

## עובדת גמר תכנון ותוכנות מערכות - טלפונים חכמים אנדרואיד

צבי בדש 214553034

כיתה י"ב-2, עתיד דביר, בהנחיית מר אורן אביעזר

### תקציר

במסמך זה,أتאר את העבודה על הפרויקט “Sudo Solve” – פרויקט הגמר אותו בחרתי לפתח במסגרת עבודה הגמר תכנון ותוכנות מערכות - טלפונים חכמים אנדרואיד, הנדרשת להשלמה ל- 10 יחידות מחשבים בתעודת הבגרות.



**תוכן עניינים**

<b>I</b>	<b>דברי פתיחה והקדמות</b>	
6	הקדמה אישית .....	1
6	הקדמה לפרויקט .....	2
6	שלבי השימוש באפליקציה .....	3
6	רעיון הפרויקט - סקירה כללית .....	4
<b>II</b>	<b>תיאור תהליכי פיתוח התוכנה</b>	
7	חוiot המשמש .....	1
8	1.1 דיאגרמת פעילות .....	
8	1.2 שפה גרפית ועיצוב .....	
8	1.3 רמת הפשטה של עקרונות התכונות באפליקציה .....	
9	2 דרישות/מוגבלות להפעלת התוכנה .....	2
9	2.1 דרישות מסוימות להפעלת התוכנה .....	
9	2.2 הרשות לצריכה התוכנה .....	
9	2.3 פירוט חומרה מיוחדת .....	
9	2.4 פירוט סביבת הפיתוח והבדיקות .....	
9	2.4.1 סביבת הפיתוח .....	
10	2.4.2 בדיקות והרצאות .....	
11	3 ספריות חיצונית .....	
12	4 תיקון ובניית מסד הנתונים .....	
12	4.1 בחירת שירותים מסד הנתונים .....	
12	4.2 בחירת הנתונים לשמירה .....	
13	4.3 תיקון מסד הנתונים .....	
13	4.4 התממשקות עם התוכנה .....	
14	5 עיצוב השירות וכנתיבת קוד ה- Backend (בקקרה) .....	5
14	5.1 תכנון ארכיטקטורת השירות .....	
16	5.2 בניית API עם Flask .....	
19	5.3 הצגת קוד שרת לדוגמה - ייצור הסודוקו .....	
20	6 תיאור קוד האפליקציה .....	6
20	6.1 תיאור הקוד - מבט מלמעלה .....	
20	6.1.1 התיקייה com.zvibadash.sudosolve - תיikit האב של קוד המקור לפרויקט .....	
21	6.1.2 התיקייה com.zvibadash.sudosolve.activities - ה- Activities בפרויקט .....	
22	6.1.3 התיקייה com.zvibadash.sudosolve.database - ניהול ה- db .....	
23	6.1.4 התיקייה com.zvibadash.sudosolve.networking - תקשורת עם השירות .....	
24	6.1.5 התיקייה com.zvibadash.sudosolve.sudokuboard - לוגיקה ועיצוב לוח הסודוקו בפרויקט .....	
25	6.2 תיאור קבצי ה- Activities .....	
26	6.2.1 הפעולות CameraModeActivity .....	
27	6.2.2 הפעולות CelebrateActivity .....	
28	6.2.3 הפעולות HelpActivity .....	

29	הפעולות	HomeActivity	6.2.4
30	הפעולות	LoginActivity	6.2.5
31	הפעולות	RegisterActivity	6.2.6
32	הפעולות	SudokuSolvingActivity	6.2.7
33	הפעולות	WelcomeActivity	6.2.8
34	טיאור קבצי ה- XML וקבצי manifest	AndroidManifest.xml	6.3
34	קובץ ה-	login_menu.xml	6.3.1
35	הקובץ	main_menu.xml	6.3.2
35	הקובץ	attrs.xml	6.3.3
35	הקובץ	styles.xml	6.3.4
36	הקובץ	themes.xml	6.3.5
37	הקובץ	colors.xml	6.3.6
37	הקובץ	7	6.3.7
38	הקובץ SudokuBoardView.java	6.4	
39	טיאור המונה על דרישות פרויקט הגמר		

40	<b>III מדריך למשתמש</b>	
41	פתרונות חשבון .....	0.1
42	מසך הבית .....	0.2
42	מצב מצלמה .....	0.3
43	מצב אקראי .....	0.4
43	מසך הפתרון .....	0.5
44	הצלחת לפתרו, יפה!	0.6

45	<b>IV רקע תיאורי</b>	
46	על ההיסטוריה של חידת הסודוקו .....	1
47	על בעיות NP-שלמות .....	2
47	2.1 מחלקות סיבוכיות .....	
47	המחלקה P .....	2.2
47	המחלקה NP .....	2.3
47	سؤالת מיליון הדולר, $P \stackrel{?}{=} NP$ .....	2.4
48	מה הקשר לסודוקו?	2.5
49	על תוכנות אילוצים 1 CSPs .....	3
49	מהי בעיתת סיפוק אילוצים (CSP) .....	3.1
50	אלגוריתם AC-3 לשימירה על עקביות קשת (Constraint Propagation) .....	3.2
52	אלגוריתם MAC .....	3.3
54	היררכיות MRV ו-LCV .....	3.4
55	על השימוש בלמידה מכונה לצורכי זיהוי וקלסיפיקציה .....	4
55	מהי למידת מכונה (בערך) ומהי קלסיפיקציה .....	4.1
55	רשומות עצביות מלאכותיות .....	4.2
56	הספרייה Tensorflow .....	4.3

57	מאגר MNIST	4.4
58	אימון ו- Benchmarking	4.5

**60** **V** **דברי סיום ורשימהביבליוגרפיה**

60	רפלקציה	1
60	תודות	2
61	ביבליוגרפיה	3

**62** **VI** **נספחים**

62	קישור ל-GitHub	1
62	כתובת השרת	2

דף זה הושאר ריק באופן מכוון

# חלק I

## דברי פיתחה והקדמה

ב חלק זה אסקור את המוטיבציה שלי לפרויקט ותן לו הקדמה.

### 1 הקדמה אישית

במסגרת לימודי הנדסת תוכנה בבית הספר עתי"ד דבר, נתקננו לפתח אפליקציית Android. כמשמעותי לראשונה שהפרויקט אותו נידרש כתוב, חששתי מעט, בעיקר בגלל שלא היה לי שום רקע בפיתוח אפליקציות מסווג כלשהו. לאחר שהבנתי שאט הפרויקט נכתב בשפת Java אותה כבר הכרתי, ונסלמד את הדריש לפיתוח האפליקציה "מאפס", ונגעתי. במהלך השנה למדנו את עקרונות הפיתוח לSYSTEM Android עם מר אורן אביעזר, ובהדריכתו רכשנו את העקרונות הדרושים (הן מבחינת הפיתוח ל-Android והן מבחינת עקרונות פיתוח התוכנה של פרויקט בסדר גודל זהה) לכנתיבת הפרויקט.

### 2 הקדמה לפרויקט

הפרויקט אותו בחרתי לפתח הוא "sudo solve" - משחק מילים על הפקודה sudo במערכות Unix ועל הדמיון למילה Sudoku - אפליקציה שמרתה היא לאפשר למשתמש הקצה לסרוק חידת סודוקו לתוכה האפליקציה, לתת לו לננות לפתרור אותה בסביבה אינטראקטיבית ולבסוף להציג לו פתרון לחידה.

בחרתי בנושא זה כי תמיד עניינו אותי תוכנות הפורטאות סודוקו, בעיקר בגלל שסודוקו היא בעיה NP-שלמה (ועל כך עוד נרחב בהמשך) וכי תמיד אהבתי לפתור חידות סודוקו.

### 3 שלבי השימוש באפליקציה

את שלבי השימוש באפליקציה ניתן לחלק לשולשה שלבים עיקריים –

1. קבלת הסודוקו קלט באמצעות מהדרכים הבאים:

- (א) סריקת הסודוקו מהמסך לתוכה האפליקציה ועיבוד התמונה.
- (ב) הגרלת סודוקו אקרי ברמת קושי מסוימת.

2. ניסיון פתרת הסודוקו על ידי המשתמש.

3. הצגת הפתרון למשתמש (אם יבחר בכך).

4. שיתוף הפתרון.

את הרקע התיאורטי הנדרש לקבלת תמונה מלאה על שלבים 1 ו- 3 ניתן ב חלק VI.

### 4 רעיון הפרויקט - סקירה כללית

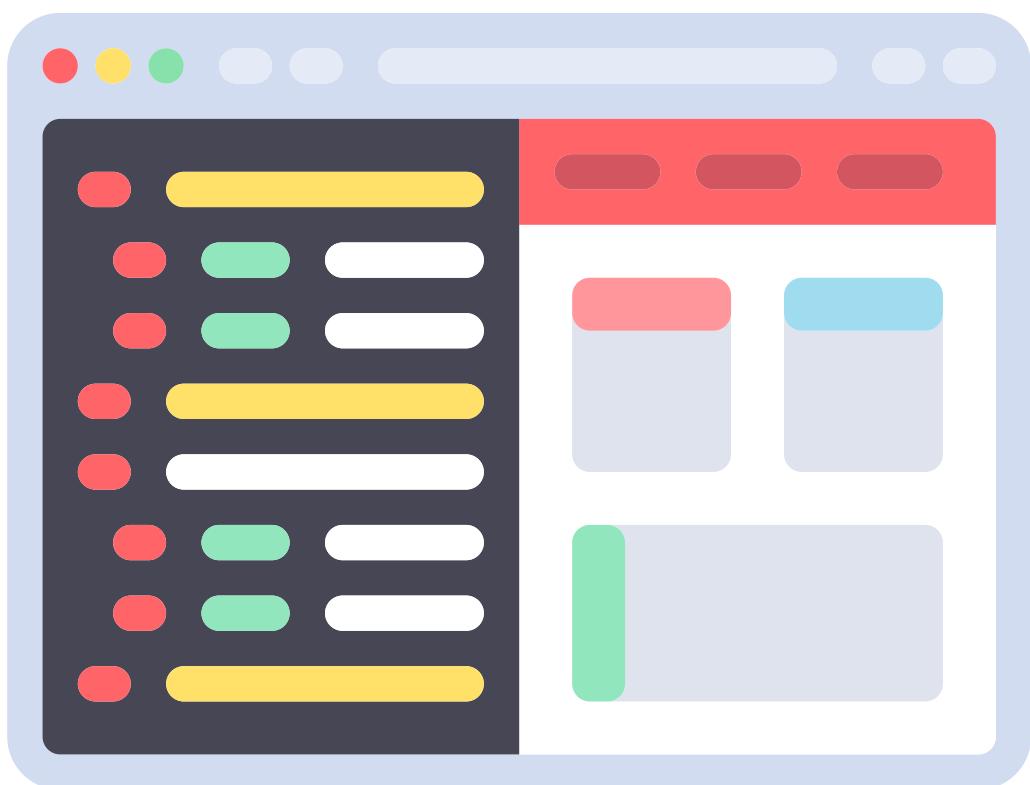
**קהל היעד** של הפרויקט הוא כולל האנשים שחויבים את משחק הסודוקו, החל במתחלים גמורים שרוצים ללמידה את חוקי המשחק טוב יותר וכלה במקצוענים שרוצים להשתפר. כולל האפליקציה וכל מאפייניה פותחו בגישה של פתרון סודוקו בסביבה מקלה, עוזרת, ומובנת.

**ובאופן מדויק** ניתן לבחור לגלוות חלק מהפתרונות, ניתן לבחור להציג על טעויות (מספרים כפולים באותו שורה, عمودה או קופסה), ניתן לסרוק סודוקו מהעיתון - למשל כזה שהתקשיטם בו ואתם מעוניינים לפתור בסביבה נוחה יותר. בפתרון הסודוקו מתתקבל מושב חיובי מעוד שאמור לעזור למשתמש להמשיך לתרגל את CISEROIN. כל הפיצ'רים הללו מפורטים בהמשך המסמך.

## חלק II

# תיאור תהליכי פיתוח התוכנה

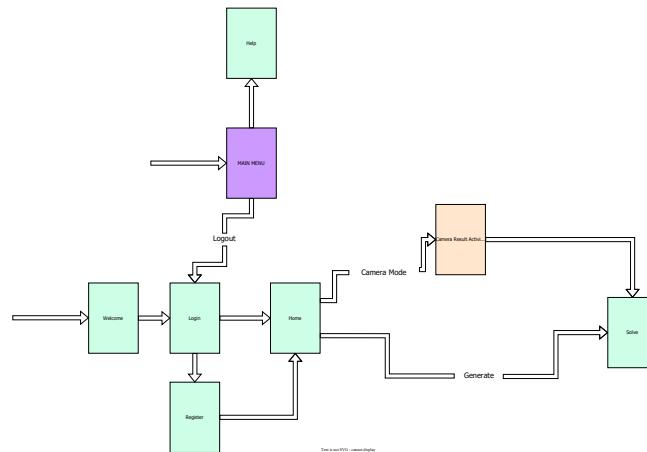
ב חלק זה האתר אספקטים של פיתוח התוכנה עצמו, חלוקם בפירוט, וחלוקם בפחות.



## 1 חווית המשתמש

### 1.1 דיאגרמת פעילות

נראה כאן את הדיאגרמה המתארת את מהלך המעבר מפעולות לפעולות באפליקציה.



בכל שלב ניתן להגעת לתפריט הראשי וממנו לצאת מהאפליקציה.

### 1.2 שפה גרפית ועיצוב

מכיוון שלצערי לא הספקתי לשים דגש מספק כפי שרציתי על העיצוב של משתמש, הוא לא נראה הכי מקצועי שניין. למראות זאת, אני די מוכחה מהתויזאות, והמשק נעים לעין לדעת. הגדרתי את הקובץ styles.xml בו מוגדרת השפה הגרפי של האפליקציה - פרטי נספחים בפרק "תיאור קוד האפליקציה - תיאור קבצי XML".

### 1.3 רמת הפשטה של עקרונות התכנות באפליקציה

ברוח סביבת העבודה הנוכחית, החמה והמחבקת של הפרויקט, העדפתית לא לכלול כלל פרטיה מימוש באפליקציה, והסתתרתי את כולם. המשתמש לא מודע לאיך התחילה ממומשים מאחורי הקלעים, וכל מה שהוא יודע לגבי ההתקשרות עם השרת הוא סיבוב של Progress wheel המורה על המתנה לקבלת התשובה לבקשתו.

## 2 דרישות/מוגבלות להפעלת התוכנה

### 2.1 דרישות מסוימות להפעלת התוכנה

גרסת ה- sdk המינימלית הנדרשת להפעלת האפליקציה היא גרסה 26.

