פתרון חלקי והערות בדיקה של מועד א' סמסטר ב' 2014 במבוא מורחב למדעי המחשב

שאלה 1

סעיף ב':
שני פתרונות אפשריים לדוגמא:

```
def find_root(self):
    return NR( self.evaluate, self.derivative().evaluate)
def find_root(self):
    return NR( lambda x: self.evaluate(x) , lambda x: self.derivative().evaluate(x) )
```

שני הפרמטרים הם מטיפוס פונקציה שמקבלת קלט מספרי ומחזירה פלט מספרי. טעות נפוצה: פרמטרים מטיפוס Polynomial. עוד טעות נפוצה: lambda x: self.evaluate (זו פונקציה שמחזירה פונקציה ולא מספר).

סעיף ג': מימוש אפשרי (ישנם כמובן מימושים אחרים נכונים)

```
class Polynomial():
    def __init__(self, coeffs_lst):
        self.sparse_coeffs = []
        if len(coeffs_lst) > 0 and isinstance(coeffs_lst[0], tuple):
            self.sparse_coeffs = coeffs_lst
        else:
            for i in range(len(coeffs_lst)):
                if coeffs_lst[i] != 0:
                    coeff_tuple = (i, coeffs_lst[i])
                    self.sparse_coeffs.append(coeff_tuple)

    def derivative(self):
        lst = []
        for coeff_tuple in self.sparse_coeffs:
            if coeff_tuple[0] != 0:
                new_tuple = (coeff_tuple[0] - 1, coeff_tuple[0]*coeff_tuple[1])
                lst.append(new_tuple)
        return Polynomial(lst)
```

טעות נפוצה: בחירה בייצוג שאינו קומפקטי, ולכן הקריאה ל __init__ מהפונקציה derivative תדרוש O(n) זמן.
עוד טעות נפוצה: קריאה ל- __init__ עם פרמטר שאינו מתאים לטיפוס הקלט שהיא יכולה לקבל.

שאלה 3

סעיף א':
הרקורסיה זהה למשחק המקורי:
- אם יש מצב המשך מפסיד, המצב הנוכחי מנצח.
- אם כל מצבי ההמשך מנצחים, המצב הנוכחי מפסיד.

ההבדל היחיד ביחס למשחק המקורי הוא שכאן קבוצת המצבים הממשיכים קטנה יותר, וניתן לתאר מצב המשך ע"י זוג המימדים של המלבן המתאים.

סעיף ג':
הטענה של יעל נכונה.
לטבלה ריבועית, השחקן הראשון מפסיד.
הוכחה באינדוקציה על גודל הטבלה:
לטבלה 1x 1 – ברור.

נניח נכונות לכל טבלה ריבועית עד גודל n x n, ונוכיח לטבלה בגודל (n+1) x (n+1).

השחקן הראשון, אם אינו אוכל את כל הטבלה ומפסיד, מותיר טבלה לא ריבועית, בלי הגבלת כלליות בגודל k x (n+1) כאשר $1 < k < (n+1)$.
השחקן השני יעשה צעד שהוא תמונת ראי ויותיר טבלה ריבועית בגודל k x k.
על פי הנחת האינדוקציה, זה מצב מפסיד לשחקן הראשון.

שאלה 4

סעיף א': יש דוגמה כזו. טעות נפוצה: הדוגמה שניתנה איננה מילת קוד חוקית.

סעיף ב': לא ייתכן. הסבר: אילו היתה דוגמה כזו, המשמעות היא שישנן שתי מילות קוד במרחק חמש. אבל בגלל ביט הזוגיות, המרחק בין שתי מילות קוד חייב להיות זוגי.
טעות נפוצה: טיעון לפיו המרחק בין שתי מילות קוד חייב להיות כמרחק של הקוד d=4. שימו לב שמרחק של קוד הוא המרחק המינימלי בין שתי מילים, ולא כל המילים בהכרח במרחק כזה זו מזו.

סעיף ג': יש דוגמה כזו. טעות נפוצה: הדוגמה שניתנה איננה מילת קוד חוקית.

סעיף ד': טעויות נפוצות:

- ביטוי עבור ec1 ו/או ec2 שבחלקו הינו פונקציה של m ובחלקו מספר מפורש (שימו לב ש- m ניתן כפרמטר, והביטוי צריך להיות כולו פונקציה של m). למשל:

```
ec1 = trans[2**m-1, 2**m+3]

ec2 = trans[2**m+3, 2**m+6]
```

- כמו כן בחלק מהתשובות האינדקסים היו שגויים, בד"כ גדולים או קטנים ב- 1.

- בתנאי לטעות בודדת, חלק כתבו if ec1 != ec2. זהו תנאי לא מספיק, שכן הטעות יכולה להיות בביט הזוגיות, והדבר יגרור ערך החזרה לא נכון בהמשך.

- אינדקס הטעות הינו err = int(xor(ec, ec1), 2) (אפשר גם עם ec2). חלק שכחו להמיר לעשרוני. לא הורדו נקודות אם היתה טעות קלה באופן ההמרה לעשרוני (למשל בלי הפרמטר השני לפונקציה int, או שם הפונקציה לא נכון).