

Python Projects

מתן חיים סנדורי

כתב ויתור:

הפרויקטים המוצגים במסמך זה הינם קנייניים ועוצבו ופותחו על ידי מתן סנדורי. פרויקטים אלה אינם מיועדים לשכפול, רווח או שימוש אישי על ידי צד שלישי כלשהו. כל הפצה, הצגה או שכפול בלתי מורשית של פרויקטים אלה אסורים בהחלט. לבירורים או אם ברצונכם לראות את קוד המקור המלא, אנא פנו ישירות למתן סנדורי.

תוכן העניינים:

2.....	מומחיות אישית Python
3.....	תעודת PCAP
4.....	תעודת Tensorflow
5.....	תעודת EF SET
6.....	מחשבון באמצעות מחרוזות
7.....	AI איקס עיגול עם אלגוריתם
8.....	סימולטור ארבע בשורה
9.....	סימולטור ספינת קרב
10.....	סימולטור משחק שחמט
12.....	מעקב אחר הוצאות (Pandas) באמצעות
13.....	מסקנה ויצירת קשר

פרטי יצירת קשר:

שם מלא: מתן חיים סנדורי.

דוא"ל: Matans806@gmail.com

מספר טלפון: 052-658-2600

מומחיות אישית Python

עם ניסיון של 5 שנים ב-Python, שלטתי ב:

Core Python: תכנות ברמה גבוהה עם צלילה עמוקה לתוך מבני נתונים כמו רשימות מקושרות.

תוספות Python:

TensorFlow: מתמחה בלמידת מכונה ורשתות עצביות. בעל ניסיון מעשי בבעיות רגרסיה, סיווג בינארי/רב-מעמדי, סיווג תמונות, NLP, חיזוי סדרות זמן, DCGANs, Cycle GAN, זיהוי אובייקטים, פילוח תמונה, מקודדים אוטומטיים ועוד.

NumPy: מומחיות ביצירה, עריכת מערכים ושימוש בפונקציות רבות.

Pandas: אחזר וטפל ביעילות במערך נתונים באמצעות מסגרות נתונים. מיומן בטיפול ב-CSV, עריכת Data Frame ותווית.

Matplotlib: מיומן בשרטוט גרפים ותמונות.

Sklearn: מכיר את train_test_split, מדדי דיוק ואלגוריתמים שונים של למידת מכונה.

תכנות Socket: הבנה בחיבורי שרתים, סריקת יציאות ואפילו פונקציות מתקדמות כמו שליחת וירוסים (למטרות אתיות).

הרחבות Python:

OS, Math, Sys, Random, Turtle, cv2, tqdm, re, pathlib, PIL, Threading ועוד.

תעודת PCAP




השגתי ציון 88/100 בבחינת PCAP-31-03 ממכון Python. הבדיקה כוללת חמישה חלקים המשתרעים על מודולים, טיפול בשגיאות, פעולות מחרוזות, תכנות מונחה עצמים וטכניקות מתקדמות כמו פונקציות lambda ופעולות I/O. הסמכה זו מדגישה את עומק הידע שלי ב-Python, ומשקפת מיומנות במושגי Python בסיסיים וגם מורכבים. [בקר באתר האינטרנט של מכון Python.](#)

תעודת Tensorflow



השגתי ציון של 24/25 (96%) בבחינת ההסמכה של TensorFlow. מבחן אינטנסיבי זה בן 5 שעות, מאורגן על ידי גוגל, מעריך את היכולת של האדם לתכנן ולהטמיע רשתות עצביות ואלגוריתמי למידה עמוקה באמצעות חבילת TensorFlow. הציון משקף את מיומנותי במינוף היכולות של TensorFlow למיצוי הפוטנציאל שלה. [גלה עוד על הסמכת TensorFlow כאן](#) או [בדוק את ספריית האישורים של Google](#).