### 2.2 הרשות שצרכיה התוכנה

הישר מקובץ ה- manifest, להלן הרשותות שהאפליקציה דרושת:

```
<uses-feature
    android:name="android.hardware.camera"
    android:required="true" />

<uses-permission
    android:name="android.permission.WRITE_EXTERNAL_STORAGE"
    android:maxSdkVersion="18" />
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.INTERNET" />
```

כלומר, הדרישות הן:

1. אפרשות חיבור לצלמה - על מנת לקבלו את הסודוקו דרך.
2. אפרשות כתיבה וקריאה לזכור - על מנת לשמור את קובץ התמונה באופן זמני לפני שליחתה לשרת.
3. אפרשות גישה לאינטרנט - על מנת לתקשר עם השרת.

### 2.3 פירוט חומרה מיוחדת

החומרה המיוחדת היחידה שנדרשת היא מצלמה. לאחר מכן כמעט בכל טלפון אנדרואיד קיים, זו לא בעיה. אחד השיקולים בגללם בחرتתי בשיטה של חישוב בענו ולא מקומית, הוא זה של חומרה מיוחדת - על טלפון המשתמש באפליקציה יש צורך רק לשלווה בקשetta TensorFlow, במקום לבצע חישוב מורכב עם מודל HTTP.

### 2.4 פירוט סביבת הפיתוח והבדיקות

#### 2.4.1 סביבת הפיתוח

את הפרויקט פיתחתי בסביבת Android Studio Bumblebee | 2021.1.1 Patch 1, לאחר שעברתי מ- Arctic Fox לשנתקליי בשגיאה קריטית (ונוראית!). בעיטה כל הקבצים ששנוראים לפרויקט שהיו לי על המחשב פשוט פשוט להושחתו! כל המאמצים לשחזר את הקבצים מגיט לא צלחו, משום שברגע שהגינו למחשב הושחתו שוב. לאחר חיפוש של שעות על גבי שעות באינטרנט ולא מעת לחץ, מצאתי שאלה באתר תמייקה של JetBrains JetBrands שטיירה בדיק את הבעה הזאת ואת דרך הפתרון שלה. מהה שהבנתי הבעה נבעה מבעיה בתוך Android Studio Arctic Fox אז כדי שלא תחזור על עצמה שוב, עידכני ל- Android Studio Bumblebee. כל הפיתוח נעשה על המחשב האישי שלי בסביבות PyCharm Pro ו- PyCharm. מעט מפתחה השרת (גימורים אחרים) נעשו בסביבת vim.

את העבודה ביצעת בצדדי ל- git (ספקטיף, GitHub). בכל פעם שרציתי להטמע פיצ'ר שימושתי חדש יצרתי ענף (branch) חדש ועובדתי בו. זה ללא ספק עזר לי לנחל את הגרסאות של הקוד בצורה מסודרת וטובה.

את המסמכ הזה אני כותב ב- LaTeX, עורך עבור שפת LaTeX. העורך בהחלט יותר נוח לי מאשר MS Word למשל, ומאפשר הרבה יותר.

**2.4.2 בדיקות והרצות**

את הבדיקות וההרצות ביצעתו בעיקר על **המכשיר האישי שלו** - Samsung Galaxy S10 Lite .Google Pixel 4a .Huawei P20 Lite .  
להלן קטע מההרצות ביצעתו **בamułtowr** מסוג בונסף, הרצתי פעמי אחת את האפליקציה **במכשיר** של חבר

### 3 ספירות חיצונית

לצורך כתיבת הפרויקט נארתי בມגוון של ספירות חיצונית, להלן החלק הרלוונטי מקובץ ה .build.gradle

```
implementation 'com.github.bumptech.glide:glide:4.0.0' // For displaying GIFs
implementation 'nl.dionsegijn:konfetti:1.3.2' // For displaying the confetti

// For internet connection abilities
implementation 'com.squareup.retrofit2:retrofit:2.9.0'
implementation 'com.squareup.retrofit2:converter-gson:2.9.0'
implementation 'com.google.code.gson:gson:2.8.7'
```

## 4 תיכון ובניות מסד הנתונים

בפרק זה, נuarתוי, בין היתר ב- [Silberschatz et al., 2020] ובספר המעלוה של [Android, 2017]

### 4.1 בחירת שירות מסד הנתונים

כדי לתוכנן את מסד הנתונים, הייתה צריכה לקבוע ולתחום גמרי את יכולות האפליקציה, אז לקבוע אילו נתונים עליי לשומר במסד הנתונים, וכייז לשמרו אותם. היו לי מספר אפשרויות לבחור מהן -

1. SQLite, עם .SQL

2. בענן, עם Firebase realtime database

3. בענן, עם אחראית האופציות תחת מטריית השירותים הרחבה שמצויה AWS.

לאחר שהגדרתי את דרישות האישון שלי, הגעת למסקנה שמסד נתונים המקומי הוא האפשרות הטובה ביותר עבורו.

אצין כי בغالל הניסיון הקודם שלי, העדפת מסד נתונים רלוונטי, על פני האפשרויות האחרות. זה גם היה פקטור בבחירה שלי ב- SQLite.

### 4.2 בחירת הנתונים לשימורה

הנתונים שהחלטתי לשמר על המשתמש וחווית השימוש שלו הם כדלקמן:

1. יחס השומר מידע על המשתמש. יחס בשם user.

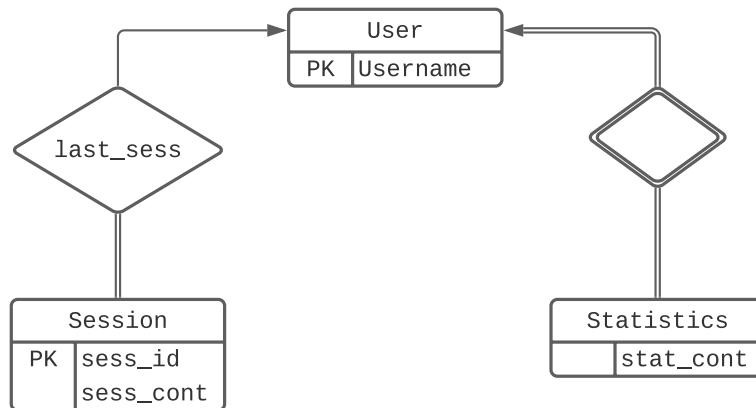
2. יחס השומר מידע על "מפגש" אחד של המשתמש עם האפליקציה - Session. יחס בשם session.

3. יחס השומר מידע על סטטיסטיות המשחק של המשתמש. יחס בשם statistics.

עבור המשתמש, נשמר את שם המשתמש שלו בלבד.

### 4.3 תיקון מסד הנתונים

כעת, נוכל לגשת לבניית דיאגרמת יישוiot-קשרים עבור המסד שתיארנו:



כasher את Statistics **“צגתי כיישות חלשה התלויה ב-”** User.  
נמיר את הדיאגרמה למודול רלציוני ונקבל את רשימת היחסים הבאים:

User(*username*)  
Last\_Sess(*username*, *sess\_cont*)  
Statistics(*username*, *stat\_cont*)

אלו הם היחסים שהגדרתי בפיתוח התוכנה.

### 4.4 התממשקות עם התוכנה

כאמור, את מסד הנתונים מימושי בעזרת SQLite, שהוא מנוע בסיס נתונים ייחסי משובץ.  
חיבור SQLite לאפליקציה היה מאד פשוט, בעיקר בזכות [Android, 2017].  
ההכנסה והשליפה מבוססת הנתונים מבוצעות באמצעות המחלקה SQLiteOpenHelper ומנגנון הד Cursor-ים.

## 5. עיצוב השירות וכתיבת קוד ה- Backend (בקצהה)

זרישיות משרד החינוך לא תאפשר אוטומציה בפרק זה. אולם ניתן בעקבות ה- API של החטבמשקות איתנו.

## 5.1 תכניו ארכיטקטורת השרת

ארכיטקטורת REST היא סגנון פיתוח לתקשורת בין-sistemas (Representational State Transfer) REST, בו בחרתי הוא שרת מסוג SGNON (או ארכיטקטורה) השרתת בו בחרתי הוא שרת מסוג REST.

**משמעות שרת-לקוח** היא שלצורך קיום התקשרות בין מרכז אחד (כموון שלצרכיהם שליו זה הספיק אבל קיימות גם שיטות ליבורן השרטטים), אליו כל הלוקחות מתקשרים.

**משמעות חסר-מצב** היא שהשרות לא שומר שום מידע קונטקטואלי על הלקחות הפוניות אלו או הבקשות, וכך כל בקשה היא למעשה חסרת-קשר.

משמעות מושב משאבים

מתוך, [Representational state transfer, REST], "משאב הוא המושג העיקרי ב- REST. לכל משאב יש ייצוג יכול להיות מספרי או גרפי. הישום מבצע מניפולציות במשאב באמצעות שינויים בייצוג שלו.

לצורך ביצוע אינטראקציה עם המשאב נדרש רק את זהה המשאב ואת הפעולה שהוא מעוניין לבצע. הוא עשוי צורך מידע מוצבם קודמים, נתונים בזיכרון מזמן ורטבים מותוכים כגון שרת פרוקסי. הסיבה לכך היא שככל המידע נשמר בשרת. האפליקציה חייבת להכיר את הפורמט בו מועבר הייצוג של המשאב. פורמט זה הוא בדרך כלל מסמך ב- XML, ב- JSON או ב- HTML.

במקרה של החלטת שענייני עיברתי, ישנו חמישה משאבים:

1. משאב :/solve

המשaab זה מקבל בבקשתו אובייקט JSON בעל שדה יחיד `solved`. השדה מכיל מחרוזת בעלת 81 תווים, כך שכל טו בה מיצג מספר יחיד ב�וץ סודוקו (0 מייצג את המשבצת הריקה). תגובת השרת היא אובייקט JSON בעל שדה `solved`, המכיל את אותו סודוקו בבקשתו, אבל פטור. דוגמה לבקשתה אופיינית:

*Request* : curl -X POST -d

```
'{"unsolved": "0750000034000000100006720000090016000003000050205600049070400000038090000000000000000"}'  
http://127.0.0.1:5000/solve -H "Content-Type: application/json" -H "Auth: AUTH_KEY"
```

**Response :** { "solved": "275184963486935712391672548539421687764398125128567394917243856643859271852716439" }

המשאב הזה מקבל כבקשה אובייקט JSON בעל שדה יחיד `image`. השדה מכיל מחרוזת בפורמט base64 (לקריאה נוספת [Base64, 2022]) שמקודדת תמונה JPEG שצולמה במכשיר הטלפון. התמונה אמורה להכיל תמונה של סודוקו, אותו אנחנו מבקשים לזהות. תגובת השירות היא אובייקט JSON בעל שדה יחיד `board`, את אותו קידוד בעלי 81 תווים שדנו בו קודם המכיל את לוח הסודוקו שבו שארת זיהה מהותנית. דוגמה לבקשת אופיינית:

*Request* : curl -X POST -d @image.json

`http://127.0.0.1:5000/identify -H "Content-Type: application/json" -H "Auth: AUTH_KEY"`  
where image.json is of the form: `{"image": "..."}`

**Response :** {"board": "0750000034000000100006720000090016000003000050205600049070400000038090000000000000000"} {

- .3. משאב `/generate`:  
המשaab זהה מקבל בקשה אחת מרמת הקושי הבאות – Easy, Medium, Hard, Insane – ומחזיר אובייקט JSON בעל שדה יחיד `generated` לוח סודוקו לא פטור אקראי מסוים, מרמת הקושי הנבחרת.
- .4. משאב `/check_connection`:  
גישה למשהab זהה פשט תחזיר אובייקט JSON בעל שדה יחיד `connected` המכיל את הערך `True`. נועד לבדיקת חיבוריות עם האפליקציה.
- .5. משאב `/help`:  
גישה למשהab זהה תחזיר דף HTML עם תיעוד מינימלי על השירות.  
לעוד מידע על REST, עיין ב-[Representational state transfer, 2022].

**Flask API עם בניית 5.2**

אראה להלן את הקוד שנדרש לבניית השרת, שנכתב באמצעות Flask.

```
import os

from PIL import Image
from flask import Flask, request, Response, jsonify, render_template
from flask_restful import Resource, Api
from numpy import asarray

from base_64_handling.B64Utils import b64ToImage
from sudoku_identification.sudoku_image_processing import analyze_image
from sudoku_logic.generating.DifficultyLevel import DifficultyLevel
from sudoku_logic.generating.generate_sudokus import get_sudoku
from sudoku_logic.solving.Sudoku import Sudoku
from sudoku_logic.solving.utils import flatten

AUTH_KEY = os.environ.get('SUDOSOLVE_API_AUTH_KEY')
app = Flask(__name__)
api = Api(app)

class SudokuSolverResource(Resource):
    def post(self):
        try:
            auth = request.headers.get('Auth')
        except LookupError:
            return Response('Authentication needed.\n', status=401)

        if auth != AUTH_KEY:
            return Response('Authentication didn\'t succeed.\n', status=401)

        try:
            req_data = request.get_json()
            unsolved = req_data.get("unsolved")
        except LookupError:
            return Response('The request is invalid.\n', status=400)

        try:
            puzzle = Sudoku(unsolved)
            solved = puzzle.backtracking_search()
        except ValueError:
            return Response('Could not build a Sudoku grid.\n', status=400)

        if not solved:
            return Response('Could not solve the Sudoku.\n', status=400)

        # puzzle.display()
        # print(jsonify({"solved": puzzle.assignment_to_str()}))
        return jsonify({"solved": puzzle.assignment_to_str()})
```

```
class SudokuIdentifierResource(Resource):
    def post(self):
        try:
            auth = request.headers.get('Auth')
        except LookupError:
            return Response('Authentication needed.\n', status=401)

        if auth != AUTH_KEY:
            return Response('Authentication didn\'t succeed.\n', status=401)

        try:
            req_data = request.get_json()
            encoded: str = req_data.get('image')
        except LookupError:
            return Response('The request is invalid.\n', status=400)

        try:
            decoded: Image = asarray(b64ToImage(encoded=encoded))
            rows = analyze_image(image=decoded, debug=False)
            board = '\'.join(map(str, flatten(rows)))
        except Exception as e:
            return Response(str(e), status=400)

        return jsonify({"board": board})

class SudokuGeneratorResource(Resource):
    def get(self):
        try:
            auth = request.headers.get('Auth')
        except LookupError:
            return Response('Authentication needed.\n', status=401)

        if auth != AUTH_KEY:
            print(auth)
            print(AUTH_KEY)
            return Response('Authentication didn\'t succeed.\n', status=401)

        try:
            level = DifficultyLevel(request.args.get('level', ''))
        except ValueError:
            return Response('That is not a correct difficulty level.', status=400)

        board = get_sudoku(level)
        return jsonify({'generated': board})
```

```
api.add_resource(SudokuSolverResource, '/solve')
api.add_resource(SudokuIdentifierResource, '/identify')
api.add_resource(SudokuGeneratorResource, '/generate')

@app.route('/', methods=['GET'])
def helpPage():
    return render_template('help.html')

@app.route('/check_connection', methods=['GET'])
def isConnected():
    return jsonify({'connected': True})

if __name__ == '__main__':
    app.run(debug=True, host='0.0.0.0')
```

### **5.3 הצגת קוד שרת לדוגמה - ייצור הסודוקו**

נראה לי נחמד להראות במסמך זהה מעט מוקד השרת שכבתבי, ואת אלגוריתם למחולל הסודוקו כתבתי我自己 בעצמי, لكن אראה אותו כאן.

