

### 3 משחק זוגי/אי-זוגי

#### 3.1 תיאור המשחק

משחק זוגי/אי-זוגי הוא משחק פשוט. המשחק מתאים להדגמת פרוטוקול הליגה.

##### 3.1.1 חוקי המשחק

1. שני שחקנים משתתפים במשחק.
2. כל שחקן בוחר "זוגי" (even) או "אי-זוגי" (odd).
3. הבחירות נעשות במקביל, בלי לדעת את בחירת היריב.
4. השופט מגריל מספר בין 1 ל-10.
5. אם המספר זוגי – מי שבחר "זוגי" מנצח.
6. אם המספר אי-זוגי – מי שבחר "אי-זוגי" מנצח.
7. אם שניהם בחרו אותו דבר ופספסו – תיקו.

##### 3.1.2 דוגמה למשחק

נניח משחק בין שחקן A לשחקן B:

טבלה 7: דוגמה למשחק זוגי/אי-זוגי

תוצאה	מספר	בחירת B	בחירת A
A מנצח	8 (even)	odd	even
B מנצח	7 (odd)	odd	even
תיקו	4 (even)	odd	odd

### 3.2 זרימת משחק בודד

#### 3.2.1 שלב 1: הזמנה למשחק

השופט שולח הזמנה לשני השחקנים. ההזמנה כוללת:

- מזהה המשחק (match\_id).
- מזהה המחזור (round\_id).
- סוג המשחק (game\_type).

### 3.2.2 שלב 2: אישור הגעה

כל שחקן מאשר קבלת ההזמנה. האישור כולל חותמת זמן.

### 3.2.3 שלב 3: איסוף בחירות

השופט פונה לכל שחקן בנפרד. הוא מבקש בחירה: "זוגי" או "אי-זוגי". השחקן מחזיר את בחירתו.

**חשוב:** השחקנים לא רואים את בחירת היריב.

### 3.2.4 שלב 4: הגרלת מספר

לאחר קבלת שתי הבחירות, השופט מגריל מספר. המספר הוא בין 1 ל-10. ההגרלה חייבת להיות אקראית.

### 3.2.5 שלב 5: קביעת מנצח

השופט בודק:

- אם המספר זוגי ושחקן בחר "זוגי" – הוא מנצח.
- אם המספר אי-זוגי ושחקן בחר "אי-זוגי" – הוא מנצח.
- אם שניהם ניחשו נכון/לא נכון – תיקו.

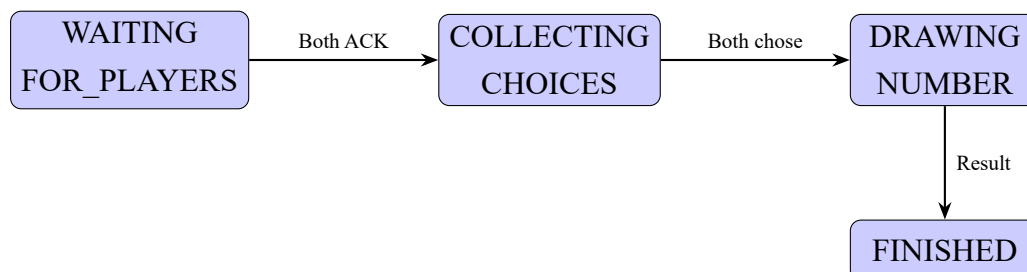
### 3.2.6 שלב 6: דיווח תוצאה

השופט שולח:

1. הודעת GAME\_OVER לשני השחקנים.
2. הודעת MATCH\_RESULT\_REPORT למנהל הליגה.

## 3.3 מצבי המשחק

המשחק עובר בין מצבים מוגדרים:



### 3.3.1 מצב WAITING\_FOR\_PLAYERS

המשחק מתחיל במצב זה. השופט מחכה שהשחקנים יאשרו הגעה. המעבר: כששני השחקנים שלחו GAME\_JOIN\_ACK.

### 3.3.2 מצב COLLECTING\_CHOICES

השופט אוסף בחירות מהשחקנים. הוא קורא ל-choose\_parity של כל שחקן. המעבר: כששתי הבחירות התקבלו.

### 3.3.3 מצב DRAWING\_NUMBER

השופט מגריל מספר וקובע מנצח. המעבר: אוטומטי לאחר החישוב.

### 3.3.4 מצב FINISHED

המשחק הסתיים. התוצאה דווחה.

## 3.4 שיטת הניקוד

### 3.4.1 ניקוד למשחק

טבלה 8: טבלת ניקוד

תוצאה	נקודות מנצח	נקודות מפסיד
ניצחון	3	0
תיקו	1	1
הפסד	0	0

### 3.4.2 דירוג בליגה

הדירוג נקבע לפי:

1. סך הנקודות (יורד).
2. מספר הניצחונות (יורד).
3. הפרש תיקו (יורד).

## 3.5 ליגת Round-Robin

בליגה עם 4 שחקנים, כל שחקן משחק נגד כולם.

### 3.5.1 מספר משחקים

עבור  $n$  שחקנים:

- מספר משחקים בליגה:  $\frac{n(n-1)}{2}$

- עבור 4 שחקנים:  $\frac{4 \times 3}{2} = 6$  משחקים

### 3.5.2 לוח משחקים לדוגמה

טבלה 9: לוח משחקים ל-4 שחקנים

משחק	שחקן A	שחקן B
R1M1	P01	P02
R1M2	P03	P04
R2M1	P01	P03
R2M2	P02	P04
R3M1	P01	P04
R3M2	P02	P03

## 3.6 אסטרטגיות לשחקנים

### 3.6.1 אסטרטגיה אקראית

הגישה הפשוטה ביותר. השחקן בוחר באקראי "זוגי" או "אי-זוגי". הסיכוי לנצח הוא 50%.

