עקרונות שפות תכנות:

תרגיל 2:

תאריך הגשה: 6.2.24

הוראות הגשה: ההגשה בזוגות / כיחידים דרך מערכת הסאבמיט. כל זוג נדרש לחשוב, לפתור ולכתוב את התרגיל בעצמו. יש לקרוא הוראות אלא בקפידה. הגשה שלא על פי הוראות אלה תוביל להורדת ניקוד.

קבצים להגשה:

- ex2.pdf
- ast.ml
- semantics.ml

קובץ עם השם משתמש בסבמיט ות.ז של כל אחד מהמגישים באופן הבא:

id.txt:

```
301111111 NOFRI
209111111 YOSIM
```

כל הקבצים צריכים להיות בקובץ zip אחד בשם: ex2.zip

חלק א': הוכחות בסמנטיקה:

שאלה 1:

א. הוכיחו את השקילות הסמנטית הבאה בסמנטיקה טבעית:

- $(S1; S2); S3 \sim S1; (S2; S3)$

ב. הוכיחו כי במקרה הכללי לא מתקיימת השקילות הבאה בסמנטיקה טבעית:

- $S1; S2 \sim S2; S1$

ג. הוכיחו את השקילות הסמנטית הבאה בסמנטיקה טבעית:

```
if b then (if c then S1 else S2) else S3 ~ if b and c then S1 else if b and not c then S2 else S3
```

שאלה 2:

נרצה לשנות בשפת While את הפקודה While לפקודה הבאה:

- **do** S **while** b

זוהי לולאה שתמיד מתבצעת פעם אחת לפחות, והביצוע שלה נפסק כאשר התנאי b אינו מתקיים.

לדוגמה:

- **do** x = x-10 **while** x>10

יסתיים במצב בו x=5 אם יתחיל במצב בו x=55, ויסתיים במצב בו x=-3 אם יתחיל במצב בו x=7

א. הוסיפו כלל/ים לסמנטיקה הטבעית עבור פקודת while do. הכללים אינם יכולים להסתמך על מבנה לולאת while הקיים בשפת while המקורית. כלומר, אסור שהפקודה while הרגילה של תופיע בשום מקום בכללים החדשים.

ב. הוכיחו את השקילות הסמנטית הבאה בסמנטיקה הטבעית המורחבת שיצרתם בסעיף א:

```
do S while b ~ S; if b (do S while b) else skip
```

שאלה 3:

- נניח כי הSyntax עבור המספרים n היה:.

```
n ::= 0 | 1 | n 0 | n 1
```

א. הגדירו את הסמנטיקה עבור המספרים הבינאריים.

ב. הוכיחו כי:

- פונקציית הסמנטיקה שהוגדרה היא פונקציה טוטאלית.
  - A **total function** returns an output for every single possible input.

חלק ב': OCAML:

מצורפים כחלק מהתרגיל הקבצים:

- ast.ml - קובץ המייצג את סינטקס מורחב של השפה - while.
- semantics.ml - קובץ המייצג סמנטיקה של ביטויים בוליאנים ואריתמטיים.
- nos.ml - הגדרת הסמנטיקה הטבעית.

ניתן לקמפל קבצים אלו באותו האופן שעשינו זאת בתרגול הראשון (ocamlc -o)

ניתן להריץ באותה הצורה שראינו בתרגיל הראשון.

בחלק זה אנו נבנה את המפרש של השפה while באוקמל. אנחנו נעזר בהגדרות מהתרגול לעשות זאת.

מוכנים ומוכנות?

שלב א':

כפי שראינו בתרגול שפת while מורכבת מסינטקס של ביטויים אריתמתיים, ביטויים בוליאנים, ופקודות.

את כל אחד מאלו ניתן לייצג על ידי Variants באוקמל.

עיינו בקובץ ה - ast.ml שם הוגדר הVariant המייצג משתנה (Var).

בקובץ ast.ml הגדירו את הVariants הבאים:

```
n ::= int -> ??
```

(טיפ: אם קיים בנאי יחיד ניתן להגדיר ללא צורך בTag)

כעת בואו נגדיר טיפוס עבור פונקציה המקבלת משתנה (Var) ומחזירה Int

```
state ::= var -> ??
```

(טיפ: אם קיים בנאי יחיד ניתן להגדיר ללא צורך בTag)

עד כה ייצגנו את הביטויים עבור המספרים הטבעיים, עבור פונקציות state ועבור variables.

