



שמונה בעקבות סָם

לפני כ-142 שנה הומצאה "חידת ה-15". בחידה זו נתון לוח של 4×4 שבו רשומים המספרים

$\{1,2,3,4,5,6,7,8,9,10,11,12,13,14,15\}$

ראו ציור 1:

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	

ציור 1

בחידה זו אנו מתחילים במצב של ציור 1 וצריכים להגיע למצב של ציור 2:

1	2	3	4
5	6	7	8
9	10	11	12
13	15	14	

ציור 2

חידה זו הומצאה על-ידי סם לויד (אם כי ייתכן שהוא גנב אותה מִדְּוֹר בשם נויס פאלמר צ'פמן). כמובן שאי-אפשר לפתור אותה. סם לויד הציע פרס גבוה למי שיפתור אותה. יותר מאוחר הוא ניסה לרשום עליה פטנט, אך בית-המשפט סירב לאשר זאת, בנימוק שאי-אפשר לרשום פטנט על משהו שלא קיים. לתיאור מפורט של החידה וההיסטוריה שלה אפשר לקרוא בקישור [הזה](#).

בחידה של היום אנו חורגים מן החידה של סם לויד בשלוש בחינות:

1. לוח ההתחלה שלנו מכיל שני מקומות ריקים ואת המספרים 1,2,3,4,5,6 (ראו ציור 3).
2. הלוח הוא 4×2 (ראו ציור 3).
3. אנו נרצה לכתוב אלגוריתם המחשב את מספר הפעולות המינימלית הנדרשות להגיע מכל מצב למצב בציור 3. כלומר הקלט של האלגוריתם שלנו הוא טבלה 4×2 , המכילה את המספרים 1,2,3,4,5,6 ושני רווחים. והפלט הוא זה מספר ההזזות המינימלי של הרווחים או המספרים כך שנסדר את הטבלה כמו בציור 3.

2	4	6	
1	3	5	

ציור 3

לדוגמה אם הלוח שלנו נמצא במצב התחלתי של ציור 4

	2	4	6
	1	3	5

ציור 4

הפלט של האלגוריתם הוא 6 (ראו ציור 5):

	2	4	6
	1	3	5

2		4	6
	1	3	5

2	4		6
	1	3	5

2	4	6	
	1	3	5

2	4	6	
1		3	5

2	4	6	
1	3		5

2	4	6	
1	3	5	

ציור 5

נ"ב. והפעם הפתעה – חידה בתוך חידה:
מה זה ירוק וארוך על גלגלים?

פתרון:

בחידה זו אנו מתבקשים לכתוב אלגוריתם המוצא את מספר הפעולות המינימלי שצריך כדי להזיז כל מצב למצב ההתחלתי שנמצא בציור 3: הרעיון של הפתרון הוא לבנות גרף $G(V,E)$ להגדרה של גרף ראה [קישור](#). הצמתים של הגרף הם המצבים האפשריים. נחבר שתי צמתים x,y של הגרף אם אפשר להזיז רווח ולהגיע מצב x למצב y . כעת נחשב את המסלול הקצר ביותר בין המצב שלנו למצב בציור שלוש. קוד שעושה בדיוק את זה אפשר לראות [בקישור](#).



ציור 6 קישור למחברת מתמטיקה המכילה את הקוד

נסביר את הקוד שפותר את החידה: הפתרון שלנו מרכב שתי פונקציות אחת
 computP והשנייה getAllelment.
 ראשית נתאר את הפונקציה computP, המחשבת את כל הפרמוטציות האפשרויות
 ליצר ממצב התחלתי p והחלפות של איבר i עם עברי הרשימה l.

1. computP מקבלת כקלט שלושה פורמטים

a. p, פרמוטציה

b. i מקום שאותו נחליף

c. l רשימה l של החלפות

הפלט של הפונקציה computP הוא רשימת כל הפרמוטציות האפשרות
 מהחלפה של רווח שנמצא במוקום i.

הפונקציה computP

```
computP[p_, i_, l_] :=
```

```
(* input varubal
```

```
p is the current permutation
```

```
i the location of the space
```

```
l the neighbor of i in the grapg of the puzzle.
```

```
*)
```

```
(*
```

```
The function compute all the possible movement of the space i  

and out \
```

```
put all the new permutation
```

```
*)
```

```
Module[{p1 = p, k, ans = {}},
```

```
For[k = 1, k <= Length[l], k++,
```

```
p1 = p;
```

```
p1[[i]] = p[[l[[k]]];
```

```
p1[[l[[k]]] = p[[i]];
```

```
AppendTo[ans, p1]
```

```
];
```

ans

]

2. הפונקציה `getAllelment` מקבלת כקלט את:

a. | מצב התחלתי של הלוח או את הפרמוטציה ההתחלתית.

b. גרף המתאר את מבנה הלוח.

c. מספר המסמל את המקומות החופשיים.

3. הפונקציה מבצעת את הפעולות הבאות

a. מצא את המקום של שני רווחים (המיקום נמצא במשתנים $i1, i2$)

b. חשב את כל ההחלפות האפשריות של הרווחים. ההחלפות נמצאות ברשימות $i1, i2$.

c. הוסף את הקשתות הרלוונטיות לרשימת הקשתות `ed`.

d. אם מצאת צומת (פרמוטציה חדשה) הוסף אותה למחסנית `ds`.

4. הפלט של הפונקציה `getAllelment` הוא רשימת הפרמוטציות שהצלחנו ליצר

באמצעות החלפת רווחים רשימה זו נמצאת בפרמטר הראשון `set`. רשימת

הקשתות של הגרף של כל פרמוטציות הקשתות נמצאות המשתנה `ed`.