הקוד דיבר מסביר את עצמו, ומשמעות המילון `givens` היא מספר המספרים המקוריים שיענהו בסופו של דבר בsolution המוחולל, בכל רמת קושי.

```
import random

from sudoku_logic.generating.DifficultyLevel import DifficultyLevel
from sudoku_logic.solving.Sudoku import Sudoku

givens = {
    DifficultyLevel.EASY: 35,
    DifficultyLevel.MEDIUM: 30,
    DifficultyLevel.HARD: 25,
    DifficultyLevel.INSANE: 15,
}

def _get_initially_filled() -> str:
    generated = '0' * 81
    mutations = list('123456789')
    for _ in range(1, 10):
        # choose random index to change
        ind = random.randint(0, len(generated) - 1)

        # swap out the char at index with a random mutation
        generated = generated[:ind] + random.choice(mutations) + generated[ind + 1:]

    # If the generated puzzle is valid, return it
    # Otherwise, repeat
    puzzle = Sudoku(generated)
    if puzzle.backtracking_search() is None:
        return _get_initially_filled()
    return generated

def get_sudoku(level: DifficultyLevel = DifficultyLevel.INSANE) -> str:
    initial = Sudoku(_get_initially_filled())
    initial.backtracking_search()

    solved = initial.assignment_to_str()
    removed = set()
    for _ in range(0, len(solved) - givens[level]):
        # choose random index to remove
        ind = random.randint(0, len(solved) - 1)
        while ind in removed:
            ind = random.randint(0, len(solved) - 1)
        removed.add(ind)

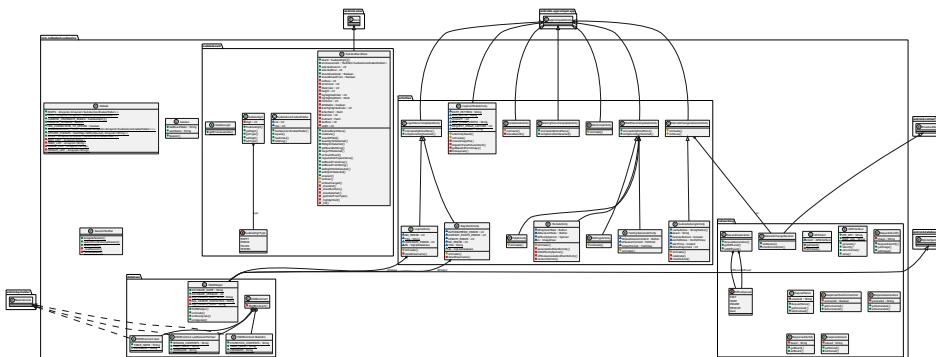
        # remove digit from said position
        solved = solved[:ind] + '0' + solved[ind + 1:]
    return solved
```

## 6 תיאור קוד האפליקציה

פרק זה הוא חלק השני של פרק II והוא כולל בתוכו את תיאור ה- Activities באפליקציה, המחלקות בקוד, קבצי ה- XML וכל מה הקשור להליך פיתוח האפליקציה.

### 6.1 תיאור הקוד - מבט לממעלה

נתחיל להסתכל על הקוד מנוקודת מבט של מעוף הציפור - נסתכל על דיאגרמת UML ונבין את מבנה הקוד בערך. לאחר כך נבחן כל רכיב ורכיב בנפרד.  
הDİאגרמה היא בכוונה בפורמט SVG המאפשר הגדלה שרירותית של התמונה, כך שניתן יהיה לראות את התמונה המלאה בדף ייחיד.



כעת, נתאר את מבנה הקוד הכללי.  
בסיס הקוד מורכב ממספר sub-directories - להלן.

#### 6.1.1 התיקייה com.zvibadash.sudosolve - תיקיית האב של קוד המקור לפרויקט

כוללת את כל תת התיקיות להלן ואת הקובץ Globals.java.

### 6.1.2 התיקייה Activities - ה- `com.zvibadash.sudosolve.activities`

כוללת את הקבצים:

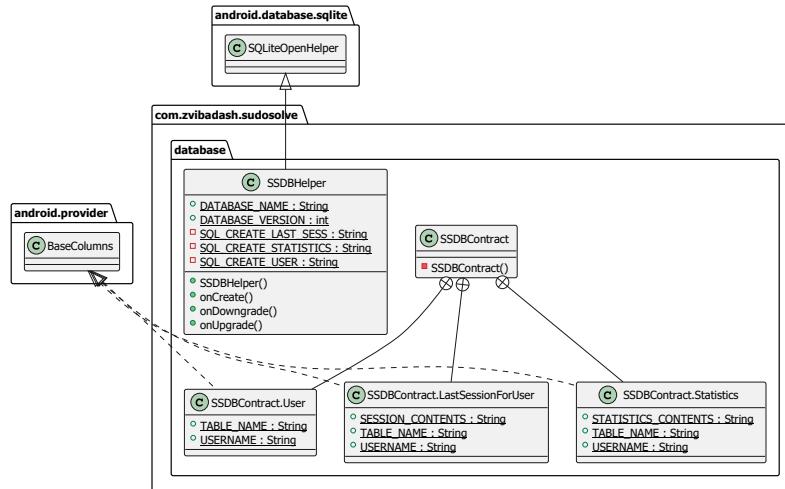
- `CameraModeActivity.java`
- `CelebrateActivity.java`
- `HelpActivity.java`
- `HomeActivity.java`
- `InternetChangeListenerActivity.java`
- `LoginActivity.java`
- `LoginMenuTemplateActivity.java`
- `MainMenuTemplateActivity.java`
- `RegisterActivity.java`
- `SettingsActivity.java`
- `SolvingMenuTemplateActivity.java`
- `SudokuSolvingActivity.java`
- `TestingSessionsActivity.java`
- `WelcomeActivity.java`

הסבר מפורט על כל פעילות יבוא בהמשך.

### 6.1.3 התיקייה db - com.zvibadash.sudosolve.database - ניהול ה-

כוללת את הקבצים:

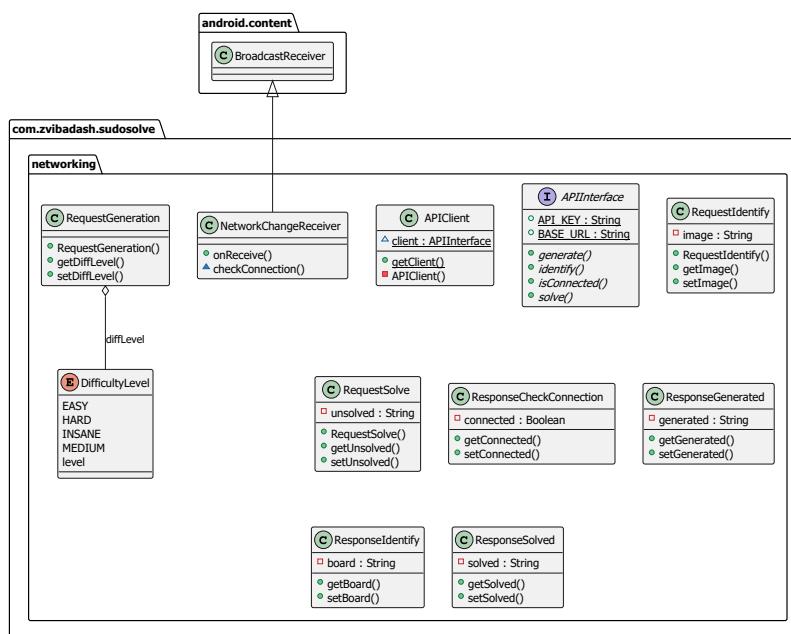
- הקובץ Session.java - מחלקה המכילה את כל הדרישות לשימוש ייחיד באפליקציה, כרגע לא בשימוש.
- הקובץ SessionHandler.java - כנ"ל, לא בשימוש כרגע.
- הקובץ SSDBContract.java - מגדירה את את הטבלאות עבור בסיס הנתונים.
- הקובץ SSDBHelper.java - מחלקה המכילה את הגדרת בסיס הנתונים בעזרת שאלות SQLite.



### 6.1.4 התיקייה - com.zvibadash.sudosolve.networking - תקשורת עם השרת

כוללת את הקבצים:

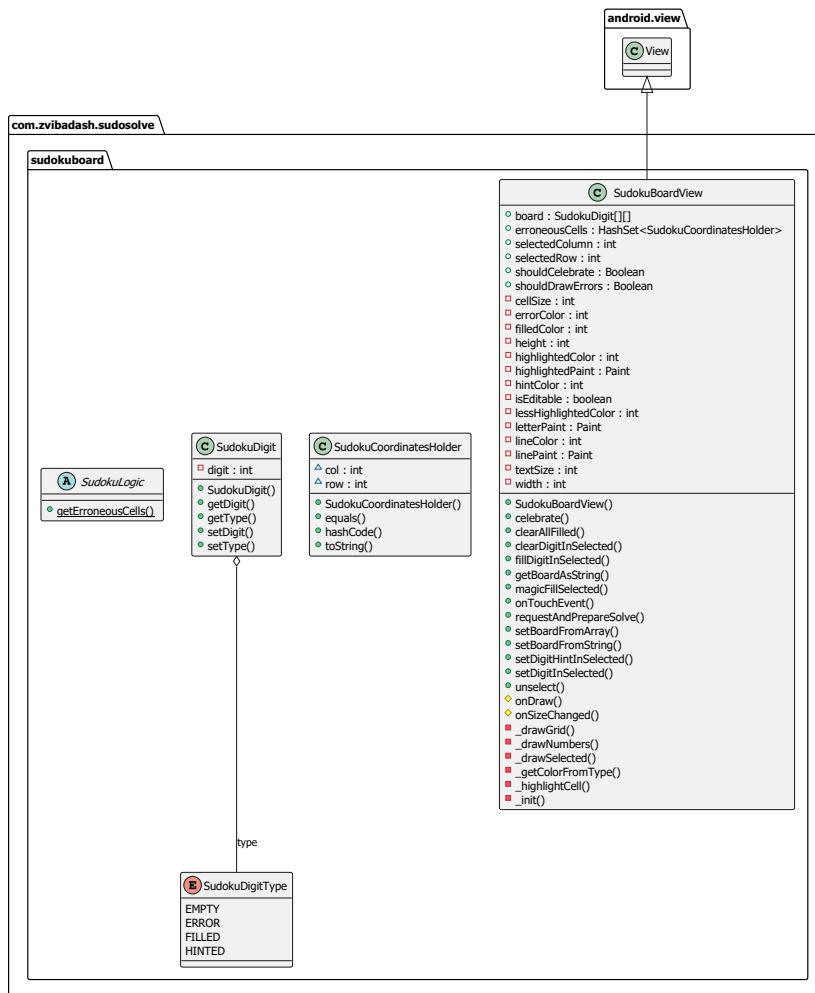
- מחלקות POJO לשימוש הקובץ
  - ResponseSolved.java
  - ResponseIdentify.java
  - RequestSolve.java
  - RequestGeneration.java
  - RequestIdentify.java
  - ResponseGenerated.java
  - ResponseCheckConnection.java
- הקובץ APIInterface.java - מגדיר את הממשק עצמו עם ה- API, ומצבעו לכתובה השרת.
- הקובץ APIClient.java - מגדיר את הלקוח. מחלקת Singletone.
- הקובץ DifficultyLevel.java - מגדיר טיפוס חדש מסוג רמת קושי, מחלוקת enum.
- הקובץ NetworkChangeReceiver.java מגדיר broadcast receiver עבור בדיקה רציפה של קישוריות אינטרנט.



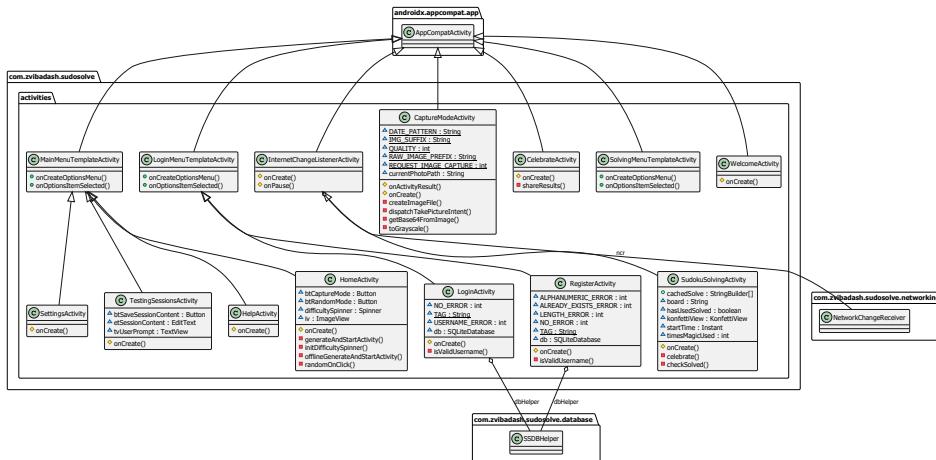
## 6.1.5 התיקייה - לוגיקה ועיצוב לוח הסודוקו בפרויקט com.zvibadash.sudosolve.sudokuboard

כוללת את הקבצים:

- הקובץ SudokuBoardView.java - המחלקה המגדירה את ה- View של הסודוקו. מפורט בהמשך בנפרד.
- הקובץ SudokuCoordinatesHolder.java - מחלקה המגדירה טיפוס גביב (ממשמש את hashCode() חדש, מסוג קורדינטת שלמה (המשמעותו מיקום בלוח סודוקו).
- הקובץ SudokuDigit.java - מחלקה המגדירה טיפוס חדש מסוג ספרה בלוח סודוקו, המורכב ממספר  $\in \{1, 2, \dots, 9\}$ .
- הקובץ SudokuDigitType.java - מגדלר טיפוס חדש מסוג "סוג ספרה" - מחלקת enum.
- הקובץ SudokuLogic.java - מחלקה אבסטרקטית המימושה מעט מהלוגיקה של לוח הסודוקו, למשל חישוב הספרות הכפולות.



## 6.2 תיאור קבצי ה- Activities



בתת-פרק זה, נראה את ה- Activities בפרויקט, ונויבור עליון אחת אחת (בסדר אלפביתי).

