פרויקט צוללות (שחקן יחיד)

- לוח: מטריצה בגודל X × X (ערך X נקבע מראש במשתנה; למתקדמים — קלט מ־CLI/קובץ).
- אתחול לוח: כל התאים מאותחלים ל־0 (או None).
- פרמטרים נוספים:
  - מספר צוללות למקם (למתקדמים — קלט מ־CLI/קובץ).
  - מספר יריות מותר (למתקדמים — קלט מ־CLI/קובץ).
- יש למקם את כל הצוללות באופן אקראי על הלוח. במקום שנמצאת בו צוללת יש להחליף את ה0 (או הNONE ב1)
- תצוגה למסך (בזמן המשחק):
  - תא שלא ירו עליו: מציגים O.
  - תא שירו עליו **ואין** בו צוללת: מציגים X.
  - תא שירו עליו **ויש** בו צוללת: מציגים V.
- תור השחקן (כל עוד נותרו יריות ולא כל הצוללות טובעו):
  - הצגת הלוח -כולל V X O קודמים.
  - בחירת קואורדינטות X,Y
  - לוודא שהקלט בתחום המותר ושטרם נורה על תא זה.
  - ירי.
  - הודעה: פגיעה/פספוס.
  - עדכון הלוח: X לפספוס, V לפגיעה
  - להדפיס סטטוס: מספר יריות שנותרו, מספר צוללות שנותרו.
- סיום:
  - ניצחון: כל הצוללות טובעו לפני שנגמרו היריות.
  - הפסד: נגמרו היריות ועדיין נותרו צוללות; מציגים לוח סופי **עם מיקומי הצוללות חשופים** (1 היכן שיש צוללת, יחד עם הX והV והO הקודמים).

מבנה תיקיות (אופציונלי)

```
submarines_project/
├── main.py              # נקודת כניסה (CLI)
├── submarines/
│   ├── __init__.py
│   ├── board.py         # יצירה/עדכון/תצוגת לוח
│   ├── placement.py     # מיקום צוללות אקראי
│   ├── game.py          # לוגיקת משחק וחוקי עצירה
│   └── io.py            # קלט/פלט (CLI)
└── tests/
    └── test_game.py     # בדיקות יחידה בסיסיות (לא חובה)
```

אפשר כמובן לאחד קבצים או לוותר על תיקיות — זה רק סדר מוצע.

ממשק פונקציות (אופציונלי)

ייצוג נתונים מוצע

כדי לתמוך גם בתצוגה במהלך המשחק וגם בחשיפה בסוף:

- ships: list[list[int]] — מטריצה של 0/1 (צוללת=1).
- shots: list[list[bool]] — מטריצה של False/True (האם נורה על התא).
- תצוגה נגזרת מחיתוך שני הלוחות:
  - אם shots[y][x] == False → O
  - אם shots[y][x] == True ו־ships[y][x] == 0 → X
  - אם shots[y][x] == True ו־ships[y][x] == 1 → V

submarines/[board.py]

- create_matrix(size: int, fill: int = 0) -> list[list[int]]
  מחזיר מטריצה מרובעת בגודל size־ל מאופסת fill.

- create_bool_matrix(size: int, fill: bool = False) -> list[list[bool]]

- in_bounds(size: int, x: int, y: int) -> bool
  בדיקת גבולות.

- count_remaining_ships(ships: list[list[int]], shots: list[list[bool]]) -> int
  כמה תאי צוללת (1) טרם נחשפו בירי.

- render_public(ships: list[list[int]], shots: list[list[bool]]) -> str
  מחזיר מחרוזת לתצוגת המשחק (עם O/X/V).

- render_reveal(ships: list[list[int]], shots: list[list[bool]]) -> str
  מחזיר מחרוזת לוח סופי עם חשיפת כל הצוללות (1 במיקומים שיש צוללת).

submarines/placement.py

- place_random_ships(ships: list[list[int]], n: int) -> None

submarines/[game.py]

- init_game(size: int, n_ships: int, max_shots: int, *, rng: random.Random | None = None) -> dict
  יוצר מצב משחק:

  ```
  {
   "size": int,
   "ships": list[list[int]],
   "shots": list[list[bool]],
   "n_ships": int,
   "max_shots": int,
   "shots_used": int
  }
  ```

- shoot(state: dict, x: int, y: int) -> tuple[bool, str]
  מבצע ירי אם לא נורה קודם ועל הגבולות; מחזיר (is_hit, message).
  מעדכן shots_used לגבול (לא ירי כפול/מחוץ לגבול רק בניסיון ירי תקין).

- is_won(state: dict) -> bool
  כל תאי ה־1 נורו.

- is_lost(state: dict) -> bool
  shots_used >= max_shots ועדיין יש תאי צוללת לא־פגועים.

- shots_left(state: dict) -> int

- remaining_ships(state: dict) -> int

submarines/io.py

- parse_coords(raw: str, *, one_based: bool = True) -> tuple[int, int] | None
  ממיר קלט משתמש x y 1 לקואורדינטות, עם אפשרות based-.

- print_status(state: dict) -> None
  מדפיס לוח ציבורי (render_public) + צוללות שנותרו + יריות שנותרו.

- print_end(state: dict, won: bool) -> None
  מדפיס תוצאה; אם הפסד — מציג חשיפה (render_reveal).

main.py

- play(size: int = 6, n_ships: int = 6, max_shots: int = 10, *, one_based: bool = True) -> None
  לולאת המשחק הראשית:

  1. אתחול מצב; 2) כל עוד לא ניצחון/הפסד ויש יריות — קלט קואורדינטות, ירי, הודעה;

  2. בסיום — תוצאה וחשיפה אם צריך.

- if __name__ == "__main__":
  size, n_ships, max_shots, לקריאת argparse — לטעון פרמטרים קבועים; למתקדמים --seed ואופציונלית.

כללי ולידציה והתנהגות (מומלץ)

- קלט:
  - קואורדינטות בתחום (לפי 0-based/1-based שנקבע).
  - דחיית ירי לתא שכבר נורה — לא מפחיתים יריות במקרה כזה, ומבקשים קלט אחר.
- מיקום צוללות:
  - ללא חפיפה; אם n_ships > size*size — להעלות שגיאה.

- ספירת יריות:
  - סופרים **רק** ירי חוקי (בתחום ועל תא שלא נורה).
- הודעות:
  - פגיעה/פספוס; סטטוס אחרי כל ירי.

- סיום:

  - ניצחון מיידי ברגע שכל הצוללות טובעו.
  - הפסד כשנגמרו היריות ועדיין נותרו צוללות; חשיפת 1 בכל מיקום צוללת.

תרחישי קצה (מומלץ לבדיקה)

- ירי כפול על אותו תא — לא להפחית יריות; להתריע ולבקש קלט חדש.
- קואורדינטות מחוץ לטווח — דחייה ולא להפחית יריות.
- n_ships == 0 (ניצחון מיידי) / max_shots == 0 (הפסד מיידי אם יש צוללות).
- לוח קטן (1×1/2×2) ומספר צוללות קיצוני.
- רנדומליות דטרמיניסטית עם seed לצורך בדיקות.