תעודת EF SET

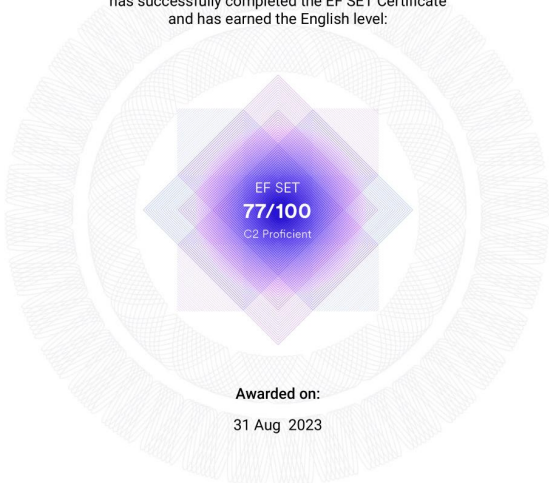


EF STANDARD ENGLISH TEST

This is to certify that

Matan Sandori

has successfully completed the EF SET Certificate
and has earned the English level:




EF SET
77/100
C2 Proficient

Awarded on:
31 Aug 2023

Understanding the results

EFSET	1-30	31-40	41-50	51-60	61-70	71-100
CEFR	A1 Beginner	A2 Elementary	B1 Intermediate	B2 Upper Intermediate	C1 Advanced	C2 Proficient


Your level of English is **77/100** on the EF SET score scale and **C2 Proficient** according to the Common European Framework of Reference (CEFR). This score is calculated as an average of your reading and listening scores



Listening Section
76/100 C2 Proficient

You are comfortable in all situations that require full comprehension of spoken English; you are almost never confused or searching for the meaning of words and phrases. You understand nuances of expression and tone, humor and emphasis in all live theatrical presentations, films or broadcast presentations in English.

- Can understand with ease any kind of spoken language, even when delivered at fast native speed, provided with time to get familiar with any regional or other accent.
- Can understand lectures and presentations with a high degree of colloquialism, regional usage and unfamiliar terminology.



Reading Section
78/100 C2 Proficient

Your command of English allows you to read virtually any kind of text (factual, literary, technical) and accurately recognize and categorize style and tone. You can understand complex technical writing on unfamiliar subjects on a wide range of topics.

- Can read with ease virtually all forms of written language, including abstract, structurally or linguistically complex texts such as manuals, specialized articles and literary works.
- Can understand a wide range of long and complex texts, appreciating subtle distinctions of style, and implicit meaning.

www.efset.org/cert/6uirkP

השגתי ציון של 77/100 בבחינה המאתגרת באנגלית EF SET, מה שמציב אותי בקטגוריית הבקיאיות הגבוהה ביותר. השליטה שלי בשפה האנגלית לא באה לידי ביטוי רק בציוני המבחנים אלא גם בחיי היומיום שלי שבהם אני קורא, צופה, לומד וכותב בעיקר באנגלית. מיומנות זו מבטיחה תקשורת והבנה ברורה, במיוחד בעת טיפול בתיעוד Python, קורסים ואינטראקציות. [למידע נוסף על בדיקת EF SET כאן.](#)

מחשבון באמצעות מחרוזות

בגיל 16 התחלתי ליצור מחשבון באמצעות מחרזות. הפרויקט, למרות שנראה פשוט, היה מורכב בדרכו. בהינתן מחרוזת קלט כמו "15 + 63 / 53 * 5 - 2", התוכנית מפרקת אותה בחריצות תוך כיבוד העדיפות המתמטית של כפל, חילוק, חיבור וחיסור. כל שגיאה אפשרית נלקחה בחשבון כדי להבטיח פעולה חלקה. פרויקט 144 שורות של קוד.

```

1 class calculator:
2     def __init__(self):
3         self.string = ""
4         self.as_arr = []
5     def set_string(self):
6         self.string = input("Enter mathematical exercise (ex: 5 + 5) \n");
7     def split_string(self):
8         result = [];
9
10
11
12     integer = False;
13
14     for let in self.string:
15         if(integer):
16             try:
17                 can_convert = int(let);
18                 result[len(result) - 1] = int(f"{str(result[len(result) - 1])}{let}");
19                 continue;
20             except:
21                 integer = False;

```

TERMINAL OUTPUT DEBUG CONSOLE PROBLEMS

Python + [icon] [icon] [icon]

```

PS C:\Users\mcfn\Desktop\The Folder\Pograming\Other\Python\work> conda activate env
PS C:\Users\mcfn\Desktop\The Folder\Pograming\Other\Python\work> & C:\Users\mcfn\anaconda3\envs\env\python.exe "C:\Users\mcfn\Desktop\The
Folder\Pograming\Other\Python\work\Other\Games\calculator.py"
Enter mathematical exercise (ex: 5 + 5)
5 + 5 * 30 / 2 + 2 / 2
[5, '+', 5, '+', 30, '/', 2, '+', 2, '/', 2]
[5, '+', 75.0, '+', 1.0]
[81.0]

