

Assignment 2

Responsible Lecturer: Meni Adler

Responsible TA: Adham Jabarin

Submission Date: 1/5/2022

General Instructions

Submit your answers to the theoretical questions in a pdf file called `id1_id2.pdf` and your code for programming questions inside the provided `q2.l3`, and `L31-ast.ts`, `q3.ts`, `q4.ts` files of the `src` folder. ZIP those files together (including the pdf file, and only those files) into a file called `id1_id2.zip`. Make sure that your code abides by the Design By Contract methodology.

Do not send assignment related questions by e-mail, use the forum instead. For any administrative issues (milu'im/extensions/etc) please open a request ticket in the Student Requests system.

You are provided with the templates `ex2.zip`.

Unpack the template files inside a folder. From the command line in that folder, invoke `npm install`, and work on the files in that directory, preferably working in the Visual Studio Code IDE (refer to the Useful Links). In order to run the tests, run `npm test` from the command line.

Important: Do not add any extra libraries and do not change the provided `package.json` and `tsconfig.json` configuration files. **The graders will use the exact provided files.** If you find any missing necessary libraries, please let us know.

Question 1: Theoretical Questions [30 points]

1.1 Is a function-body with multiple expressions required in a pure functional programming? In which type of languages is it useful? [3 points]

בתכנות פונקציונלי, צורך בביטויי יחידים
במקום גוף פונקציה, כי הם עוזרים
לדבר את הסקסציה של התוכנה לביטוי יחיד.
על אף שיש צורך בביטויי יחידים, הם לא
הם לא.

Q1.2

- a. Why are special forms required in programming languages? Why can't we simply define them as primitive operators? Give an example [3 points]

מחר - special forms הם למעשה פונקציות, אבל יש להם
מיון. הם לא הם special forms והם קובעים סטנדרט למעשה אותם.

סיווג זה נותן פרופורציות מיוחדות special form את האפשרות
לדבר binding כל U'P'P פרימטביל.

$\lambda - \int p' \sim \lambda, ((\lambda(x) x+1)^2)$

lambda - \int p' on' λ n of p/c. 3 o' d' λ , special form - \int

→ special form - $\int_0^1 \frac{1}{x} dx$ undefined in 1 in 2

+ $\int_0^x \lambda(x) dx = x+1$

לפי זה, נראה וזו הסיבה שיש לנו Special form- (ב')

error $\sim N^{-1/2}$

b. Can the logical operation 'or' be defined as a primitive operator, or must it be defined as a special form? Refer in your answer to the option of shortcut semantics. [4 points]

ה) $\sqrt{-1}$ לא יכול להיות special form כי זה לא \mathbb{R} וזה לא \mathbb{C}

מאמץ/אור מיוחד נעל משמעות מיוחדת. גלוי, אף-אין, מובן

מ form special, נותן לנו את היחסים הנכונים ביותר

קורטקס, גלימה, הוסכת את or δ - shortcut semantic

Q1.3 What is a syntactic abbreviation? Give two examples [4 points]

ה'צ"ו מחגיד. נוסח לקצור ול"סות קום. קל"סו, ק'ב"ו - מחגיד.

י"ב, ה' = א"ת ויק'ק משימ'ם באינ'ר'ר'ס פ'ר'מ'ר'ג'ר'ט מ'ר'ר'ר'ט. פ'ק'ר'ר'ר'

$$\int_0^1 ((\lambda(x) \cdot x) \cdot (3 + 5)) \, dx$$

2) $\int_{10}^1 \ln(x^{3+5}) (* x x)$

$$\left(\text{lambda } (x) \quad x \cdot \left((\text{lambda } (x) \quad x + x) \quad (3 \cdot 6) \right) \quad 7 \right) \quad 11(12.1) \quad \text{Ans}$$

(let (x 3.6) (y 7)) (y . x) 2005 10

Q1.4

a. What is the value of the following L3 program? Explain. [2 points]

```
(define x 1)
(let ((x 5)
      (y (* x 3)))
  y)
```

המכנית:
 $x = 1$
 $\{$
 $x = 5$
 return $\{$
 $y = x \cdot 3$
 return y
 $\}$
 $\}$

ה- let מבצע binding בעוד של מחזיק ולכן "ההיסטוריה" לא נכרת אחת את השני. במסל המיון - כל שטר נעשה פונקציונלית, ואזק שיחנה הוא הזרק האחרון, כלומר $1 \cdot 3 = 3$.

b. Read about let* [here](#).

What is the value of the following program? Explain. [2 points]

```
(define x 1)
(let* ((x 5)
       (y (* x 3)))
  y)
```

המכנית:
 $x = 1$
 $\{$
 $x = 5$
 $y = x \cdot 3$
 return y
 $\}$


מכיוון ו- let* מבצע binding לכל המשתנים ביחד, אכן הישגו של y מכיוון שטר וההיסטוריה של let* - x ולכן $y = 5 \cdot 3$ כלומר (מכיוון ו- כל פונקציונלית ואזק שיחנה (הוא האחרון) יחנה $5 \cdot 3 = 15$).

c. Annotate lexical addresses in the given expression [6 points]

```
(define x 2)
(define y 5)

(let
  ((x 1)
   (f (lambda (z) (+ x y z))))
  (f x))

(let*
  ((x 1)
   (f (lambda (z) (+ x y z))))
  (f x))
```



$y = 2$
 $x = 1$
 $f = z \rightarrow x + y + z$
return $f(x) = x + x + y = 1 + 1 + 2 = 4$

d. Define the let* expression in section c above as an equivalent let expression [3 points]

$\text{let } (x \ 1) \ x + x + y$

e. Define the let* expression in section c above as an equivalent application expression (with no let) [3 points]

$(\text{define } x \ 1)$
 $(\text{lambda } (z) (+ x \ y \ z) \ 1)$