### 6.2.1 הפעולות CameraModeActivity

הפעולות זו אחראית לקבל מהמשתמש את תמונה הסודוקו (למשל מהעיתון) ממצלמת המכשיר. הפעולות משתמשות את `onActivityResult`, שמחכה לסיום לכידת התמונה במכשיר ומעבירה אותה בכמה שלבים:

1. שומר את התמונה בתיב `SUDOSOLVE_RAW_YYYY-MM-DD_HH:mm:ss.SSS.jpg` ובשם הקובץ `com.zvibadash.sudosolve.provider`.

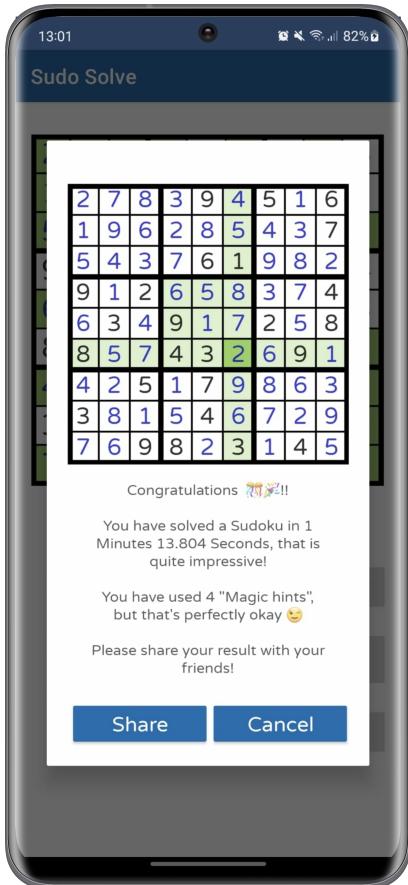
2. הוצאת התמונה מהזיכרון בפורמט `Bitmap`, שמור אותה באובייקט `BitMap`.

3. הפוך את `Bitmap` לשחור ולבן (אין חשיבות לעורכי `chroma`, רק לעורכי `luma`).

4. קודד את התמונה לפורמט `Base64` (מחוזות עצומה, אבל עדין נשמרת בכל זיכרון נדייף למשול).

5. שלח את התמונה לשרת, בכתבוב המשאב `./identify`.

הפעולות מחכה לתשובה זיהוי מהשרת, מקבלת את התשובה, וועברת אותה לפעולות `SudokuSolvingActivity`.

**CelebrateActivity 6.2.2 הפעולות**

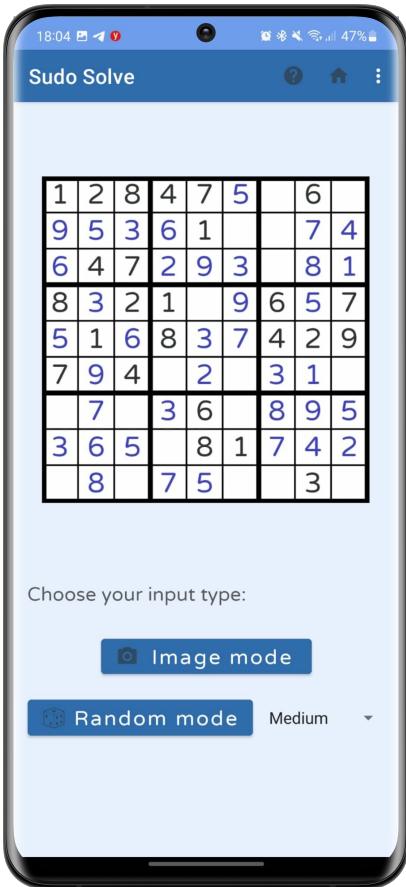
הפעולות זו מיוחדת באופיה, מכיוון שהוא מוגדרת במניפסט כפעולות שנפתחת ב- Dialog Theme של Activity.

היא נפתחת לא במצב של Full screen אלא במצב של דיאלוג רגיל.

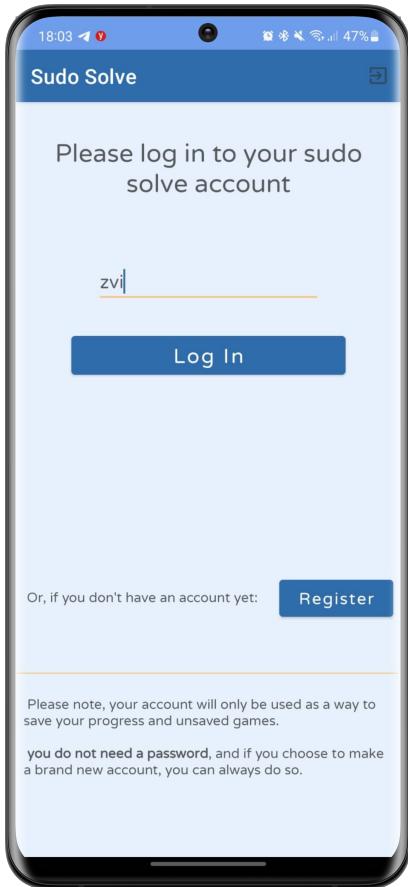
הפעילות נקראת כאשר מסריהם לפטור את הסודוקו וממנה ניתן לשתף למספר מקומות שיתוף (למשל SMS, WhatsApp, Instagram, Gmail, Facebook וכו') בפורמט מיוחד של סיום הסודוקו. ממנה ניתן גם לחזור למסך הבית.

**6.2.3 הפעולות HelpActivity**

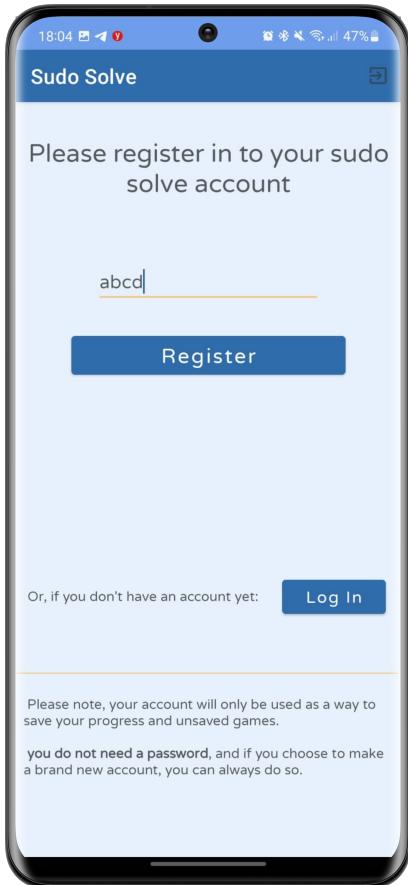
הפעולות הזו היא פעילות עם תוכן סטטי שמסביר על אופן השימוש באפליקציה (המבנה שלו הוא תיבת נגלת שבתוכה ישנו GridLayout המכיל את המידע).  
(הפעולות ממשיכת להיגלל למטה).

**6.2.4 הפעולות HomeActivity**

זהו למעשה המ██ך הראשי באפליקציה, וממנו ניתן  
להגיע ל- CameraModeActivity ו- SudokuSolvingActivity.  
במ██ך יושב ImageView סטטי המציג תמונה נעה  
.Glide בפורמט GIF תוך שימוש בספרייה  
את ה- GIF יצרתי באמצעות פiython, מצילומי מסך  
מתוך האפליקציה!

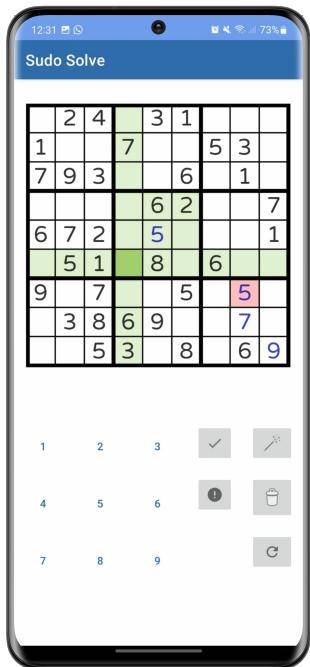
**6.2.5 LoginActivity הפעולות**

פעולות Login סטנדרטיביות, מתחשרת עם המשתמש  
בעזרת חרמות Toasts.

**RegisterActivity 6.2.6 הפעולות**

פעילות Register סטנדרטיבית, מתחברת עם  
המשתמש בעזרת חרמות Toasts.

## 6.2.7 הפעולות SudokuSolvingActivity



זו למעשה הפעולות המרכזיות ביוטר באפליקציה, בה המשתמש אמר לבלוט את רוב זמנו.

**בפעולות מוצב View** מסוג **SudokuBoardView**, תשעה כפתורי מספרים, ועוד חמשה כפתורי פעולה.

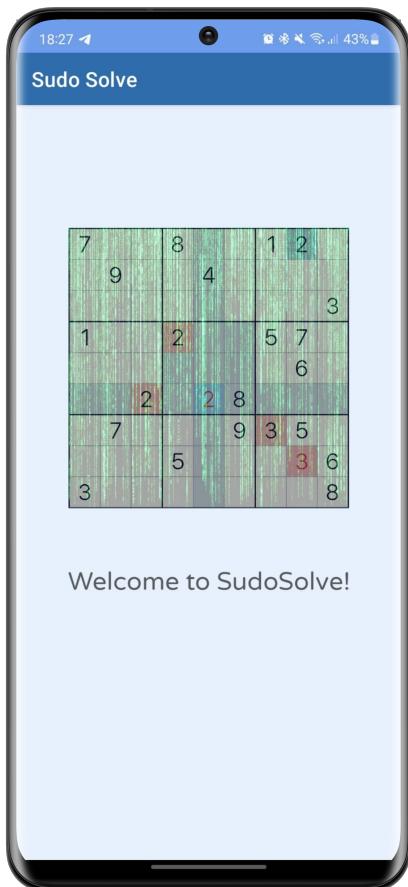
**כפטור השרביט - Magic hint** - להיזה על הכפטור בעת הצבת הסמן בלוח הסודוקו על משבצת מסוימת תגליה את הספרה במיקום זהה.

**כפטור פח האשפה - מחיקה**: להיזה על הכפטור תמחק את הספרה במקומות המסומנים.

**כפטור הגלגל - אתחול**: התחול את הסודוקו מחדש, מחק את כל הספרות שהוזנו.

**כפטור הווי - פתרון**: פתרת הסודוקו - הצגת כל הספרות.

**כפטור סימון הקריאה - בדיקה**: להיזה על הכפטור תציג את כל הספרות החזרות על עצמן באותה שורה, عمودה או קופסה במשך 1.5 שניות בצבע אדום.

**6.2.8 הפעולות WelcomeActivity**

אקטיביטי בעלת תפקיד מנון ייחסי - משמשת כמסך עבורי האפליקציה, וממלאת שני תפקידיים:

**הציגת תמונה** כמסך המתנה לטעינת האפליקציה.

**בדיקות קישוריות עם השרת** - ממבצע ניסיון שליחת בקשה לשרת, אם הניסיון צלח ויישנה תקשורת נדלק דגל .Globals.HAS\_CONNECTION\_TO\_SERVER אחרת, הדגל כבוי ובמהלך מחזור החימں הנוכחי של האפליקציה לא יתבצעו ניסיונות יצירת בקשות לשרת.

**6.3 תיאור קבצי ה- XML וקבצי****6.3.1 קובץ ה- AndroidManifest.xml**

להלן הקובץ:

```

<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.zvibadash.sudosolve">

    <uses-feature
        android:name="android.hardware.camera"
        android:required="true" />

    <uses-permission
        android:name="android.permission.WRITE_EXTERNAL_STORAGE"
        android:maxSdkVersion="18" />
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.INTERNET" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:networkSecurityConfig="@xml/network_security_config"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.SudoSolve">

        <activity
            android:name=".activities.CelebrateActivity"
            android:excludeFromRecents="true"
            android:theme="@style/AppDialogTheme"
            android:exported="false" />
            ...
        <activity
            android:name=".activities.WelcomeActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <meta-data
            android:name="preload_fonts"
            android:resource="@array/preloaded_fonts" />

        <provider
            android:name="androidx.core.content.FileProvider"
            android:authorities="com.zvibadash.sudosolve.fileprovider"
            android:exported="false"
            android:grantUriPermissions="true">
            <meta-data
                android:name="android.support.FILE_PROVIDER_PATHS"
                android:resource="@xml/file_paths" />
        </provider>
    </application>

</manifest>

```

כאשר החלק שמוופיע ב ... הוא פשטן תיאור משועם של הפעולות באפליקציה (בכוונה השמטה את הפעולות בהן אין XML Attributes מיוחדים).

הקובץ לא מעניין מדי ומכל בעיקר הגדרות של activities והרשאות.

**login\_menu.xml הקובץ 6.3.2**

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:app="http://schemas.android.com/apk/res-auto"
      xmlns:android="http://schemas.android.com/apk/res/android">

    <item
        android:id="@+id/opExit"
        android:icon="@drawable/ic_exit_action"
        android:title="Exit"
        app:showAsAction="ifRoom" />
</menu>
```

**main\_menu.xml הקובץ 6.3.3**

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:app="http://schemas.android.com/apk/res-auto"
      xmlns:android="http://schemas.android.com/apk/res/android">

    <item android:id="@+id/opHelp"
          android:icon="@drawable/ic_help_action"
          android:title="Help"
          app:showAsAction="ifRoom" />

    <item android:id="@+id/opHome"
          android:icon="@drawable/ic_home_action"
          android:title="Go to home"
          app:showAsAction="ifRoom" />

    <item android:id="@+id/opSettings"
          android:icon="@drawable/ic_settings_action"
          android:title="Settings" />

    <item android:id="@+id/opLogout"
          android:icon="@drawable/ic_logout_action"
          android:title="Logout" />
</menu>
```