```

```

36         m = self.as_arr[j] + self.as_arr[k];
37
38     self.update_arr(j, m);
39     i = 0;
40     elif(self.as_arr[i] == "-"):
41         m = self.as_arr[j] - self.as_arr[k];
42
43     self.update_arr(j, m);
44     i = 0;
45     i += 1;
46
47 cal = calculator();
48 cal.set_string();
49 cal.split_string();
50 print(cal.as_arr);
51 cal.multiply_and_divide();
52 print(cal.as_arr);
53 cal.add_and_subtract();
54 print(cal.as_arr);

```

TERMINAL OUTPUT DEBUG CONSOLE PROBLEMS

[81.0]

PS C:\Users\mcfn\Desktop\The Folder\Pograming\Other\Python\work> & C:\Users\mcfn\anaconda3\envs\env\python.exe "C:\Users\mcfn\Desktop\The Folder\Pograming\Other\Python\work\Other\Games\calculator.py"

Enter mathematical exercise (ex: 5 + 5)

1000 * 2053 / 10 + 1000 / 10000 * 5354364 + 53255 - 12351 / 242 * 244

[1000, '+', 2053, '/', 10, '+', 1000, '/', 10000, '*', 5354364, '+', 53255, '-', 12351, '/', 242, '*', 244]

[205300.0, '+', 535436.4, '+', 53255, '-', 12453.07438016529]

[781538.3256198347]

איקס עיגול עם אלגוריתם AI

איקס עיגול - החלטתי לקחת את זה צעד קדימה. עיצבתי לוח CLI שמתעדכן על סמך קלט הנגן. מערך מייצג את הלוח, אשר לאחר מכן הופך למחרוזת לתצוגה. הוספתי תכונה של AI שמנבא ומחשב את התוצאה של כל פעולה. אם תוצאה חותכת אינה בהישג יד, הAI יבחר פעולה אקראית. פרויקט זה בן 344 שורות הושלם בגיל 16.

```
5 class XO():
6     def __init__(self):
7         self.board = [[0, 0, 0],
8                       [0, 0, 0],
9                       [0, 0, 0]]
10        self.player_turn = 1;
11
12    def show_location(self):
13        location = [(1, 1), (1, 2), (1, 3)],
14                  [(2, 1), (2, 2), (2, 3)],
15                  [(3, 1), (3, 2), (3, 3)]];
16        for arr in location:
17            print(arr);
18            print("\n");
19
20    def get_board(self, copy=True):
21        if(copy):
22            board_copy = [];
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

```
1 | - | - | x |
2 | - | o | - |
3 | - | - | x |

[*] Player: 2 Turn!

[*] location X:
1 | - | - | x |
2 | - | o | - |
3 | - | - | x |

[*] Player: 1 Turn!
[+] [[2, 0, 1], [0, 2, 2], [0, 0, 1]] False 0 0
[+] [[0, 2, 1], [0, 2, 2], [0, 0, 1]] False 0 1
[+] [[0, 0, 1], [2, 2, 2], [0, 0, 1]] True 1 0
[*] Board updated at ( 2 , 1 )
1 | - | - | x |
2 | x | o | o |
3 | - | - | x |

[*] Player: 2 Turn!

[*] location X:
1 | - | - | x |
2 | x | o | o |
3 | - | - | x |

395         i, j = arr[random.randint(0, len(arr)-1)];
396         print("[X]" , i, j);
397
398         return i, j;
399
400     xo = XO();
401     result = xo.play_with_bot(raw=False, player=2, lvl=2);
402     #result = xo.play(raw=False);
403     xo.show_board();
404     print(result, "\n");

1 | o | - | - |
2 | x | x | x |
3 | - | - | o |

Player 1 won!
```

סימולטור ארבע בשורה

פיתחתי סימולטור של ארבעה בשורה בגיל 16. מכניקת המשחק עובדת על אותו עיקרון כמו איקס עיגול, אבל עם כללים מתקדמים ולוח גדול יותר המיוצג באמצעות מערכים. המשיכה החזותית של ה-CLI מוגברת באמצעות `termcolor.colored`. כשהגיע ל-198 שורות קוד.