כעת נגדיר את הsyntax עבור ביטויים אריתמתיים ובוליאנים:

```
a ::= Num | Var | Add | Mult | Sub
```

```
b ::= TT | FF | Aeq | Beq | Leq | Neg | And
```

(טיפ: ההגדרה עבור Beq ו Aeq כדי שנוכל להגדיר שוויון כבר עבור ביטויים בוליאנים ואריתמטיים)

וכמובן נגדיר את הSyntax עבור פקודות:

```
stm ::= Ass | Skip | Comp | If | While
```

<u>שלב ב:</u>

כעת בואו נגדיר את הסמנטיקה עבור הSyntax שייצרנו.

פתחו את הקובץ semantics.ml שם מחכות לכם הסמנטיקות למימוש, כפי שהסברנו בתרגול בעצם מדובר על פונקציות.

תחילה נגדיר את המצב ההתחלתי

```
let default_state x = ??;;
```

(<u>טיפ: בואו ננית כי default_state היא הפונקציה שממפה כל variable לאפס, זוכרים שהיא גם state?</u>)

כעת עלינו לדאוג לעדכון המצב, בדומה לדוגמה שראינו בתרגול

$$s1 \ = \ s0[x \ -> \ A[e]_{s0}]$$

נגדיר את הפונקציה הבאה:

```
let create_state prev_state x e= ??;;
```

(<u>טיפ: חשבו! אנחנו רוצים להחזיר state, מהו בעצם state?</u>) **<u>מאתגר אך חשוב</u>**

וקדימה רוצו לממש את הסמנטיקה!

```
let rec aritmetic_semantic exp s = ???;;

let rec boolean_semantic exp s = ???;;

let rec nos exp = ???;;
```

(<u>טיפ: שימו לב טוב טוב להגדרת Ass, נרצה להשתמש בפונקציה create_state</u>).

<u>חלק ג':</u>

טוב עד עכשיו היה כיף, אבל בואו נוסיף הגדרות נוספות לסמנטיקה שלנו!

זוכרים את if-ass?

$$\textbf{if-ass}_{ns}^{tt}\ \frac{\langle x:=e\ ,\ s\rangle \to s' \qquad \langle S_1\ ,\ s'\rangle \to s''}{\langle \text{if } (x:=e) \text{ then } S_1 \text{ else } S_2,\ s\rangle \to s''} \qquad \text{if } \mathbf{A}[\![x]\!]s' \neq 0$$

$$\textbf{if-ass}_{ns}^{ff}\ \frac{\langle x:=e\ ,\ s\rangle \to s' \qquad \langle S_2\ ,\ s'\rangle \to s''}{\langle \text{if } (x:=e) \text{ then } S_1 \text{ else } S_2,\ s\rangle \to s''} \qquad \text{if } \mathbf{A}[\![x]\!]s' = 0$$

הוסיפו כלל זה לעץ האבסטרקטי:

```
stm ::= Ass | Skip | Comp | If | While | If_Ass
```

וכמובן לסמנטיקה הטבעית גם כן.

(כלומר להוסיף לעץ האבסטרקטי כמו כן לסמנטיקה הטבעית כלומר פונקציית הnos)

בתרגול היו שטענו ש Repeat Until היה יכול להיות תוספת טובה לשפת While, הוסיפו תוספת זו גם כן:

(לקובץ ast ולקובץ הsemantics גם כן)

$$\frac{\langle s,\sigma\rangle \to \sigma'}{\langle \texttt{repeat}\ s\ \texttt{until}\ b,\sigma\rangle \to \sigma'}\ \mathcal{B}[\![b]\!]\sigma' = tt$$

$$\frac{\langle s,\sigma\rangle \to \sigma',\quad \langle \texttt{repeat}\ s\ \texttt{until}\ b,\sigma'\rangle \to \sigma''}{\langle \texttt{repeat}\ s\ \texttt{until}\ b,\sigma\rangle \to \sigma''}\ \mathcal{B}[\![b]\!]\sigma' = ff$$

```
stm ::= Ass | Skip | Comp | If | While | If_Ass | Repeat
```

אני מקווה שהגעתם עד כאן וקיבלתם הבנה עמוקה ואמיתית כיצד סמנטיקה עובדת.