**attrs.xml הקובץ 6.3.4**

הקובץ זהה מגדיר את התכונות של ה **SudokuBoardView** - Custom view שהגדרתי

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <declare-styleable name="SudokuBoardView">
        <attr name="errorColor" format="color"/>
        <attr name="hintColor" format="color"/>
        <attr name="filledColor" format="color"/>
        <attr name="lineColor" format="color"/>
        <attr name="highlightedColor" format="color"/>
        <attr name="isEditable" format="boolean"/>
        <attr name="cellTextSize" format="integer"/>
    </declare-styleable>
</resources>
```

**הקובץ styles.xml 6.3.5**

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <style name="BasicButtonStyle">
        <item name="android:layout_width">wrap_content</item>
        <item name="android:layout_height">wrap_content</item>
        <item name="android:backgroundTint">@color/Green_Blue</item>
        <item name="android:fontFamily">@font/varela_round_regular</item>
        <item name="android:includeFontPadding">true</item>
        <item name="android:textAllCaps">false</item>
        <item name="android:textSize">20sp</item>
    </style>

    <style name="BasicHeadingStyle">
        <item name="android:fontFamily">@font/varela_round_regular</item>
        <item name="android:gravity">center</item>
        <item name="android:textAlignment">center</item>
        <item name="android:textSize">13pt</item>
        <item name="android:textColor">@color/textGray</item>
    </style>

    <style name="BasicUsernameEditTextStyle">
        <item name="android:layout_width">wrap_content</item>
        <item name="android:layout_height">wrap_content</item>
        <item name="android:autofillHints">username</item>
        <item name="android:backgroundTint">@color/Gold_Crayola</item>
        <item name="android:ems">10</item>
        <item name="android:fontFamily">@font/varela_round_regular</item>
        <item name="android:inputType">textPersonName</item>
        <item name="android:minHeight">48dp</item>
        <item name="android:hint">Username...</item>
        <item name="android:textColorHint">@color/textGray</item>
        <item name="android:textColor">@color/textGray</item>
    </style>

    <style name="BasicSwitchStyle">
        <item name="android:layout_width">wrap_content</item>
        <item name="android:layout_height">wrap_content</item>
        <item name="android:buttonTint">@color/Green_Blue</item>
        <item name="android:foregroundTint">@color/Gold_Crayola</item>
        <item name="android:minHeight">48dp</item>
    </style>

    <style name="BasicTextViewStyle">
        <item name="android:fontFamily">@font/varela_round_regular</item>
        <item name="android:textSize">7pt</item>
        <item name="android:gravity">center</item>
        <item name="android:textAlignment">center</item>
        <item name="android:textColor">@color/textGray</item>
    </style>

    <style name="AlertDialogTheme" parent="Theme.AppCompat.Light.Dialog">
        <item name="windowActionBar">false</item>
        <item name="android:windowNoTitle">true</item>
    </style>
</resources>
```

**הקובץ themes.xml 6.3.6**

```

<resources xmlns:tools="http://schemas.android.com/tools">
    <!-- Base application theme. -->
    <style name="Theme.SudoSolve" parent="Theme.MaterialComponents.Light.DarkActionBar">
        <!-- Primary brand color. -->
        <item name="colorPrimary">@color/Green_Blue</item>
        <item name="colorPrimaryVariant">@color/Cornflower_Blue</item>
        <item name="colorOnPrimary">@color/white</item>
        <!-- Secondary brand color. -->
        <item name="colorSecondary">@color/Green_Blue</item>
        <item name="colorSecondaryVariant">@color/Cornflower_Blue</item>
        <item name="colorOnSecondary">@color/black</item>
        <!-- Status bar color. -->
        <item name="android:statusBarColor" tools:targetApi="l">?attr/colorPrimaryVariant</item>
        <!-- Customize your theme here. -->
    </style>
    <style name="Theme.SudoSolve.NoActionBar">
        <item name="windowActionBar">false</item>
        <item name="windowNoTitle">true</item>
    </style>
    <style name="Theme.SudoSolve.AppBarOverlay" parent="ThemeOverlay.AppCompat.Dark.ActionBar" />
    <style name="Theme.SudoSolve.PopupOverlay" parent="ThemeOverlay.AppCompat.Light" />
</resources>

```

**הקובץ colors.xml 6.3.7**

```

<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="purple_200">#FFBB86FC</color>
    <color name="purple_500">#FF6200EE</color>
    <color name="purple_700">#FF3700B3</color>
    <color name="teal_200">#FF03DAC5</color>
    <color name="teal_700">#FF018786</color>
    <color name="black">#FF000000</color>
    <color name="white">#FFFFFFFF</color>

    <!--
        This are my colors, defined by a color scheme designed by colors.co
    -->
    <color name="textGray">#A0000000</color>
    <color name="Green_Blue">#306BAC</color>
    <color name="Cornflower_Blue">#6F9CEB</color>
    <color name="White_Baby_Blue_Eyes">#E9F1FF</color>
    <color name="Gold_Craola">#F9C784</color>
</resources>

```

## 6.4 הקובץ SudokuBoardView.java

הקובץ הזה מגדיר בתוכו מחלקה הירושת מהמחלקה View, המחלקה מימושת את כל שיטות (מתודות) הציר והрендור הנדרשות ממנה, ואת כל המשטמע מכך.  
להלן השיטות הציבוריות של המחלקה:

```
public void setDigitInSelected(int d) {  
    ...  
}  
  
public void setDigitHintInSelected(int d) {  
    ...  
}  
  
public void fillDigitInSelected(int d) {  
    ...  
}  
  
public void unselect() {  
    ...  
}  
  
public String getBoardAsString() {  
    ...  
}  
  
public void clearDigitInSelected() {  
    ...  
}  
  
public void setBoardFromString(String newBoard) {  
    ...  
}  
  
public void setBoardFromArray(SudokuDigit[][] newBoard) {  
    ...  
}  
  
public void clearAllFilled() {  
    ...  
}  
  
public void magicFillSelected(Context context, StringBuilder[] cachedSolve) {  
    ...  
}  
  
public void requestAndPrepareSolve(Context context, StringBuilder[] cachedSolve) {  
    ...  
}  
  
public void celebrate() {  
    ...  
}
```

## 7 תיאור המענה על דרישות פרויקט הגמר

נראה לי ראוי לסכם בקצרה את התוכנות עליהן אני מונען בפרויקט, לפי דרישות משרד החינוך.

<p>שימוש בתפריטים - ✓ (למשל, התפריט המאפשר Logout) (Logout)</p> <p>תיבות דו-שיח - ✓ (למשל, אישור התחלת חדש)</p> <p>פקדים ומאזינים - ✓ (קיימים בכל פעילות)</p>	<p>התוכנית מהוות אפליקציה לטלפונים חכמים ומאפשרות שימוש מלא של כל דרישות הפרויקט. שימוש בתפריטים, תיבות דו-שיח. שימוש באירועים (פקדים ומאזינים, מקשים, מצבייע)</p>
<p>אין לי דרך אובייקטיבית לומר האם הדרישות הללו מתקיימות, אבל אני מאמין שכן.</p>	<p>הפרויקט מהוות תוכנית אינטראקטיבית המנוהלת על ידי משק גרפי למשתמש, תוך התאמת לדרישות היישום. מוגנות (הנדסת אנוש, אסתטיקה, נוחות שימוש, שימושיות, משק ברור וכו')</p>
<p>סוגנו התוכנות הוא בדיקת סוגנו אליו אני רגיל, והוא נבדק על ידי ה- linter של Android Studio (über Kod Java) ונכתב בכללי PEP8 עבור קוד השרת.</p> <p>שימוש בתוכנות מונחה עצמים - יש שימוש בירושא בפרויקט (למשל, הפעולות מונחה עצמים MainMenuTemplateActivity ממנה יוצרים מספר פעילות)</p>	<p>שימוש נכון בתוכנות מונחה עצמים (דגש על ירושה, הכללה וכו'). ובסוגנו התוכנות (בחירה משתנים, קריאות, תיעוד, חלוקה לפעולות וכו')</p>
<p>שימוש ב- Activity ✓ שימוש ב- Intent ✓</p> <p>שימוש ב- Service - ✓ - ✓ לא נכתב על ידי אלא הוא מובנה בספרייה retrofit2.</p> <p>שימוש ב- Broadcast Receiver - ✓ - מותבצעת בדיקה רציפה האם החיבור לאינטרנט קיים, כבידה מקדימה לחיבוריות עם השרת.</p>	<p>שימוש באבני היסודות Service, Intent, Activity ואחד Content Provider או Broadcast Receiver מהשניים</p>
<p>אחסון וטיפול נתונים (פחות שמירה ושליפה) לפחות בדרך אחת ✓ - קיים על ידי SQLite</p>	

## חלק III

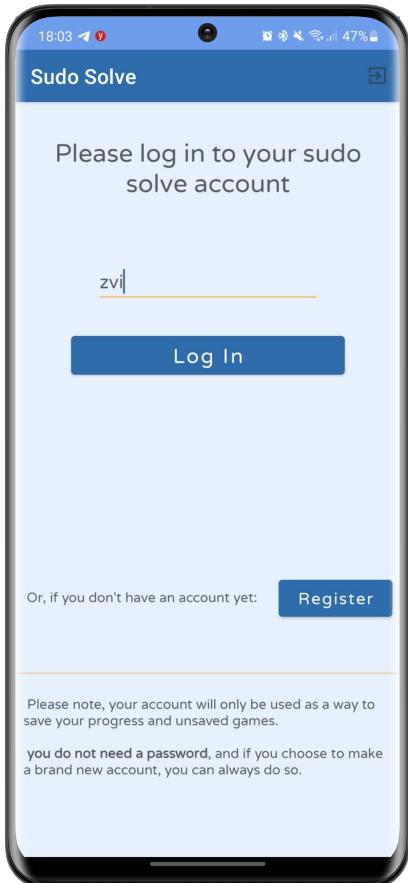
# מדריך למשתמש

בחלק זה אתן מדריך למשתמש הקצה, הדורש שום רקע בפתרונות.



## 0.1 פתיחת חשבון

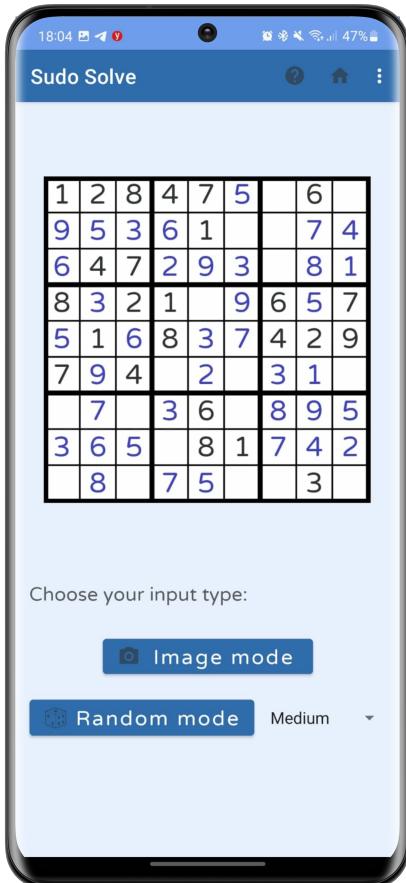
מיד ברגע להשתמש באפליקציה, תיתקל במסך הבא שיורה לך להירשם עם שם המשתמש שלך.



אם כבר יש לך שם משתמש (למשל zvi), המשך אליו. אחרת, עקוב אחר ההוראות ולחץ על כפתור ה-.register. במסך זהה, בחר את שם המשתמש שלך, ולחץ register.

**0.2 מסך הבית**

במסך הבית



תוכן לבחור: האפשרות הראשונה היא לסרוק סודוקו מתוך המצלמה (Image mode) והאפשרות השנייה היא לפתרו סודוקו אקראי.

**0.3 מצלמה**

בעת לחיצה על הcptor, תיפתח מוליך המצלמה, וודא שאתה מצלם רק את הסודוקו, וודא שאין מסיחי דעת בפריים המצלום וודא שמסגרת הסודוקו נמצאת כולה בפראיים ולא חתוכה ובעיקר הקפד על איקות כמה שיותר גבוהה של התמונה (כמה שפחות תזוזות בזמן הצילום).

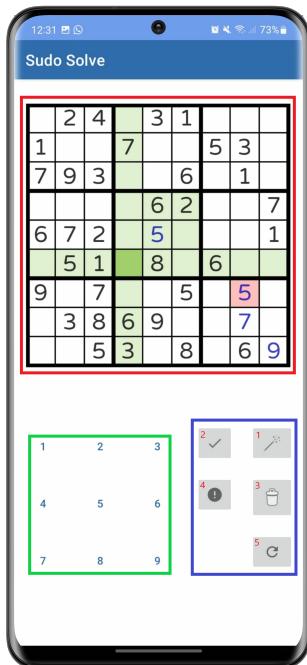
לאחר מספר שניות של זיהוי, יפתח לך מסך הפתרון. אם האפליקציה מורה לך לאלים שנייה, צלם שניית.

**0.4 מצב אקראי**

בחר רמת קושי מהבות Easy, Medium, Hard, Insane והמשך לשלב הבא.

**0.5 מסך הפתרון**

במסך זה מספר אלמנטים:



- לוח הסודוקו.

- מקלדת הספרות.

- מקלדת הפעולות.

**לוח הסודוקו** הוא לחץ, וניתן לנווט בו באמצעות החלקת האצבע עליו. המיקום הנבחר כרגע בסודוקו יודגש בצבע יירוק כהה, ושכניו בצבע יירוק בהיר. בעת לחיצה על **מקלדת הספרות**, תוכב הספרה עלייה לחצת במיקום המסומן ב**לוח הסודוקו**. ב**מקלדת הפעולות** מספר פעולות:

1. לחיצה על השרביט, תמלא את המיקום המסומן בלוח הסודוקו בספרה שאמורה להיות במיקום זה בסוף הפתרון - תכוונה זו יכולה מאד לעזור כאשר יש לך לבטים בקשר לתא מסוים, או שאתה מחפש蹊ור דרך קל של יקטע את קו מחשבך על חלק אחר בלוח הסודוקו.

2. לחיצה על הווי ת מלא את לוח הסודוקו לגמרי, ותפתור אותו.

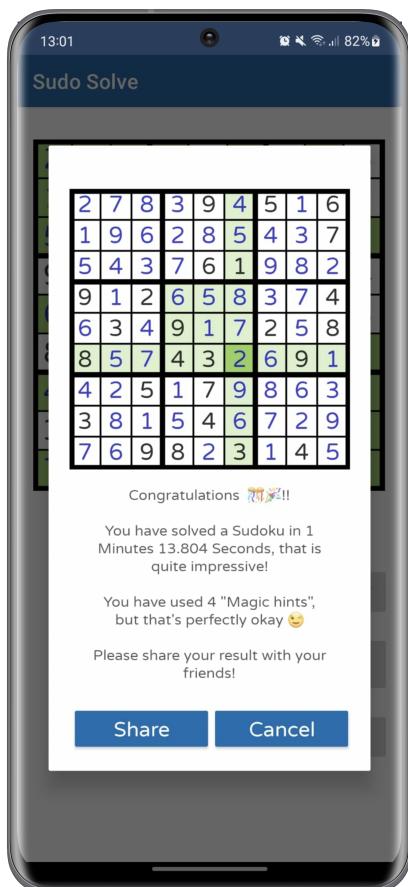
3. לחיצה על כפטור המהיקה, תמחק את הספרה שהזנת במיקום המסומן.

4. לחיצה על הכפטור הזה תdagish לך למשל 1.5 שניות את כל המספרים שהזנת במקומות לא אפשריים (כלומר מספרים שחזרים על עצם בשורה, عمودה או קופסה) בצבע אדום (ראה 5 בתמונה).

5. לחיצה על כפטור הריענון, תתחל מחדש את הסודוקו על מנת שתוכל לפתור אותו מחדש.