```
72     if(rev_i):
73         cur_index -= 1;
74     else:
75         cur_index += 1;
76     if(diag != []):
77         final_result.append(diag);
78     return final_result;
79
80 diagonales_0 = get_diagonales(board, rev_i=False, rev_j=False);
81 diagonales_1 = get_diagonales(board, rev_i=True, rev_j=False);
82 diagonales_2 = get_diagonales(board, rev_i=False, rev_j=True);
83 diagonales_3 = get_diagonales(board, rev_i=True, rev_j=True);
84
85 diagonales = [diagonales_0, diagonales_1, diagonales_2, diagonales_3];
86
87 for diag in diagonales:
88     # ...

```

TERMINAL OUTPUT DEBUG CONSOLE PROBLEMS

Python + - [] [x] [y]

1	2	3	4	5	6	7
-	-	-	-	-	-	-
-	-	-	-	-	-	-
-	-	-	O	-	-	-
-	-	O	O	-	O	-
-	O	O	O	-	O	-
O	O	O	O	-	O	-

-- Player 2 turn --

Enter number <1 - 7>:

```
167
168
169     def start(self, raw=False):
170         win = four_in_a_row_game.win;
171
172         while True:
173             self.show(raw=raw);
174
175             print(f"\n -- Player {self.player} turn -- \n");
176
177             num = self.ask();
178             suc = self.add_to_board(num);
179
180             if(not(suc)):
181                 print("\n -- Unexpected error | Try again | --\n");
182                 continue;
```


סימולטור ספינת קרב

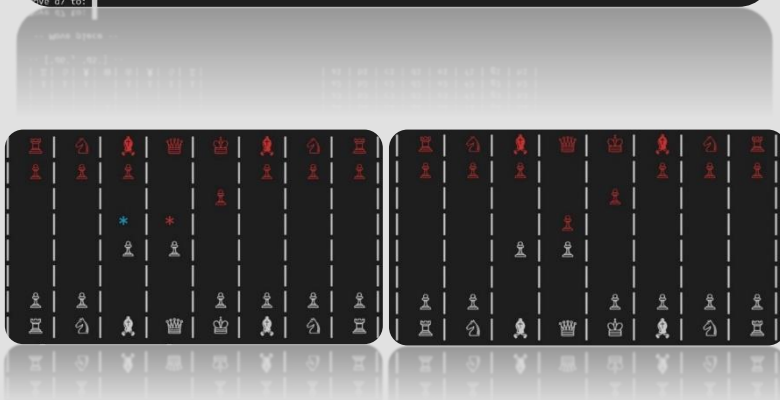
Battleship, משחק קלאסי של אסטרטגיה וניחושים. במקום הדירקטוריון המסורתי, בחרתי בנציגות של CLI. שחקנים מתחילים במיקום הצי שלהם ואז מתחלפים להפיץ זה את הספינות של זה. תכונה ייחודית ששילבתי הייתה הוספת יכולות מיוחדות. שחקנים יכלו להשתמש בסורק עד 3 פעמים כדי לקבל רמזים על מיקומי הספינה של האויב. היו להם גם התקפות נפץ (2 שימושים) להשפעה רחבה יותר וגרעין הרסני (שימוש אחד) כדי לזרוע הרס. כמובן, המתקפה הרגילה הייתה זמינה עד אין סוף. עם 2 מערכים המייצגים את מצב המשחק עבור כל שחקן ופונקציות שונות לטיפול בלוגיקה של המשחק, ויזואליה והיכולות המיוחדות הללו, פרויקט הקוד הזה בן 395 השורות היה מהנה ומאתגר מאוד. נבנה בגיל 16, סימולטור זה הציג את היצירתיות שלי בשיפור המשחק המסורתי.

