1. כן. Special Form הוא ביטוי שיש לה שיטת איוולואציה שונה מהרגיל. Let היא בסך הכל syntax Abbriviation של define, ואנחנו רואים בקוד שלdefine יש מסלול איוולואציה אחר, כלומר, הוא איננו פרימיטיבי מבחינת דרך האיוולואציה שלו.
   1. אנחנו יודעים שזה syntazx sugar שהופך לאפליקציה של למדה
   2. בגלל שזה מומר לAST של appExp אז זה לא ספיישל פורם
   3. הערה: אנחנו תריכים להכיר איך דרך האולואציה הרגילה עובדת
2. משפטים שהם קבילים מבחינה תחבירית אבל בהערכה אפליקטיבית היינו מקבלים שגיאה לעומת הערכה נורמאלית בה היינו עשויים לדלג על חישוב הערך שלהם וכך לא לאתר את השגיאה.
   1. (- #t 1) אופרטור חיסור לא יודע לקבל בוליאנים
   2. (/ n 0) אסורה חלוקה באפס
   3. (-) אופרטור פרימיטיבי ללא אופרנדים
3. להשלים
4. ValueToLit נצרכת כאשר אנחנו משתמשים בApplicative evaluation במודל ההחלפה – אנו מחליפים משתנים בערכים ולאחר מכן צריכים להחזיר את הערך לצורה של ביטוי ע"מ שיתקבל בAst. לעומת זאת, בNormal Evaluation ההחלפה נעשית ע"י ביטויים (CExps) לאורך הדרך ורק לבסוף מבוצעת הערכה, לכן הפונקציה הנ"ל לא נצרכת.
5. נורמאל מהיר מאפליקטיב:  
   ((lambda (x)(if #t 1 (\* x x x x x x x x))) #t)  
   בשיטה הנורמאלית, בחישוב הערך של הקוד לא נידרש לבצע את ההכפלה הארוכה של איקס, וישר יחזיר 1.  
   בשיטת האפליקטיב, נחשב את הערך של כל אחד מהצעדים, כולל את חישוב (המיותר) ההכפלה של איקס, ונחזיר 1.  
   אפליקטיב מהיר מנורמאל:  
   (define double100 (\* 100 2))

(define sum5 (lambda (x) (+ x x x x x)))

(double100 sum5)  
בקוד לעיל, בשיטה הנורמאלית נחליף כל אחד מהמשתנים בsum5 בביטוי (\*100 2) ולאחר מכן נחבר אותם אחד לשני ונאלץ לעשות הערכה של כל אחד מהם.  
לעומת זאת, בהערכה אפליקטיבית נחשב את (\* 100 2) פעם אחת בשורה הראשונה, ונחבר 200 חמש פעמים. (פי 2 פחות חישובים).