

הסבר תרגיל 2 אימות פורמלין:

כדי להוכיח קשת (b,a) נציין אותה כ-a b כאשר כל קשת תופר עם פסיקים בדיק כמו שנכתב בסמוך המופיעין של Ex2 כלומר:

```
Please enter your graph or EXIT if you want to exit the software
1 2,3 4,4 5
```

הפלט יראה כך:

```
[['x1'], ['x2'], ['x3'], ['x4'], ['x5'],
```

שינוי אחד הוא שלא ניתן להוציא 0 כצומת מאחר ש-0 משמש להכנסת צומת שאין לא קשותות כלומר:

```
Please enter your graph or EXIT if you want to exit the software
0 1,0 2,3 0,4 0
['x1'], ['x2'], ['x3'], ['x4'],
```

0 a וההפר

בעצם אומר שצומת a הוא חלק מהגרף גם אם אין לו קשותות בכלל מבון שצירוף של 0 a וההפר לא יפגע ב-a אם המשתמש יכניס קשותות ש-a משתתף בהם לדוגמה:

```
Please enter your graph or EXIT if you want to exit the software
1 0,2 1,1 2
['x1', 'x2'],
```

ליציאה הקלד EXIT

```
Please enter your graph or EXIT if you want to exit the software
EXIT
```

כמה דברים על הקוד:

(1) הקוד מימוש את הפוואדו קוד שנראה בהרצאה 3 אחד לאחד כלומר בניתי 2 פונקציות DFS-A SMBATZUL GRF G AT ALGORITM DFS, COMON BNITI AT B-SHMBATZUL AT ALGORITM DFS UL GRF G HAFUR.

(2) בניתי 2 מחלקות:

1. vertex - מייצגת צמת ומכליה את התכונות:
א..שם של הצומת - הערך המספרי שלו.

ב. קשותות בגרף G - לדעת את השכנים שלו בגרף G.

ג. קשותות בגרף G ההפר - לדעת את השכנים בגרף G ההפר.

ד. צומת חדשה/ישנה תכונה זאת תשמש לדעת אם עברנו על הצומת במהלך ה-.dfs

edge - מייצגת קשת ומכליה את התכונות:

א. המספר של הצומת ממנו יוצא.
ב. המספר של הצומת אליה נכנס.

(3) הקוד מתחילה בכך שהוא מקבל את string שהמשתמש הכניס בודק אם תקין מבחינה התבנית שהכנסנו, אם כן מפרק אותו לחליים באמצעות פסיקים לאחר מכן עובר על כל מקטע ובונה את הקשותות והצמתים הרלוונטיים שלו,

לאחר מכן מבצע dfs על G כמו הפואידו קוד בהרצאה:

```
DFS-A ( $G(V, E)$ ,  $h$ )
for every  $e \in E$  mark  $e$  "new"
for every  $u \in V$  do {
     $f(u) \leftarrow \text{NULL}$ 
    mark  $u$  "new"
}
 $i \leftarrow 1$ 
```

Strongly Connected Components

```
while there are new vertices do {
    let  $v$  be such a vertex
    mark  $v$  "old"
    while  $v$  has new outgoing edges or  $f(v) \neq \text{NULL}$  do {
        if  $v$  has new outgoing edges then do {
            let  $v \xrightarrow{e} w$  be a new edge
            mark  $e$  "old"
            if  $w$  is new then do }
```

Strongly Connected Components

```
mark  $w$  "old"
 $f(w) \leftarrow v$ 
 $v \leftarrow w$ 
}
else ( $f(v) \neq \text{NULL}$ ) then do {
     $h(v) \leftarrow i$ 
     $i \leftarrow i + 1$ 
     $v \leftarrow f(v)$ 
}
```

Strongly Connected Components

```

}
 $h(v) \leftarrow i$ 
 $i \leftarrow i + 1$ 
}
DFS-A labels vertices  $h(v)$  by finishing
visit order
```

אחר מכן מבצע dfs על G ההפוך כמו הפואדו קוד בהרצאה

```

DFS ~ B (G(V, Et), h)
for every e ∈ Et mark e "new"
for every u ∈ V do {
    f(u) ← NULL
    mark u "new"
}

```

Strongly Connected Components

```

while there are new vertices do {
    let v be a new vertex for which f(v) is maximal
    S ← {v}
    mark v "old"
    while v has new outgoing edges or f(v)=NULL do {
        if v has new outgoing edges then do {
            let v → w be a new edge
            mark e "old"
            if w is new then do {

```

Strongly Connected Components

```

        mark w "old"
        f(w) ← v
        S ← S ∪ {w}
        v ← w
    }
    else (f(v) ≠ NULL) then do {
        v ← f(v)
    }
print set S is an SCC
}

```

לאחר מציאת רכיבי הקשרות אנו קוראים לפונקציה שמדפיסה את הפלט, היא מוסיף X לכל השמות.

4) כל הקוד מכיל הערות להבנה של היישום.