```
9         self._init_guess();
10
11         self.board_1 = [[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
12                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
13                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
14                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
15                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
16                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
17                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
18                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]];
19
20         self.board_2 = self.copy(self.board_1);
21
22         self.guess_board_1 = self.copy(self.board_1);
23         self.guess_board_2 = self.copy(self.board_1);
24
25         self.player = 1;
26
27         self.abilities_1 = [3, 2, 1];
28
29     def _init_guess(self):
30         self.guess_board_1 = self.copy(self.board_1);
31         self.guess_board_2 = self.copy(self.board_1);
32
33         self.player = 1;
34
35         self.abilities_1 = [3, 2, 1];
36
37     def _init_board(self):
38         self.board_1 = [[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
39                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
40                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
41                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
42                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
43                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
44                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
45                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]];
46
47         self.board_2 = self.copy(self.board_1);
48
49         self.guess_board_1 = self.copy(self.board_1);
50         self.guess_board_2 = self.copy(self.board_1);
51
52         self.player = 1;
53
54         self.abilities_1 = [3, 2, 1];
55
56     def _init_guess(self):
57         self.guess_board_1 = self.copy(self.board_1);
58         self.guess_board_2 = self.copy(self.board_1);
59
60         self.player = 1;
61
62         self.abilities_1 = [3, 2, 1];
63
64     def _init_board(self):
65         self.board_1 = [[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
66                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
67                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
68                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
69                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
70                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
71                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
72                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]];
73
74         self.board_2 = self.copy(self.board_1);
75
76         self.guess_board_1 = self.copy(self.board_1);
77         self.guess_board_2 = self.copy(self.board_1);
78
79         self.player = 1;
80
81         self.abilities_1 = [3, 2, 1];
82
83     def _init_guess(self):
84         self.guess_board_1 = self.copy(self.board_1);
85         self.guess_board_2 = self.copy(self.board_1);
86
87         self.player = 1;
88
89         self.abilities_1 = [3, 2, 1];
90
91     def _init_board(self):
92         self.board_1 = [[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
93                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
94                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
95                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
96                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
97                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
98                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
99                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]];
100
101         self.board_2 = self.copy(self.board_1);
102
103         self.guess_board_1 = self.copy(self.board_1);
104         self.guess_board_2 = self.copy(self.board_1);
105
106         self.player = 1;
107
108         self.abilities_1 = [3, 2, 1];
109
110     def _init_guess(self):
111         self.guess_board_1 = self.copy(self.board_1);
112         self.guess_board_2 = self.copy(self.board_1);
113
114         self.player = 1;
115
116         self.abilities_1 = [3, 2, 1];
117
118     def _init_board(self):
119         self.board_1 = [[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
120                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
121                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
122                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
123                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
124                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
125                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
126                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]];
127
128         self.board_2 = self.copy(self.board_1);
129
130         self.guess_board_1 = self.copy(self.board_1);
131         self.guess_board_2 = self.copy(self.board_1);
132
133         self.player = 1;
134
135         self.abilities_1 = [3, 2, 1];
136
137     def _init_guess(self):
138         self.guess_board_1 = self.copy(self.board_1);
139         self.guess_board_2 = self.copy(self.board_1);
140
141         self.player = 1;
142
143         self.abilities_1 = [3, 2, 1];
144
145     def _init_board(self):
146         self.board_1 = [[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
147                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
148                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
149                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
150                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
151                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
152                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
153                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]];
154
155         self.board_2 = self.copy(self.board_1);
156
157         self.guess_board_1 = self.copy(self.board_1);
158         self.guess_board_2 = self.copy(self.board_1);
159
160         self.player = 1;
161
162         self.abilities_1 = [3, 2, 1];
163
164     def _init_guess(self):
165         self.guess_board_1 = self.copy(self.board_1);
166         self.guess_board_2 = self.copy(self.board_1);
167
168         self.player = 1;
169
170         self.abilities_1 = [3, 2, 1];
171
172     def _init_board(self):
173         self.board_1 = [[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
174                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
175                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
176                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
177                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
178                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
179                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
180                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]];
181
182         self.board_2 = self.copy(self.board_1);
183
184         self.guess_board_1 = self.copy(self.board_1);
185         self.guess_board_2 = self.copy(self.board_1);
186
187         self.player = 1;
188
189         self.abilities_1 = [3, 2, 1];
190
191     def _init_guess(self):
192         self.guess_board_1 = self.copy(self.board_1);
193         self.guess_board_2 = self.copy(self.board_1);
194
195         self.player = 1;
196
197         self.abilities_1 = [3, 2, 1];
198
199     def _init_board(self):
200         self.board_1 = [[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
201                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
202                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
203                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
204                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
205                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
206                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
207                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]];
208
209         self.board_2 = self.copy(self.board_1);
210
211         self.guess_board_1 = self.copy(self.board_1);
212         self.guess_board_2 = self.copy(self.board_1);
213
214         self.player = 1;
215
216         self.abilities_1 = [3, 2, 1];
217
218     def _init_guess(self):
219         self.guess_board_1 = self.copy(self.board_1);
220         self.guess_board_2 = self.copy(self.board_1);
221
222         self.player = 1;
223
224         self.abilities_1 = [3, 2, 1];
225
226     def _init_board(self):
227         self.board_1 = [[0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
228                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
229                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
230                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
231                         [0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
232                         [0, 0, 0
```

סימולטור משחק שחמט

שחמט הוא לא רק משחק; זה ריקוד מורכב של אסטרטגיה, חיזוי וטקטיקה. בהתחשב בחוקים המורכבים שלו ושלל תנועות הכלים, קידוד סימולטור היה אתגר שלקחתי עליו בגיל 17. עם מערך המייצג את הלוח, כל חלק - מהחייל ועד המלכה - היה צריך להיות מקודד בקפידה. מהלכים מיוחדים כמו קידום פיון ו-en passant הביאו למורכבות נוספת. אבל אחרי 615 שורות של קוד, הפרויקט הסתיים.