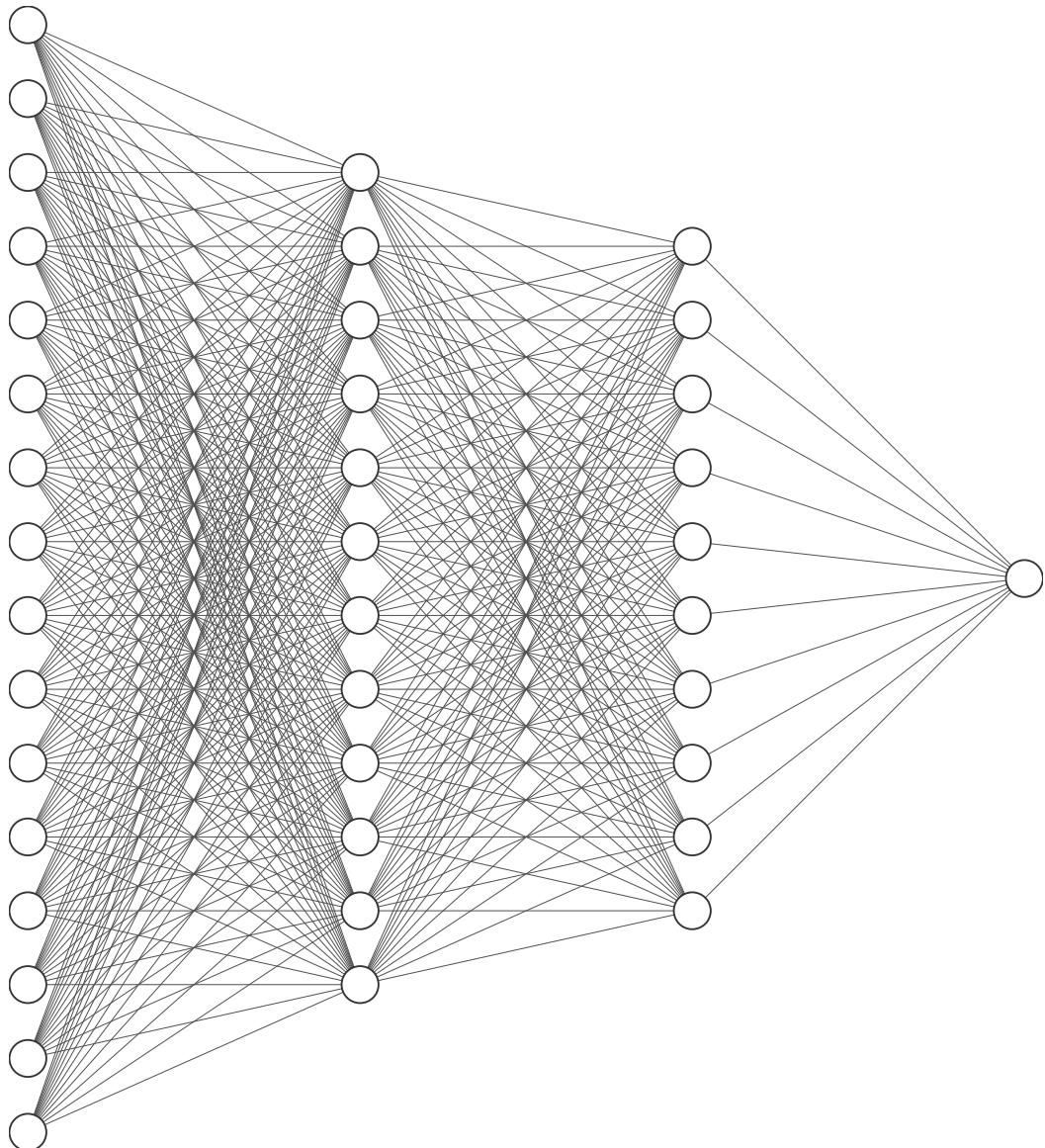
## 0.6 הצלחת לפתור, יפה!

כעת יוצגו בפניכך פירורי קונספטוי, וישמעו זיקוקים!  
תישלח למסך בו תוכל לצפות בלוח הסודוקו שפתרת (הוא אינטראקטיבי, נסה לנעת בו!), ולשתף את התוצאה שלך (זמן  
פתרונות + מספר רמזים שהשתמשה בהם) עם חבריך!



**חלק IV****רקע תיאורטי**

בחלק זה אסקור את היסודות התיאורטיים מאחוריו אלגוריתמים וריעונות מרכזיים המופיעים בפרויקט, בין היתר למידת מכונה ובעיות NP-שלמות.



## 1 על ההיסטוריה של חידת הסודוקו

אצין כי אין מומחה להיסטוריה, וכי הסתמכתי בעיקר על הויקיפדיה האנגלית כדי לסקור פרק זה, אבל היה נראה לי נכון לכלל בכל זאת פרק קצר בעניין.

ניתן להתחקות אחר מקור בעיתת הסודוקו עד לאונרד אוילר<sup>1</sup>, במאה ה-18. הוא חקר את התוכנות של בעייה דומה - בעיתת ריבוע הקסם; חידה בה צריך למלא מטריצה ריקה מסדר  $n \times n$  במספרים מד- $\{1, 2, \dots, n^2\}$  כך שסכום כל שורה, عمودה ואלכסון יהיה זהה. ככל הנראה, החידה עברה מספר גלגולים עד שהגיעה לצורה בה אנו מכירים אותה כיום - ה-数字は独身に限る (בעתיק): "סואזי ווא דוקושין ני קאגירו" (משמעותו "היותו בלתי נשוי") (משמעות השם בצורה יותר מופשטת היא "היותו בודד", מה שרומי ליחודיות המספרים בכל שורה, عمودה וקוופסה).

החידה רוחה ביפן בשנת 1984 ונותרה בחזקת היפנים עד שנת 2004 - בשנה אז ניו זילנדי בשם ויין גולד, בעבר שופט בהונג קונג, פיתח תוכנית מחשב המייצרת תשਬצים אלה, ובשנת 2004 החל לפקסם לעיתון 'hetymist' הבריטי, אשר החל לפרסם במדור יומי קבוע.

כיום החידות מפורסמות ביפן ובריטניה גם בעיתונים יומיים. גם העיתונים האוסטרליים 'האוסטרליין' וה'סידני מורנינג הרלד' החלו בפרסום מדורים קבועים. בשנת 2005 גם עיתונים ישראלים החלו לפרסם סודוקו על בסיס קבוע.

אם כן, ראיינו כי מקורות חידת הסודוקו כפי שהוא מכירם אותה כיום מושרשים בעבודתו של אוילר וכי חידת הסודוקו של ימינו מגיעה מיפן.

---

<sup>1</sup>לאונרד אוילר (1707-1783) היה מתמטיקאי ופיזיקאי שוודי, מהמתמטיקאים הפורים והמוסכרים ביותר שידעו האנושות.

## 2 על בעיות NP-שלמות

### 2.1 מחלקות סיבוכיות

בתחום מדעי המחשב התיאורטיים, דנים בשאלות כגון "האם ניתן לחשב את זה?" (תורת החישוביות) ו- "האם ניתן לחשב את זה בזמן סביר?" (תורת הסיבוכיות) בקשר לעוויות מסוימות. את הפורמליזם של השאלות אלה מבאים באמצעות מודל חישובי מופשט - מכונת טיריניג<sup>2</sup>. אותן יונין במיוחדם של העוויות והסיבוכיות, וכן נטען עיקר בשאלת השניה.

בקשר לאוסף בעוויות מסוימות, להן סיבוכיות דומה, נהוג להגדיר מחלקה סיבוכיות באופן הבא<sup>3</sup>: נאמר כי אוסף בעוויות ניתנות להכרעה על ידי מכונה  $M$  תוך שימוש בזמן  $(n)^f \mathcal{O}$  כאשר  $n$  הוא גודל הקלט, מהוות מחלקה סיבוכיות.

בקשר של הבעיה שלנו נגדיר בתת-הפרקם להן מספר מחלקות סיבוכיות, שיעזרו לנו להבין איפה בעיה מסוימת נמצאת בעולם הזה, ומה המשמעות של מיקום זה.

### 2.2 המחלקה P

מחלקה הסיבוכיות מהפשות ביותר ביותר היא המחלקה P. משמעותו שהיא מילאה Polynomial. מושג זה מרכיבת מכל בעוויות שניתנות לפתרון בזמן  $(n)^f \mathcal{O}$  כאשר  $f(n)$  פולינומיאלית בגודל הקלט,  $n$ .

בעיה לדוגמה במחלקה זו היא למשל בעיה החיפוש במערך, בעית מיצאת עצ פורש מינימלי בגרף ומיצאת הד  $\text{gcd}$  של שני מספרים.

בעיה לדוגמה שאינה נמצאת ב- P היא בעית העזירה.

### 2.3 המחלקה NP

מחלקה הסיבוכיות השניה עליה נדבר היא המחלקה NP. משמעותו שהיא מילאה Non-deterministic Polynomial. מושג זה מרכיבת מכל בעוויות שבהן ניתן פתרון אפשרי למופיע של הבעיה, ניתן לוודא שהוא אכן פתרון בזמן  $(n)^f \mathcal{O}$  כאשר  $f(n)$  פולינומיאלית בגודל הקלט,  $n$ . הגדרה זו שקופה להגדירה אחרת המתאימה לתבנית ההגדירה לעיל, אבל אין זה הכרחי לנו.

כלומר, NP הוא אוסף בעוויות שנitinן לוודא עבורן פתרון פולינומיאלי. ברור לנו כי  $P \subseteq NP$  מכיוון ש כדי לוודא האם פתרון למופיע מסוים של הבעיה הוא נכון ניתן ניתן פשטוט לפתרור את הבעיה בזמן פולינומי ולברר אם הפתרונות זרים.

נגיד בנוסח עוד מחלקה - מחלקה NPC (או NP Complete): נאמר כי בעיה היא ב- NP אם היא ב- NP וגם ניתן לפתור בעורה כל בעיה אחרת ב- NP.

עבור רוב בעוויות, יחסית קשה להראות כי היא NPC, אבל את הדוגמה הרבים בשנת 1927 ריצ'רד קארפ שהראה עבור 12 בעיות חשובות במדעי המחשב כי היא NP-שלמות (בהתבסס על כך ש- SAT היא NP-שלמה). מאמר זה הצבע על כך שיש טעם בהגדרת מחלקה סיבוכיות חדשה, NPC, של אוסף בעוויות הד NP-שלמות.

### 2.4 שאלת מיליון דולר, $P = ? NP$

אחד משבע בעוויות של מכון קלוי (שבע בעיות פתוחות במתמטיקה, ש"הונח פרס על ראש" וחותמו מיליון דולר וכבוד אדיר לפוטר אותה מהן) היא בעית  $P = ? NP$ . אם נחזור להגדרות ששללוינו ניתן את הדריך להבנת השאלה הזאת נוכל לנ释 אותה מחדש: "האם בהינתן בעיה שאני יודע לוודא בעורה פתרון בזמן יעיל אני יכול גם לפתור בעיה בזמן יעיל?". השאלה זו מעסיקה מדעני מחשב ומתמטיקים שונים לගמרי של המתמטיקה כבר עשרות שנים.

כיום הדעה הרווחת היא שדווקא מתקיים  $NP \neq P$ , אבל טענה זו לא הוכחה עדין.

<sup>2</sup>Alan Turing היה מתמטיקאי בריטי, ממניחי היסודות למדעי המחשב. הגע להישגים יוצאי דופן בצד התאורטי ובצד המעשי של מדעי המחשב. בצד התאורטי הנחיל טיריניג לעולם (בין היתר) את מכונת טיריניג - מודל מופשט למכונת חישוב אוניברסלית, אשר מותאר באמצעות אופן פעולה של המחשב. טיריניג נחשב אבי מדעי המחשב והבינה המלאכותית.

<sup>3</sup>ההגדרה האמיתית כלילת הרבה יותר, אבל הגדרה זו תספק לנו בהצלת.

## 2.5 מה הקשר לסודוקו?

את ההקדמה המיגעת למדи הזו קלلتיב שبيل (אך גם מושם יופי בעית  $NP \stackrel{?}{=} P$ ) הדיון בעיתת הסודוקו: אחרי הכל, מדובר בעיה חשובה, ונוכל לשאול באיזו מחלוקת סיבוכיות היא נמצאת. לפי [Yato and Seta, 2003] התשובה היא שבעית הסודוקו מסדר  $n \times n$  היא  $NP$  שלמה.

כלומר, ניתן לוודא פתרון לסודוקו בזמן פולינומי (כמובן) אבל הרבה יותר חשוב לכך - ניתן לבצע רדוקציה פולינומית של כל בעיה בעיתת הסודוקו! (כלומר, ניתן לפתור כל בעיה ב-  $NP$  על ידי פתרון של סודוקו  $n \times n$  מסוים).

מהעובדה הזו נסיק את המסקנה העיקרית הבאה:

- אם  $NP \neq P$ , זהו כנראה המצב - אין שום תקווה למצוא אלגוריתם פולינומיאלי במובן החזק לפתרת סודוקו כלל-משמעות  $n \times n$ .

נעוזב את הפרק הזה בנימה מעט פסימית, אבל כשניבור בפרק הבא נסביר למה זה ממש לא מפחיד אותנו!

### 3 על תכנות אילוצים ו-CSPs

בפרק זה אתאר את השיטות הבסיסיות בתכנות אילוצים, המהוות את אלגוריתם פתרת הסודוקו אותו מימושי עבור הפרויקט.

#### 3.1 מהי בעיית סיפוק אילוצים (CSP)

נעזר בספרם של [Russell and Norvig, 2020] כדי ללמוד על בעיית סיפוק אילוצים ולתאר אותו כאן.  
באופן פורמלי, בעיית סיפוק אילוצים או (*Constraint Satisfaction Problem*) היא שלשה סדרה  $\langle \mathcal{X}, \mathcal{D}, \mathcal{C} \rangle$  המוגדרת כך:

- $\mathcal{X} = \{X_1, \dots, X_n\}$  היא קבוצת משתנים,
- $\mathcal{D} = \{D_1, \dots, D_n\}$  היא קבוצת התחומים המתאימים למשתנים,
- $\mathcal{C} = \{C_1, \dots, C_m\}$  היא קבוצת אילוצים. נגדיר אילוץ באופן הבא:

- אילוץ  $C_j$  הוא יחס  $k$ -ארי  $C_j \subseteq \mathcal{D}^k$

השמה לביקורת סיפוק אילוצים היא פונקציה שעבור חלק מהמשתנים  $X_i$  מחזירה איברים בתחוםים המתאימים שלהם, נאמר כי ההשמה עקciית אם היא לא מפירה אף אילוץ.

נאמר כי ההשמה שלמה אם היא משימה כל משתנה.

נאמר על השמה כי היא פטרו או שהיא פותרת את הבעיה, אם היא שלמה ועקבית.

נגדיר כעת את בעיית הסודוקו מסדר  $n \times n$  כבעיית סיפוק אילוצים.

