveIn}(m) \cup \text{liveIn}(p)$$

- Liveness Equations
  - $\text{liveOut}(\text{exit}) = \{\}$
  - $\text{liveIn}(n) = (\text{liveOut}(n) - \text{def}(n)) \cup \text{use}(n)$
  - $\text{liveOut}(n) = \bigcup_{m: \text{succ}(n, m)} \text{liveIn}(m)$
- Computed iteratively from the exit node



# Constructing interference graphs

- Compute liveness information at every instruction
- Variables 'a' and 'b' **interfere** when there exists an instruction  $n: a \leftarrow \text{exp}$  and  $'b' \in \text{LiveOut}[n]$

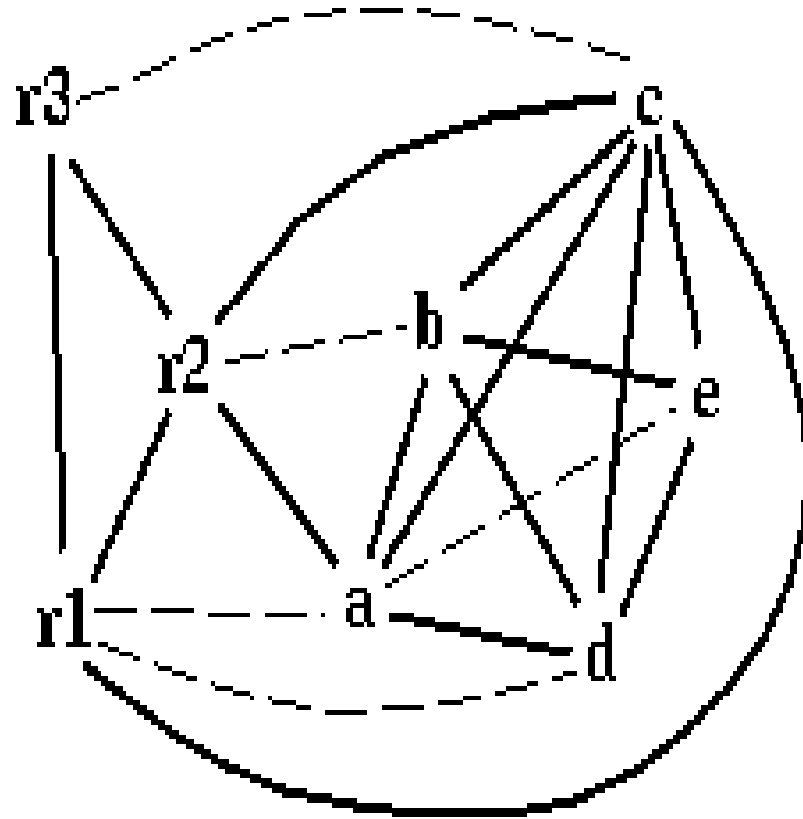
```

enter          /* r2, r1, r3 */
c := r3 /* c, r2, r1 */
a := r1 /* a, c, r2 */
b := r2 /* a, c, b */
d := 0 /* a, c, b, d */
e := a /* e, c, b, d */

loop:
d := d+b /* e, c, b, d */
e := e-1 /* e, c, b, d */
if e>0 goto loop /* c, d */
r1 := d /* r1, c */
r3 := c /* r1, r3 */

return /* r1, r3 */

```



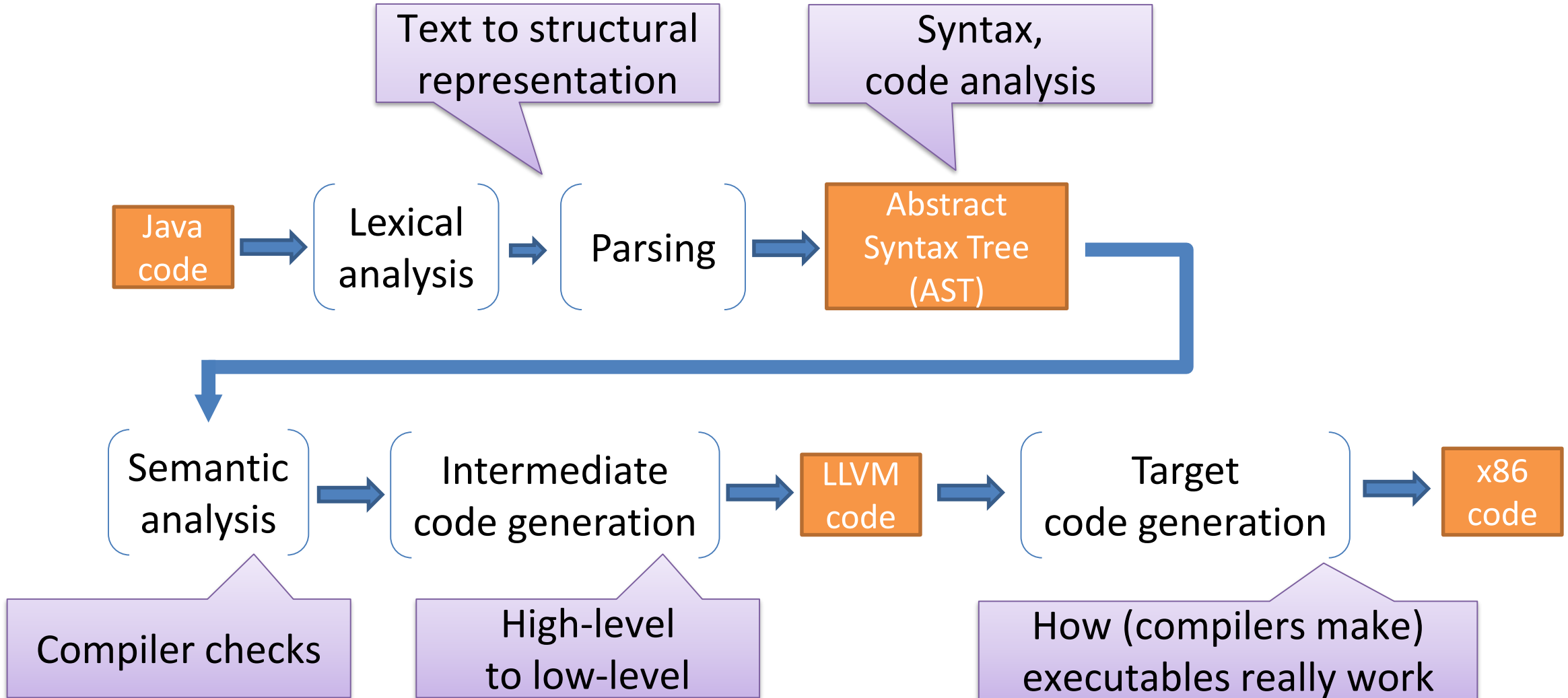
שאלה (2007 ב')

**היזכרי באלגוריתם הקצאת האוגרים הגלובאלי ע"י צביעת גרף ההפרעות.**  
ד. (5 נק') בהינתן  $k$  אוגרים, ציירי סכמטית (או תארי) את הגרף שבו יהיה  
actual spill, כלומר מספר האוגרים הנדרש יהיה גדול מ- $k$ .

## שאלה (2007 ב')

- צייני לכל פריט ברשימה הבאה האם הוא חלק מרשומת הפעלה או לא, ובאילו מקרים (מספיקה שורת הסבר לכל פריט).
- כתובות הפונקציות הוירטואליות של מחלקה A.
  - (ערכים של) ה- frame pointer.
  - המחרוזת "hello world".
  - הפקודה `mov eax, ebx`.

# Compilation Phases





# Compilation Phases