```
1 from termcolor import colored
2
3 class game:
4     class play:
5         def __init__(self):
6             self.board = [[-4, -2, -3, -5, -6, -3, -2, -4],
7                             [-1, -1, -1, -1, -1, -1, -1, -1],
8                             [0, 0, 0, 0, 0, 0, 0, 0],
9                             [0, 0, 0, 0, 0, 0, 0, 0],
10                            [0, 0, 0, 0, 0, 0, 0, 0],
11                            [0, 0, 0, 0, 0, 0, 0, 0],
12                            [1, 1, 1, 1, 1, 1, 1, 1],
13                            [4, 2, 3, 5, 6, 3, 2, 4]]
14
15             self.white = {'pawn':1,
16                           "knight":2,
```



```
187         return 0;
188     def knight(x, y):
189         result = []
190         locations = [(x + 2, y - 1), (x + 2, y + 1), (x + 1, y - 2), (x + 1, y + 2),
191                     (x - 2, y - 1), (x - 2, y + 1), (x - 1, y - 2), (x - 1, y + 2)]
192
193         for loc in locations:
194             x_, y_ = loc;
195             if(x_ >= 8 or y_ >= 8 or x_ <= -1 or y_ <= -1):
196                 continue;
197             if(x_b(self.board[x_][y_]) == self.player):
198                 continue;
199             if(w_b(self.board[x_][y_]) == self.opposite_player()):
200                 result.append((x_, y_, 11));
```



```

203
204     return result;
205
206     def bishop(x, y):
207         result = [];
208         loop = 0;
209
210         while True:
211             x_ = x;
212             y_ = y;
213             stop = False;
214             while True:
215                 if(stop):
216                     break;
217                 if(loop == 0):
218                     x = x_ - 1;
219
220
221 Piece location: f8
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999

```

```

333
334     return result;
335
336     def queen(x, y):
337         result_bishop = bishop(x, y);
338         result_rock = rock(x, y);
339         result_king = king(x, y);
340
341         results = [];
342
343         for b in result_bishop:
344             results.append(b);
345         for r in result_rock:
346             results.append(r);
347         for k in result_king:
348             results.append(k);
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999

```