קבוצת המשתנים תהיה  $\mathcal{X} = \{X_{1,1}, X_{1,2}, \dots, X_{1,n}, X_{2,1}, \dots, X_{n,n}\}$ .  
קבוצת התחומים תהיה  $\mathcal{D} = \{D_{1,1}, D_{1,2}, \dots, D_{1,n}, D_{2,1}, \dots, D_{n,n}\}$  כאשר

$$D_{i,j} = \begin{cases} \{v\} & , X_{i,j} \text{ Already has a value } v \text{ in the Sudoku} \\ \{1, 2, 3, \dots, n\} & , \text{Otherwise} \end{cases}$$

בקבוצת האילוצים יהיו בדיק  $3n$  אילוצים -  $n$  אילוצים לכל השורות,  $n$  אילוצים לכל העמודות ו-  $n$  אילוצים לכל הקופסאות. האילוצים יוגדרו כך:

$$\mathcal{C} = \mathcal{C}_{\text{rows}} \cup \mathcal{C}_{\text{columns}} \cup \mathcal{C}_{\text{boxes}}$$

כאשר:

$$\mathcal{C}_{\text{rows}} = \left\{ \begin{array}{l} \text{AllDiff}(X_{11}, X_{12}, X_{13}, \dots, X_{1n}), \\ \text{AllDiff}(X_{21}, X_{22}, X_{23}, \dots, X_{2n}), \\ \vdots \\ \text{AllDiff}(X_{n1}, X_{n2}, X_{n3}, \dots, X_{nn}) \end{array} \right\}$$

$$\mathcal{C}_{\text{columns}} = \left\{ \begin{array}{l} \text{AllDiff}(X_{11}, X_{21}, X_{31}, \dots, X_{n1}), \\ \text{AllDiff}(X_{12}, X_{22}, X_{32}, \dots, X_{n2}), \\ \vdots \\ \text{AllDiff}(X_{1n}, X_{2n}, X_{3n}, \dots, X_{nn}) \end{array} \right\}$$

$$\mathcal{C}_{\text{boxes}} = \{ \text{AllDiff}(\mathcal{X}_{B_i}) \mid \mathcal{X}_{B_i} \subset 2^{\mathcal{X}}, \text{Where } \mathcal{X}_{B_i} \text{ is exactly the } i\text{-th box} \}$$

כאשר  $\text{AllDiff}$  הוא אילוץ נפוץ שכדי לכבד אותו כל המשתנים המשותפים בו מוכרים לקבל ערכים שונים זה מזה. אם כן, הגדרנו מהי בעיית סיפוק אילוצים וראינו כיצד להציג את בעיית הסודוקו כזו. בהמשך הפרק נראה דרכים עיליתות למדיד לפתרון בעיות כאלה.

### 3.2 אלגוריתם AC-3 לשמרה על עקביות קשת (Constraint Propagation)

אלגוריתם AC-3 יהיה האלגוריתם הראשון שנראה הנוגע לביעיות סיפוק אילוצים. האלגוריתם מקבל בעיית סיפוק אילוצים ומחייב בועיה מצומצמת הנקראית עקביות קשות. נגידו: נאמר כי משתנה  $X_i$  עקי בנסיבות ביחס למשתנה  $X_j$  אם עבור כל ערך בתחום של  $X_i$  קיים ערך בתחום של  $X_j$  כך שהערכים האלה לא מפרים שום אילוץ. נאמר כי בעיית סיפוק אילוצים היא עקבית בנסיבות אם כל זוג משתנים שלא עקי בנסיבות ביחס לאחר.

לדוגמא, נסתכל בעייתי סיפוק האילוצים הבא:

$$\left\langle \begin{array}{ll} \mathcal{X} : & \{X, Y\} \\ \mathcal{D} : & (\{0, 1, \dots, 9\}, \{0, 1, \dots, 9\}) \\ \mathcal{C} : & \{\langle(X, Y), \{(0, 0), (1, 1), (2, 4), (3, 9)\}\} \end{array} \right\rangle$$

כמשמעותו האילוץ היחיד היא למשה  $X^2 = Y$ . כרגע היא אינה עקבית בנסיבות, למשל כי עבור הערך 7 בתחום של  $X$  אין שום ערך מתאים בתחום של  $Y$  שמספק את הטעיה. אם נצמצם את התוחומים כך:

$$D_X = \{0, 1, 2, 3\}$$

$$D_Y = \{0, 1, 4, 9\}$$

נקבל בעיה עקבית בקשותות. נציין כי החשיבות הסמנטיבית של עקביות בקשותות היא בעצם הפעלת האילוצים במערכת: אילוץ בין  $X_i$  ל- $X_j$  יכול לשנות את התחום של שניהם, ושינוי זה יכול להשפיע בתורו על התחום של  $X_k$  ו שינוי זה יכול להשפיע על התחום של  $X_\ell$  וכן על זו הדרך. אנו מפיצים את המידע על האילוץ ברחבי המערכת. נציג את המרץ את AC-3 במנוע הפתירה שכתבתי עבור הפרויקט:

```
# -----
# INFERENCE -----
def revise(self, Xi: Variable, Xj: Variable, removals: List) -> bool:
    revised = False
    for x in self.domains[Xi][::]:
        sat_vars = [y for y in self.domains[Xj] if self.constraints(Xi, x, Xj, y)]
        if not sat_vars:
            self.domains[Xi].remove(x)
            removals.append((Xi, x))
            revised = True
    return revised

def AC3(self, removals, var: Variable = None) -> bool:
    q: Set[Tuple[Variable, Variable]] = {(Xi, Xk) for Xi in self.variables for Xk in self.neighbors[Xi]}
    \  if var is None else {(X, var) for X in \
                           self.neighbors[var]}
    while len(q) != 0:
        (Xi, Xj) = q.pop()
        if self.revise(Xi, Xj, removals):
            if not self.domains[Xi]:
                return False
            for Xk in self.neighbors[Xi]:
                if Xk != Xj:
                    q.add((Xk, Xi))
    return True
```

### 3.3 אלגוריתם MAC

אלגוריתם MAC הוא אלגוריתם ממשחת אלגוריתמי "גישוש נסוג" (Backtracking) ונתאר אותו להלן.<sup>4</sup>

בURITY הסודוקו נועדה לפתרה על ידי היפצת אילוצים. אבל במקרים רבים, לא תמיד ניתן לפתר סודוקו או בעיית סיפוק אילוצים באופן מלא רק על ידי טכניקות הסקה, נדרש גם חיפוש. אלגוריתמי גישוש נסוג עובדים עם השמה הלקטיבית, ומשלימים אותה תוך כדי עבודה תוך שמרירת העקביות שלה בכל שלב (לעומת למשל טכניקות פתרון מבוססות חיפוש מקומי בהן בניית קודם כל השמה מלאה ותוך ריצת האלגוריתם משפרים את העקביות שלה במונחים של היריסטיות עקביות כלשהי). כמו כן כל השמה מלאה ותוך ריצת האלגוריתם משפרים את העקביות מהמצבים הקיימים קובץ החשיבות שניתן לנסה את הגישה הנאייבית לחיפוש פתרון: ננסה בעייה חיפוש קלאסית בה קובץ הממצבים הקיימים החלקיים וקובץ הפעולות מכל מצב תחיה הוספה  $var = value$ . הבעיה הגדולה בפתרון זהה היא שעבור CSP בגודל  $n$  משתנים וגודל תחומים  $d$  נקבל כי בעץ החיפוש המלא ישנו  $d^n \cdot n!$  עלים, למרות שיש בדיקות  $d^n$  השמות אפשריות! הבעיה הזאת נובעת מכך שהתעלולמנו מתכוונה מכרעת של בעיות סיפוק אילוצים: הן **חילופיות** (Commutative) במובן שסדר הפעולות הפעולות על הבעיה כלל לא משנה! סדר הוספה של השמות  $var = value$  נסיף להשמה החלקית כלל לא משנה את ההשמה החלקית בסוף. בעזרת הבדיקה הזאת נשנה את התהיליך החיפוש הקלסי בצורה הבאה: בכל צומת נסתכל רק על הפעולות  $var = value$  כאשר  $var$  הוא **קבוע** ורק ערכי  $value$  משתנים. ככלומר נתלבט בין האופציות  $v_1 = v_9, X_1 = v_5, X_1 = v_1$  אבל לא נשkul את האופציה  $v_2 = X_2$  למשול.

כעת, מכשיש לנו את העץ הזה, בו בדיקות  $d^n$  עליים, נרייצ' עליו חיפוש DFS פשוט ובכל פעם שנגיע להשמה לא עקבית, נחזרה. במקרה הזה אני מעדיף קודם כל להציג את הפסואודו-קוד ולא את קוד הפיתון כי הוא לא יהיה עמוס בפרטי מימוש מבלבלים.

```

function BACKTRACKING-SEARCH(csp) returns a solution, or failure
  return BACKTRACK({ }, csp)

function BACKTRACK(assignment, csp) returns a solution, or failure
  if assignment is complete then return assignment
  var  $\leftarrow$  SELECT-UNASSIGNED-VARIABLE(csp)
  for each value in ORDER-DOMAIN-VALUES(var, assignment, csp) do
    if value is consistent with assignment then
      add {var = value} to assignment
      inferences  $\leftarrow$  INFERENCE(csp, var, value)
      if inferences  $\neq$  failure then
        add inferences to assignment
        result  $\leftarrow$  BACKTRACK(assignment, csp)
        if result  $\neq$  failure then
          return result
      remove {var = value} and inferences from assignment
  return failure

```

---

<sup>4</sup>המונחים בהם אני משתמש בהסביר מגעים כאמור מתוך תרגום חופשי שלי של המושגים ב- [Russell and Norvig, 2020] ואולי לא סטנדרטיים לגמרי בעברית.

להלן הקוד מהמנוע שכתבתי:

```
# ----- SEARCH -----  
  
def backtracking_search(self) -> Assignment:  
    """[Figure 6.5]"""  
  
    def backtrack(assignment):  
        if len(assignment) == len(self.variables):  
            return assignment  
        var = self.mrv(assignment)  
        for value in self.lcv(var, assignment):  
            if self.nconflicts(var, value, assignment) == 0:  
                self.assign(var, value, assignment)  
                removals = self.suppose(var, value)  
                if self.AC3(var=var, removals=removals):  
                    result = backtrack(assignment)  
                    if result is not None:  
                        return result  
                self.restore(removals)  
                self.unassign(var, assignment)  
            return None  
  
    return backtrack({})
```

הדברים שחורים באלגוריתם הם אלה:

1. השגורה SELECT-UNASSIGNED-VARIABLE(Clomer), יש לבחור באיזה סדר נבחר את המשתנה הלא-מושם הבא.
  2. השגורה ORDER-DOMAIN-VALUES(Clomer), יש לבחור באיזה סדר נבחר ערכים מהתחום של אותו המשתנה.
  3. השגורה INFERENCE(וותה) מפעילה כבר יש לנו, בחזרתי לכלת בכיוון של עקביות קשת, למורות שיש עוד גישות בנושא (למשל לבחור את אלגוריתם Forward Checking שמסיק פחות עקביות קשת אבל זמן הריצה שלו קטן יותר).
- את 1 ו- 2 נגדיר בתת-פרק הבא.  
בתת-פרק זה ראיינו את החישובות של החילופיות של בעיית סיפוק האילוצים, ואיך ניתן ליחס אותה על ידי שזירה משולבת של היסק וחיפוי.

### 3.4 היוריסטיקות MRV ו- LCV

מכיוון שאין לי מספיק ידע בתכנון היוריסטיקות כאלו, אציג אותן בקצרה ואפנה שוב ל- [Russell and Norvig, 2020].

להוכיח פורמלית של האופטימליות של להן.

**את המשתנים** נבחר לפי היוריסטיקת *MRV*: Minimum Remaining Values. בכל שלב בו נדרש לבחור משתנה נבחר את המשתנה לו התחום הכי קטן.

**את הערכות למשתנים** נבחר לפי היוריסטיקת *LCV*: Least Constraining Value. בכל שלב בו נדרש לבחור ערך עבור משתנה נתון, נבחר את הערך שאם נבחר אותו נמוך מהשכנים של אותו משתנה כמה שפחות ערכים.

להלן צילום מסך מתוך הקוד של המנווע שכתבתי.

```
# ----- ORDERING -----
def num_legal_values(self, var, assignment) -> int:
    if self.domains:
        return len(self.domains[var])
    else:
        return count(self.nconflicts(var, val, assignment) == 0 for val in
self.domains[var])
def lcv(self, var, assignment) -> Any:
    """Least-constraining-values heuristic."""
    return sorted(self.domains[var], key=lambda val: self.nconflicts(var, val, assignment))

def mrv(self, assignment) -> Variable:
    """Minimum-remaining-values heuristic."""
    return argmin_random_tie([v for v in self.variables if v not in assignment],
key=lambda var: self.num_legal_values(var, assignment))
```

## 4 על השימוש בلمידת מכונה לצורכי זיהוי וקלסיפיקציה

בפרק זהה אגע בנושא המכונה היליבי מבין הנושאים בחולק VI, למידת מכונה. נתבונן בכוח התיאורטי העצום ברשותות עצביות, ובין איך הן שולבו בפרויקט.

### 4.1 מהי למידת מכונה (בערך) ומהי קלסיפיקציה

למידת מכונה היא אסכולה בתחום הבינה המלאכותית העוסק באלגוריתמים \ תוכניות משתפות בביוז משימה מסוימת לאחר ניסיון כלשהו.

בתוך תחום למידת המכונה ישנו תת-תחום העוסק בעיית הסיווג: בהינתן פונקציה מסוימת

$$f : \mathcal{D} \mapsto \mathcal{C}$$

מתוך המאפייניות  $\mathcal{D}$  בתחום התוויות  $\mathcal{C}$  المسؤولות עצמן ב-  $\mathcal{D}$  למחלקות ב-  $\mathcal{C}$  אנו רוצים למצוא פונקציה  $h$  המקربת את  $f$  באופן אופטימלי במובן כלשהו.

נאמר כי מאוסף זוגות סדרדים  $\{(x, f(x)) \mid x \in \mathcal{D}\}$  אנו לומדים את הפונקציה  $h$  בעזרת אלגוריתם למידת מכונה כלשהו.

לדוגמה, אם  $\mathcal{D}$  היא קבוצת כל לקוחות בנק דיסקונט ו-  $\mathcal{C}$  היא הקבוצה המכילה את 'בסיסן אשראי גובה', 'בסיסן אשראי' בינו לבין 'אינסיקון אשראי' ו-  $f$  היא הפונקציה המתאימה לכל אדם את דירוג האשראי שלו, נאמר כי מקבוצת דגימות מההתקפות של המשתנים המקרים  $X$  (אדם בנק) ו-  $(X, f(X))$  אנו לומדים את  $h$  בעזרת אלגוריתם למידת מכונה כלשהו.