5. הפונקציה `getAllelment`

```
getAllelment([l_, G3_, n_] := (* l is the Initial permutation,
```

```
G3 is the board structure of the puzzle,
```

```
n A number that symbolizes the space.*)
```

```
Module[{set, i, ds, x, y, ed, i1, i2, l1, l2},
```

```
(* set is the set of all vertex,
```

```
i is an index go over all Neighbor
```

```
ds is Stack Data Structure contains all nodes that have yet to be \  
discovered and have not yet been addressed
```

```
x the node that we visit know
```

```
y the list of the neighbors of x
```

ed a list of all edges of the permutation graph

i1 the position of the first n

i2 the position of the second n

l1 the neighbors of node i1

l2 the neighbors of node i2

*)

```
set = {};
```

```
ed = {};
```

```
ds = CreateDataStructure["Stack"];
```

```
ds["Push", l];
```

```
While[Not[ds["EmptyQ"]],
```

```
  x = ds["Pop"];
```

```
  i1 = Position[x, n][[1, 1]];
```

```
  i2 = Position[x, n][[2, 1]];
```

```
  l1 = AdjacencyList[G3, i1];
```

```
  l2 = AdjacencyList[G3, i2];
```

```
  y = Flatten[{computP[x, i1, l1], computP[x, i2, l2]}, 1];
```

```
  For[j = 1, j <= Length[y], j++,
```

```
    AppendTo[ed, x \[UndirectedEdge] y[[j]]];
```

```
    If[Not[MemberQ[set, y[[j]]],
```

```
      AppendTo[set, y[[j]]];
```

```
      ds["Push", y[[j]]];
```

```
    ];
```

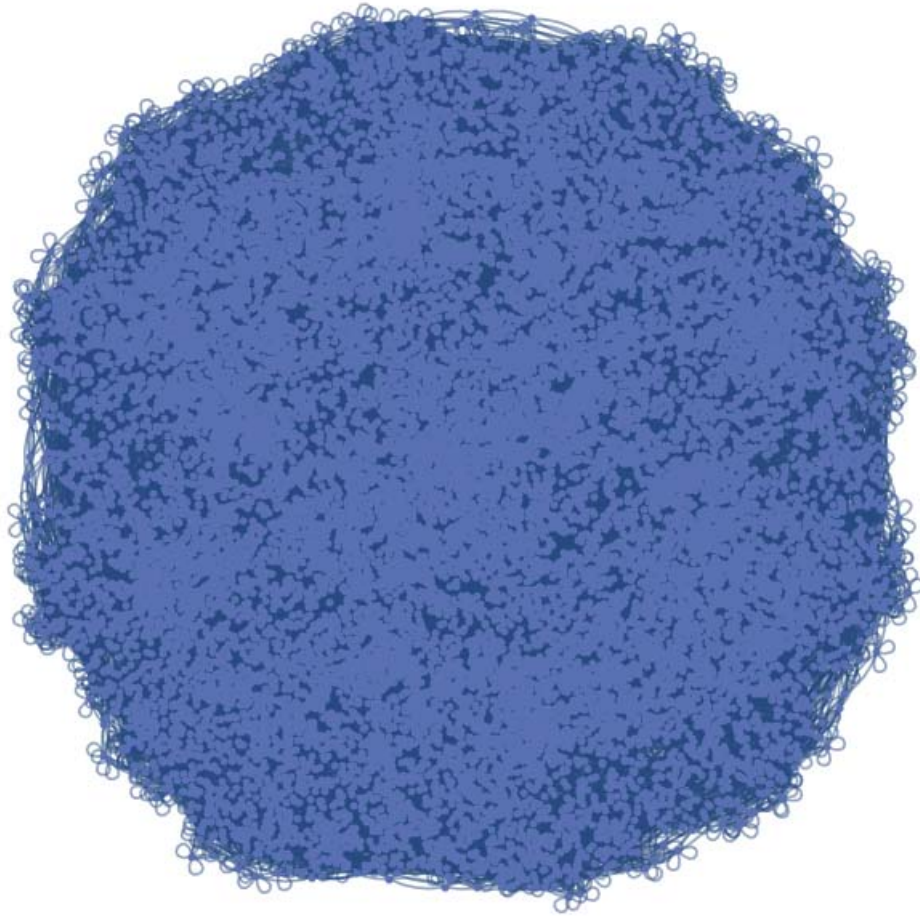
```
  ];
```

```
];
```

```
{set, ed}
```

```
];
```

ניתן לראות את הגרף שחושב באמצעות הפונקציה `computER` בציר 6.



ציור 7 הגרף myg של כל הפרמוטציות

כעת, מה שנישאר לעשות זה לחשב את המרחק בין מצב הלוח בהתחלה למצב לוח
בציור 3. זה ניתן לעשות על-ידי חישוב המרחקים הקצרים זאת ניתן לעשות על-ידי
הפקודה

```
A = GraphDistanceMatrix[myg];
```

כאשר myg הוא הגרף מציור 7. כדי לחשב את המרחק של מצב פשוט כל מה שאנו
צרכים זה למצוא את המיקום של המצב שאנו רוצים לחשב לו את המרחק ולהסתכל
במקום המתאים במטריצה A. נסתכל על הדוגמה מציור 4. מטריצה זו מייצגת על-ידי
הפרמוטציה

$$p = \{7,7,1,2,3,4,5,6\}$$

נמצא את המיקום של פרמוטציה זו במערך השמות שלנו על-ידי הפקודה

```
pos = Position[vname, {7,7,1,2,3,4,5,6}][[1,1]]
```

והמרחק יוצא

$A[[1, pos]]$

6 במקרה הזה.