```

444
445     board[move[1][1]] = 0;
446     break;
447
448     elif(promote == "k" or promote == "knight"):
449         board[move[1][1]] = 2*m;
450     else:
451         print("\n -- Piece was not found | Try again | -- \n");
452         continue;
453
454     if(player == 1):
455         promote(0, 1);
456     if(player == 2):
457         promote(7, -1);
458
459
460 chess = game.play();
461 chess.start();
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999

```

מעקב אחר הוצאות (באמצעות Pandas)

בעולם מודרני שבו ניהול כספים הוא מכריע, התעמקתי ביצירת מעקב אחר הוצאות בגיל 19. הפרויקט הזה משתמש ב-Pandas, ספריית Python רבת עוצמה, כדי להקליט, לעקוב ולנתח נתונים פיננסיים לאורך חודשים או שנים. עסקאות נשמרות בקובץ CSV, מה שמבטיח התמדה של הנתונים לאורך הפעלות. המשתמש יכול בקלות לראות דוחות מפורטים, להבין את ההכנסות, ההוצאות ומדדים חיוניים אחרים עבור כל תקופה נתונה. עם 190 שורות קוד, כלי השירות הזה מציג את היישום המעשי של תכנות בתרחישים בעולם האמיתי.

date	food	entertainment	salary	others
19/09/2023 08:45	540	0	0	0
19/09/2023 08:55	0	-5000	0	0
19/09/2023 09:52	0	-5000	10000	0

```
36 file = open(os.path.join(self.store_data_folder, "object.txt"), "wb")
37 pickle.dump(obj.data, file)
38 def load_object(self):
39     file = open(os.path.join(self.store_data_folder, "object.txt"), "rb")
40     return pickle.load(file)
41 def save_history(self, obj):
42     try:
43         df = self.load_history()
44     except:
45         columns = ["data", "food", "entertainment", "salary", "others"]
46         df = pd.DataFrame(columns=columns)
47         df.to_csv(os.path.join(self.store_data_folder, "history.csv"), index=False)
48     df = pd.DataFrame(obj.data, index=[0])
49     df.to_csv(os.path.join(self.store_data_folder, "history.csv"), mode="a", index=False, header=False)
50 def load_history(self):
51     return pd.read_csv(os.path.join(self.store_data_folder, "history.csv"))
52
53 class Object():
54     ...
55
56 hon/Expense Tracker.py
57 What would you like to do? (Add/View/Report/Exit) Report
58 Enter the date (format: YYYY-MM) / (Today) / (None): today
59 ...
60
61 [*] Report for month: 2023-9
62 Expenses: 10000.0
63 Income: 20000.0
64
65 [-] Breakdown:
66 food 540.0
67 entertainment 10000.0
68 salary 10000.0
69 others 0.0
70
71 hon/Expense Tracker.py
72 What would you like to do? (Add/View/Report/Exit) Exit
```

```
165 df["date"] = pd.to_datetime(df["date"]);
166
167 if (year != None and month != None):
168     df = df[(df["date"].dt.month == month) & (df["date"].dt.year == year)];
169 else:
170     year = "all";
171     month = "";
172
173 print("\n---\n");
174 df = df.drop(columns=["date"]);
175
176 print(f"[*] Report for month: {year} {month}\n");
177 print(f"Expenses: {df[df < 0].sum().sum()}\n");
178 print(f"Income: {df[df > 0].sum().sum()}\n");
179 print(f"[-] Breakdown:\n");
180 print(df.sum().head());
181 print(f"\nTotal: {df[df < 0].sum().sum() + df[df > 0].sum().sum()}\n");
182 print("\n---\n");
183
184 self.default()
185 def __init__(self):
186     sys.exit();
187
188 data = Data();
189 data.default();
190
191 hon/Expense Tracker.py
192 What would you like to do? (Add/View/Report/Exit) Exit
```

מסקנה ויצירת קשר

במהלך המסע שלי עם Python, התמודדתי עם פרויקטים גדולים וקטנים כאחד, מסימולטורים של משחקים ועד מודלים מורכבים של רשתות עצביות. כל מאמץ חיזק את המומחיות שלי, לימד אותי גישות חדשות והעמיק את אהבתי לתכנות. הסמכה של PCAP, EF SET, TensorFlow Developer מדגישה את המחויבות שלי למלאכה ואת השאיפה שלי לשכלל את הכישורים שלי.

לבירורים, שיתוף פעולה או לבקשת הקוד המלא של כל הפרויקטים המפורטים בתיק זה, ניתן ליצור איתי קשר בכתובת Matans806@gmail.com. אני מוכן לעסוק בשיחות מעוררות, לקחת על עצמי אתגרים חדשים ולתרום לעולם הטכנולוגיה המתפתח ללא הרף.

תודה שהקדשת מזמנך לסקור את קורות החיים שלי לפרויקטים של Python.

ברכות חמות,

מתן חיים סנדורי.