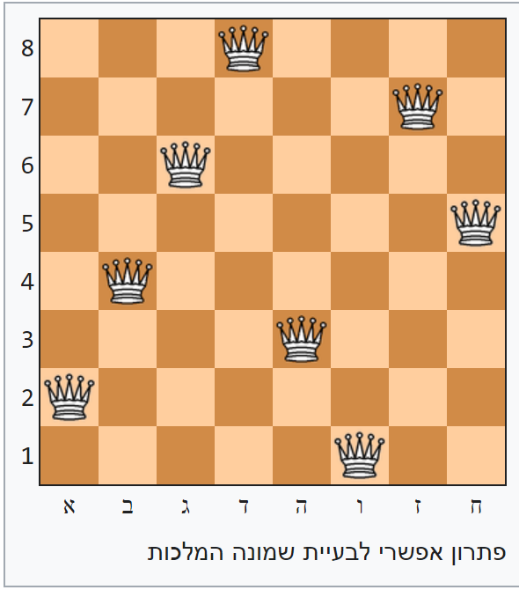
תרגיל #3—בעיית המלכות

כמו שלמדנו בכיתה, בעיית המלכות ידועה בתחום של מדעי המחשב ויש לנו מה להוסיף על רקע השיטות שלמדנו בקורס שלנו. ניתן לראות מידע כללי על הבעיה בכתובת: 

[https://he.wikipedia.org/wiki/חידת\_שמונה\_המלכות](https://he.wikipedia.org/wiki/%D7%97%D7%99%D7%93%D7%AA_%D7%A9%D7%9E%D7%95%D7%A0%D7%94_%D7%94%D7%9E%D7%9C%D7%9B%D7%95%D7%AA)

בקישור למעלה מופיע את החישוב למספר המצבים כאשר n=8:

4,426,165,368 . מספר זה מייצג, לכאורה את הממוצע של ריצות עבור British Museum Algorithm שלמדנו בכיתה. ולכן כנראה נעבוד רק עם דברים יותר חכמים.

בתרגיל, נתמקד בכמה אפשריות:

1. British Museum – ברירת המחדל כשיטה לא טובה
2. DFS עם Backtracking – ברירת המחדל של שיטה סבירה
3. אלגוריתם שלמדנו בכיתה: Forward Checking
4. אלגוריתם שלמדנו בכיתה: Heuristic Repair

אנחנו ניתן לכם קוד בסיסי במודל / Colab בכתובת:

<https://colab.research.google.com/drive/1zcpkeDW5CNrTSDpespNpnsDkODLof-Ob?usp=sharing>

עליכם ליישם כמה אלגוריתמים:

1. האלגוריתם הבריטי
2. אלגוריתם עם Forward Checking
3. אלגוריתם עם רכיב תהליך סטוכסטי כמו נתינת חלק מהמלכות בלוח, ואז שימוש בתיקון יוריסטי (Heuristic Repair) כמו על ידי DFS, Forward Checking או שיטה אחרת, כמו Hill Climbing.

הקוד עובד עם DFS כולל Backtracking ונעבור על הקוד בזמן התרגיל. בתוך הקוד קיים משתנה, *number\_of\_moves* שאמור לספור כמה פעמים הוזזה מלכה – או כדי לשים אותה על הלוח, או להוריד אותה. משתנה זה קיים כדי להשוות בין השיטות. בתוך DFS, כל פעולה של Backtracking יעלה 2 פעולות בתוך המונה—אחד להוריד את המלכה ואחד כדי לשים אותה במקום חדש. שימו לב שאין מחיר להפעיל את הפונקציה, next\_row\_is\_safe, ושווה להרחיב את הפונקציה הזאת כדי להוריד את מספר הפעולות הנספרות בתוך האופציות של Forward Checking וHeuristic Repair. לעומת זאת, המונה, *number\_of\_iterations* סופרת את המספר האיטרציות של הלולאה—ללא קשר עם הוזזה מלכה או לא. ניתן לראות שהמספר יהיה גבוה משמעותית מהמספר במונה השני.

שלבים בתוך התרגיל:

1. המשיכו ליישם את המונה .*number\_of\_moves* כאשר n=4, הפלט אמור להיות:

Number of iterations: 31

Number of queens placed + backtracks: 12

וכאשר n=8 הפלט אמור להיות:

Number of iterations: 982

Number of queens placed + backtracks: 218

1. תיישמו את האלגוריתם הבריטי. הנח שאין מחיר נוסיף לשים n מלכות, ולכן בתוך האלגוריתם הזה המונה "רק" יספור את המספר האיטרציות עד שנמצא פתרון. הריצו את הקוד עבור n=4 וn=6. שימו לב שבגלל שמדובר על אלגוריתם סטוכסטי (עם רנדומאליות) יהיה צורך לקחת את הממוצע של כ20 ריצות ואז לפלוט את הממוצע של המונה. אולי יהיה לכם צורך לעבור למבנה אחר כדי לבדוק אם הפתרון טוב או לא.
2. יישם אלגוריתם עם Forward Checking. כמה חסכת ביחס לאלגוריתם הפשוט של DFS? עד איזה n הצלחת לקבל פתרון? הדפיסו את המונים בשביל הפתרון עם הn הגדול ביותר שמצאת.
3. יישם אלגוריתם עם רכיב סטוכסטי. כמו באלגוריתם הבריטי, אין מחיר לשים n מלכות לעומת 1. אעפ"כ, יהיה מחיר של 1 במונים להזיז מלכה כמו באלגוריתמים האחרים. האם הצלחתם לשפר את הביצועים בממוצע של 20 ריצות על פני האלגוריתם עם Forward Checking?

ציון התרגיל יהיה מורכב מ:

20 נקודות: יישום נכון של המונה וגם של האלגוריתם הבריטי

20 נקודות: יישום נכון של אלגוריתם עם Forward Checking

20 נקודות: יישום נכון של אלגוריתם עם רכיב סטוכסטי

20 נקודות: תיעוד לפתרון שלכם—כולל הסבר של שימוש במערכים או מבנים נוספים

20 נקודות: ביצועים. מה היו הערכים של המונים שלכם עבור n=15 בממוצע? יותר נקודות יינתנו עבור פתרונות עם ביצועים טובים יותר (כימות על ידי האחוזים שירד במונים לעומת הפתרון של DFS)

בהצלחה!!!

כמו בתרגיל הראשון, הגשות על ידי מודל