### 4.2 רשותות עצביות מלאכותיות

רשותות עצביות מלאכותיות הם מודל מתמטי חישובי, שפותח בהשראת תהליכי מוחיים או קוגניטיביים המתרחשים בראש עצבית טבעית ומשמש במסגרת למידת מכונה.

רשות מסוג זה מכילה בדרך כלל מספר רב של יחידות מידע (קלט ופלט) הקשורות זו לזו, קשרים שליעתיים קרובות עבורם דרך יחידות מידע 'חבירות'. צורת הקישור בין היחידות, המכילה מידע על חזק הקשר, מדמה את אופן חיבור הנוירונים במוח.

השימוש ברשותות עצביות מלאכותיות נפוץ בעיקר במקור במדעים קוגניטיביים, ובמערכות תוכנה שונות - בהן: מערכות רבות של אינטיגניציה מלאכותית המבצעות משימות מגוונות - זיהוי תווים, זיהוי פנים, זיהוי כתוב יד, חייזר שוק ההון, מערכת זיהוי דיבור, זיהוי תמונה, ניתוח טקסט ועוד.

את החזק של הרשותות העצביות אני חושב שראוי להציג בעזרת המשפט המתמטי הבא:  
משפט. הקירוב האוניברסלי.

לכל פונקציה  $f$ -ובוכנרג'לב אינטגרבילית  $\mathbb{R}^n \mapsto \mathbb{R}^m$  ולכל  $\epsilon > 0$  קיימת רשות עצבית קשירה לחלווטן עם פונקציה הפעלה  $h$ , בעלת עומק  $d_{m,n} = \max\{n+1, m\}$  ורוחב חסום  $w$  עבורה:

$$\int_{\mathbb{R}^n} \|f(x) - h(x)\|^p dx < \epsilon$$

ambilי להיכנס לפרטי פרטיים, המשפט טוען כי עבור כל פונקציה "נחמדה" במובן מסוים מ-  $\mathbb{R}^n$  ל-  $\mathbb{R}^m$  ניתן לקרב אותה בדיקות רבת כפל שנרצה על ידי רשותנו נוירונים עמוק  $d_{m,n}$ .

לכואורה, זה יהפוך את הרשותות העצביות לאלגוריתם הנדרש היחיד בלמידת מכונה, אבל יש לנו גם בעיות שלא אוכל לדון בהם בשלב הזה מפאת חוסר ידע מלא בעניין, שהופכות את מימוש הפטונציאל המלא שלهن למאוד קשה.

למרות זאת, ניקח מעת-פרק זה את העוצמה הגדולה של רשותות נוירונים, ונזכיר כי הן מודל סטטיסטי רב עצמה.

### 4.3 הספרייה Tensorflow

למרות הכוח העצום של רשותות עצביות באlgorigitms לומדים, הן לא תמיד שימושות כברירת המחדל גם בגלל שהן נחשות למודול ייחסי מורכב, הדורש הבנה عمוקה בסטטיסטיקה, חישוב אינפיניטיסימלי רב מושתנים, אלגריתם, אלגוריתמייה, בעיות אופטימיזציה ועוד.

הקשהזה מכבד על הרבה מתחילה בנושא (כמוני) וכאן באה לידי ביוטי הקהילה הגדולה של Data science שרצה להנגיש את הנושאזה להמוניים.

אחת מהחברות המרכזיות להרים את הcapeה ולהנגיש את הנושא היא גול. גול פיתחה את ספריית הקוד הפתוח Tensorflow (וاثת ממשק משתמש הקצה שלה, Keras פיתחו באופן עצמאי צוות בראשו עמוד פרנסואה צ'ולט) מתוך צורך אישי שלה וגם מתוך צורך של קהילת למידת המכונה להנasha של אלגוריתמי למידה הכלולים רשותות עצביות.

הרעיון העומד מאחורי Keras הוא לספק ממשק פשוט להבנה לתוכנת הקצה, מבלי שיצטרך לצלול לפרטי הפרטים של המתמטיקה המאיימת למדוי מאחוריו הקלעים.

מתוך הויקיפדיה העברית, על היכولات העיקריות של Keras:

"Keras מכילה בתוכה הרבה אבני בניין נפוצות של רשותות עצביות מלאכותיות כגון שכבות, מייעלים, פונקציות הפסד, מדדים לבדיקת המודול ודרךים לנהל דאטא.  
Keras תומכת בין השאר בראשת קובולוציה, רשת נירוניים נשנית, ורשותות עצביות מלאכותיות בעזרת שכבות "batch normalization", dropout, כינוס, אגרגציה ו-OCR.

על מנת להמיץ את הכוח העצום של Keras, נראה להלן את הקוד המלא שמנדר את הרשות העצבית בה השימושי לצורך OCR בפרויקט.

```
# import the necessary packages
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D
from tensorflow.keras.layers import MaxPooling2D
from tensorflow.keras.layers import Activation
from tensorflow.keras.layers import Flatten
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Dropout

class SudokuCNN:
    @staticmethod
    def build(width, height, depth, classes):
        # initialize the model
        model = Sequential()
        inputShape = (height, width, depth)

        # first set of CONV => RELU => POOL layers
        model.add(Conv2D(32, (5, 5), padding="same", input_shape=inputShape))
        model.add(Activation("relu"))
        model.add(MaxPooling2D(pool_size=(2, 2)))

        # second set of CONV => RELU => POOL layers
        model.add(Conv2D(32, (3, 3), padding="same"))
        model.add(Activation("relu"))
        model.add(MaxPooling2D(pool_size=(2, 2)))

        # first set of FC => RELU layers
        model.add(Flatten())
        model.add(Dense(64))
        model.add(Activation("relu"))
        model.add(Dropout(0.5))

        # second set of FC => RELU layers
        model.add(Dense(64))
        model.add(Activation("relu"))
        model.add(Dropout(0.5))

        # softmax classifier
        model.add(Dense(classes))
        model.add(Activation("softmax"))

        # return the constructed network architecture
        return model
```

מדהים! תוך שימוש בכ- 40 שורות קוד בלבד יצרנו רשות עצבית קובולוציונית, עם שכבות כינוס, אגרגציה ואפיילו שכבת dropout לצמצום התאמת יתר.  
קוד מקביל שכולל את החישובים מאחוריו הקלעים היה בערך בסדר גודל של עשרות אלפי שורות קוד, וכנראה שלא היה מגיע לרמות אופטימיזציה כמו קוד ספרייה מוכן.

#### 4.4 מאגר MNIST

השימוש שעשיתי ברשות העצבית בפרויקט שלי היה לצורך בניית מנוע OCR לצורך זיהוי ספרות (של חידת הסודוקו, מtower תמונה).

את המודל הזה רציתי למשהו עצמי כבר הרבה זמן, וסוף סוף היה לי תירוץ לעשות כן (:).  
מכיוון שהאלגוריתם צריך **ללמוד** ממוקם כלשהו, יש צורך להשיג "סט אימון" (המוח בעולם למידת המכונה לדוגמאות מהן ניתן **ללמוד**).

סט אימון אחד כמעט מושלם למטרתנו הוא מאגר *MNIST*. המאגר מורכב מ- 60,000 דוגמאות אימון (המגיעות מבוחנים סרוקים של תלמידי בת ספר *תיכוניים בארה"ב* ומכתבים של מורים האוכולסין באלה"ב), וכל דוגמת אימון היא תמונה מסדר  $28 \times 28$  פיקסלים בגוני אפור שונים בטווח 0...255 יחד עם הספרה המיוצגת בתמונה. למשל, הציג הסדר הבא

$$\left( \boxed{7}, 7 \right)$$

מהווה דוגמת אימון יחידה מתוך 60,000.

## Benchmarking 4.5 אימון ו-

הרשת העצבית לומדת את המאגר כך שבעתיד היא תוכל לסwoג תמונות חדשות של ספרות שלא ראתה. לאחר אימון הרשת העצבית בעזרת הקוד הבא:

```

from MnistCNNArchitecture import SudokuCNN
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.datasets import mnist
from sklearn.preprocessing import LabelBinarizer
from sklearn.metrics import classification_report, roc_auc_score, fbeta_score
import argparse

# construct the argument parser and parse the arguments
ap = argparse.ArgumentParser()
ap.add_argument("-m", "--model", required=True, help="path to output model after training")
args = vars(ap.parse_args())

# initialize the initial learning rate, number of epochs to train
# for, and batch size
INIT_LR = 1e-3
EPOCHS = 10
BS = 128

# grab the MNIST dataset
print("[INFO] accessing MNIST...")
((trainData, trainLabels), (testData, testLabels)) = mnist.load_data()

# add a channel (i.e., grayscale) dimension to the digits
trainData = trainData.reshape((trainData.shape[0], 28, 28, 1))
testData = testData.reshape((testData.shape[0], 28, 28, 1))

# scale data to the range of [0, 1]
trainData = trainData.astype("float32") / 255.0
testData = testData.astype("float32") / 255.0

# convert the labels from integers to vectors
le = LabelBinarizer()
trainLabels = le.fit_transform(trainLabels)
testLabels = le.transform(testLabels)

# initialize the optimizer and model
print("[INFO] compiling model...")
opt = Adam(lr=INIT_LR)
model = SudokuCNN.build(width=28, height=28, depth=1, classes=10)
model.compile(loss="categorical_crossentropy", optimizer=opt,
    metrics=["accuracy"])

# train the network
print("[INFO] training network...")
H = model.fit(
    trainData, trainLabels,
    validation_data=(testData, testLabels),
    batch_size=BS,
    epochs=EPOCHS,
    verbose=1

# evaluate the network
print("[INFO] evaluating network...")
predictions = model.predict(testData)
print(classification_report(
    testLabels.argmax(axis=1),
    predictions.argmax(axis=1),
    target_names=[str(x) for x in le.classes_]))

# serialize the model to disk
print("[INFO] serializing digit model...")
model.save(args["model"], save_format="h5")

```

גע לאחורי דיק precision, recall,(accuracy) הבא מדים:

[INFO] evaluating network...				
	precision	recall	f1-score	support
0	0.98	1.00	0.99	980
1	0.99	0.99	0.99	1135
2	0.99	0.99	0.99	1032
3	0.99	0.99	0.99	1010
4	0.99	1.00	0.99	982
5	0.98	0.99	0.99	892
6	1.00	0.99	0.99	958
7	0.99	0.99	0.99	1028
8	0.99	0.99	0.99	974
9	0.99	0.97	0.98	1009
accuracy				0.99
macro avg	0.99	0.99	0.99	10000
weighted avg	0.99	0.99	0.99	10000

## חלק V

# דברי סיום ורישימהביבליוגרפיה

ב חלק זה אביא את רישימת ההפניות למקורות בהם נזرت לארוך כל כתיבת העבודה, ואסכם את חוויות העבודה שלי על הפרויקט.

## 1 רפלקציה

אתחליל מהוסף - לדעתי היה מאוד שווה לעבור את המסע הזה של פרויקט 20 ייחדות מחשבים. החוויה העיקרית שאוכל לסכם אליה את התהליך הייתה **למיזה**. בכך כל השנה לא הפסkontי ללמידה בזוכות הפרויקט: פיתוח אנדרואיד, עיצוב ממשק משתמש deployment של שרת לענן ולא ספק עוד הרבה מאד.

אמנם בעבר יצא לי לעבוד על פרויקט בסדר גודל דומה (אסמבלר 15-ביתי), מעולם לא ציתוי לעבוד על פרויקט כל כך מגוון, זה שדרש ממני לפתח מחרבת משבצות כדי לציר עשרות לוחות סודוקו קטנים, לדבר (בראש) עם rubber ducky עם שעות ארכוכות כדי למצואו את אותו באג חמকם (בין בתוכנה עצמה ובין בעורק הדק X-FLATLAB) נכתב מסמך זה, לתכנן ממשק משתמש עם נראות מושכת, לחזור על חומרים ממספר קורסים כמו למידת מכונה ובסיסי נתונים ובעיקר לכתוב אלף שורות קוד לצורך הפרויקט.

נתקלתי בלא מעט קשיים לאורך הדרך, אותם פתרתי בעזרה לא מעט מקומות: החל מהמורה אורן וכלה בסרטוני יוטיוב בעלי 20 צפיות וקריין בעל מבטא הודי כבד (:). הנהנתי מאוד לעבוד על הפרויקט, ושמחווי שהחלהתי להרחב ל-10 ייחדות מחשבים.

## 2 תודות

בעת כתיבת הפרויקט, נעזרתי בלא מעט אנשים ארצת להודאות לחלקים במעמד זה של סיכום ספר הפרויקט.

ארצת להודאות כМОון למיר אורן אביעזר - המורה שלי - על שלימד אותנו את החומר הדורי שצורך כתיבת הפרויקט, על שהזכיר לנו לעבוד עליו ללא הרף, על הנחמדות והণינוחות והאוירה הטובה בכיתה ששרה תמיד כשהנהנה אותנו.

ארצת להודאות גם למיר אורן יפרח - אותו כינו בכיתה "המנטור" - חבר של המורה אורן שהסביר לנו מהגע פעם בשבועיים לכיתה, ולתרום לנו מהידע המשעי שלו.

הוא תכננו איתי את השירות, הציע רעיון, ועזר לי לעבוד עם פלטפורמת Azure. בludeיו העבודה נראהה כלל לא הייתה מוגעה לשלב הסופי הזה, ועל כך אני מאוד מודה לו.

**3 ביבליוגרפיה****References**

- [Android, 2017] Android (2017). Save data using sqlite : android developers. <https://developer.android.com/training/data-storage/sqlite>.
- [Base64, 2022] Base64 (2022). Base64 — Wikipedia, the free encyclopedia. <https://en.wikipedia.org/wiki/Base64>.
- [Representational state transfer, 2022] Representational state transfer (2022). Representational state transfer — Wikipedia, the free encyclopedia. [https://en.wikipedia.org/wiki/Representational\\_state\\_transfer](https://en.wikipedia.org/wiki/Representational_state_transfer).
- [Russell and Norvig, 2020] Russell, S. J. and Norvig, P. (2020). *Artificial Intelligence: A Modern Approach (4th Edition)*. Pearson.
- [Silberschatz et al., 2020] Silberschatz, A., Korth, H. F., and Sudarshan, S. (2020). *Database System Concepts, Seventh Edition*. McGraw-Hill Book Company.
- [Yato and Seta, 2003] Yato, T. and Seta, T. (2003). Complexity and completeness of finding another solution and its application to puzzles. *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, 86-A(5):1052–1060.

## חלק VI נספחים

### 1. קישור ל- GitHub

להלן זוג הקישורים לקוד המקור של האפליקציה ולשרת האפליקציה, בהתאם:

<https://github.com/Zvi-Badash/Sudo-Solve>  
<https://github.com/Zvi-Badash/Sudosolve-Backend>

### 2. כתובות השרת

נכון ליום, השרת יושב בכתובת <http://sudosolveapi.eastus.cloudapp.azure.com>