#### אסטרטגיה אקראית

```
import random

def choose_parity_random():
    return random.choice(["even", "odd"])
```

### 3.6.2 אסטרטגיה מבוססת היסטוריה

השחקן זוכר תוצאות קודמות. הוא מנסה לזהות דפוסים בהגרלות. **הערה:** מכיוון שההגרלה אקראית, אסטרטגיה זו לא תשפר תוצאות לטווח ארוך.

### 3.6.3 אסטרטגיה מונחית LLM

השחקן יכול להשתמש במודל שפה. הוא בונה prompt ושואל את המודל.

#### דוגמה ל-prompt

```
prompt = """
You are playing Even/Odd game.
Choose "even" or "odd".
Previous results: even won 3 times, odd won 2 times.
Your choice (one word only):
"""
```

**הערה:** השימוש ב-LLM מעניין אך לא ישפר ביצועים סטטיסטית. המשחק הוא משחק מזל.

### 3.7 מודול חוקי המשחק

מודול החוקים הוא רכיב נפרד בשופט. הוא מגדיר את הלוגיקה הספציפית למשחק.

#### 3.7.1 ממשק המודול

המודול מספק פונקציות:

- `init_game_state()` – אתחול מצב משחק.
- `validate_choice(choice)` – בדיקת חוקיות בחירה.
- `draw_number()` – הגרלת מספר.
- `determine_winner(choices, number)` – קביעת מנצח.

#### 3.7.2 יתרון ההפרדה

בעתיד, אפשר להחליף את המודול. במקום זוגי/אי-זוגי, אפשר:

- איקס-עיגול (Tic-Tac-Toe).
- 21 שאלות.
- משחק זיכרון.

הפרוטוקול הכללי נשאר זהה. רק מודול החוקים משתנה.

### 3.8 הרחבה למשחקים נוספים

הפרוטוקול מתוכנן להיות כללי ולא ספציפי למשחק זוגי/אי-זוגי. סעיף זה מתאר את השכבה הגנרית המאפשרת הוספת משחקים נוספים.

#### 3.8.1 הפשטת מהלך – `GAME_MOVE`

הודעות `CHOOSE_PARITY_CALL` ו-`CHOOSE_PARITY_RESPONSE` הן מקרה פרטי של הפשטה כללית יותר:

טבלה 10: הקבלה בין הודעות ספציפיות לגנריות

הודעה ספציפית	הודעה גנרית
<code>CHOOSE_PARITY_CALL</code>	<code>GAME_MOVE_CALL</code>
<code>CHOOSE_PARITY_RESPONSE</code>	<code>GAME_MOVE_RESPONSE</code>

**GAME\_MOVE\_CALL – בקשת מהלך גנרית**

```
{
  "protocol": "league.v2",
  "message_type": "GAME_MOVE_CALL",
  "sender": "referee:REF01",
  "timestamp": "2025-01-15T10:30:15Z",
  "match_id": "R1M1",
  "player_id": "P01",
  "game_type": "even_odd",
  "move_request": {
    "move_type": "choose_parity",
    "valid_options": ["even", "odd"],
    "context": {}
  },
  "deadline": "2025-01-15T10:30:45Z"
}
```

**GAME\_MOVE\_RESPONSE – תגובת מהלך גנרית**

```
{
  "protocol": "league.v2",
  "message_type": "GAME_MOVE_RESPONSE",
  "sender": "player:P01",
  "timestamp": "2025-01-15T10:30:20Z",
  "match_id": "R1M1",
  "player_id": "P01",
  "game_type": "even_odd",
  "move_data": {
    "move_type": "choose_parity",
    "choice": "even"
  }
}
```

### 3.8.3 רישום סוגי משחקים – Game Registry

מנהל הליגה מחזיק רישום של סוגי משחקים נתמכים:

#### רישום משחקים

```
{
  "game_registry": {
    "even_odd": {
      "display_name": "Even/Odd",
      "move_types": ["choose_parity"],
      "valid_choices": {
        "choose_parity": ["even", "odd"]
      },
      "min_players": 2,
      "max_players": 2
    },
    "tic_tac_toe": {
      "display_name": "Tic-Tac-Toe",
      "move_types": ["place_mark"],
      "valid_choices": {
        "place_mark": ["0-8"]
      },
      "min_players": 2,
      "max_players": 2
    }
  }
}
```

### 3.8.4 יתרונות ההפשטה

1. **הוספת משחקים חדשים** – ללא שינוי בפרוטוקול הבסיסי.
  2. **גילוי יכולות** – שחקן יכול לשאול אילו משחקים נתמכים.
  3. **וולידציה אחידה** – השופט מוודא שהמהלך חוקי לפי ה-schema.
  4. **תאימות קדימה** – סוכנים ישנים יכולים להמשיך לעבוד עם הודעות ספציפיות.
- הערה:** בתרגיל זה משתמשים בהודעות הספציפיות (CHOOSE\_PARITY\_\*). ההפשטה הגנרית מוצגת לצורך הבנת הארכיטקטורה.