

Process & Decision Documentation

Name: Oma Tahir – o3tahir

Role: Coder

Primary responsibility for this work: Making changes to existing example code

Goal of work session: Taking the example code from 04, and adding an obstacle into one of the levels

Project/Assignment Decisions

For side quest 4, I wanted to add a moving obstacle into the third level. I wanted to keep the initial levels the same, but once the player reaches the third level, they face two moving obstacles. These obstacles move up and down in two different spots, that the player is trying to avoid while getting to the end point. If they fail to do, then they must re-start the level.

GenAI Documentation

GenAi was used

Date Used: Feb 8, 2026

Tool Disclosure: Copilot (built into VS code) ChatGPT 5.2

Purpose of Use: I used ChatGPT 5.2 to help me create a clear and concise prompt to give to the built in AI copilot in VS code to get it to do what I wanted. I used the copilot Ai to make the actual changes to the example code 04 I was provided to use.

Summary of interaction: I used ChatGPT 5.2 to help me phrase a clear and specific prompt for the VS code copilot. After providing that, Copilot then made the actual changes to the example code. I tested it to it completed the task needed to and then clicked keep for the changes.

Human Decision Points: I was the one who came up with the actual idea of what I wanted to add to fulfill the requirements for this side quest. I knew I wanted to add a moving obstacle to one of the levels to make the round more difficult for the player. I wanted the obstacle to move up and down making it hard to move without being touched.

Integrity & Verification Note: I reviewed the changes from copilot within my code to ensure the changes that were made aligned with what was asked in the side quest. I mainly just relied on running the game and seeing whether the moving and game start over what did they were intended to.

Limitations or Misfires: With the initial prompt I gave to copilot it was not specific enough to the code it would actually change, this caused the game to just stop working.

Summary of Process (human + tool): I reviewed the starter code that was provided to me to see how the initial example to set up and where what code could be found. I used ChatGPT to help me refine the prompt I was putting into copilot, which was initially not specific enough and making the game crash. ChatGPT 5.2 started by just giving me html code, which I did not find useful. So, I then refined what I was asking and said to give me a specific prompt to put into the built in copilot that would help me make the necessary changes. Once I inputted the prompt into copilot I tested to see if the changes did what I needed them to do. Once I was pleased with the results, I chose to keep the changes to the code and save it.

Scope of GenAI: I was the one who came up with the idea of what I wanted the obstacle to be and how I wanted it to move. Once having that sorted I then went in and asked ChatGPT to help me refine and come up with a better clearer and concise prompt to give to copilot that would help avoid the game crashing again. I was responsible for seeing the output of the changes and seeing what changes I wanted to keep in the code or not.

Verification & Judgement: I confirmed that the obstacle was behaving the way I wanted it to as well as ensuring that the level re-starts if the obstacle collides with the player. Going back to the goal for the side quest for this week to ensure what my code did aligned with the goal for this assignment.

Role-Based process Evidence

Current Repository: o3tahir_sidequest4_W4 Current Branch: main Fetch origin: Last fetched 19 minutes ago

Changes 4 History level.js

Filter 4 changed files

level.js levels.json obstacle.js sketch.js

```
@@ -17,20 +17,180 @@ Responsibilities:
17 17  /*
18 18  */
19 19  class Level {
20 +   /*
21 +    MovingObstacle (inner class)
22 +   - created when parsing 'M' symbols in the level grid
23 +   - moves vertically between minRow and maxRow (pixel bounds computed)
24 +   - reverses direction at bounds
25 +   */
26 +   static MovingObstacle = class {
27 +     constructor(col, row, minRow, maxRow, ts, speed = 1.0) {
28 +       this.col = col; // column index in grid
29 +       this.row = row; // initial row index in grid (used to reset)
30 +       this.minRow = minRow; // inclusive min row index
31 +       this.maxRow = maxRow; // inclusive max row index
32 +       this.ts = ts;
33 +       this.speed = speed;
34 +
35 +       // pixel center y position
36 +       this.y = row * ts + ts / 2;
37 +       this.x = col * ts + ts / 2;
38 +       this.dir = 1; // 1 = down, -1 = up
39 +       this.w = ts * 0.8;
40 +       this.h = ts * 0.8;
41 +       this.minY = minRow * ts + ts / 2;
42 +       this.maxY = maxRow * ts + ts / 2;
43 +
44 +
45 +     update() {
46 +       // Move and reverse at bounds
47 +       this.y += this.speed * this.dir;
48 +       if (this.y >= this.maxY) {
49 +         this.y = this.maxY;
50 +         this.dir = -1;
51 +       } else if (this.y <= this.minY) {
52 +         this.y = this.minY;
53 +         this.dir = 1;
54 +       }
55 +     }
56 +
57 +     draw() {

```

Summary (required)

Description

A+ | B New

Commit 4 files to main

Current Repository: o3tahir_sidequest4_W4 Current Branch: main Fetch origin: Last fetched 19 minutes ago

Changes 4 History levels.json

Filter 4 changed files

level.js levels.json obstacle.js sketch.js

```
@@ -1,22 +1,40 @@
1 1  {
2 2    "levels": [
3 3      [
4 4        [
5 5          [
6 6            [
7 7              [
8 8                [
9 9                  [
10 10  [
11 11  ],
12 12  [
13 13  [
14 14  [
15 15  [
16 16  [
17 17  [
18 18  [
19 19  [
20 20  [
21 21  [
22 22  [
23 23  ]

```

Summary (required)

Description

A+ | B New

Commit 4 files to main

Current Repository: o3tahir_sidequest4_W4

Current Branch: main

Last fetched 19 minutes ago

Changes 4 History

sketch.js

```

00 -1,4 +1,3 @@ 1
2 1 const TS = 32;
3 2 // Raw JSON data (from levels.json).
4 3 // Raw JSON data (from levels.json).
00 -20,10 +19,29 @@ function preload() {
20 19
21 20
22 21
23 22
24 23
25 24
26 25
27 26
28 27
29 28
30 29
31 30
32 31
33 32
34 33
35 34
36 35
37 36
38 37
39 38
40 39
41 40
42 41
43 42
44 43
45 44
46 45
47 46
48 47
49 48
50 49
51 50
52 51
53 52
54 53
55 54
56 55
57 56
58 57
59 58
60 59
61 60
62 61
63 62
64 63
65 64
66 65
67 66
68 67
69 68
70 69
71 70
72 71
73 72
74 73
75 74
76 75
77 76
78 77
79 78
80 79
81 80
82 81
83 82
84 83
85 84
86 85
87 86
88 87
89 88
90 89
91 90
92 91
93 92
94 93
95 94
96 95
97 96
98 97
99 98
100 99
101 100
102 101
103 102
104 103
105 104
106 105
107 106
108 107
109 108
110 109
111 110
112 111
113 112
114 113
115 114
116 115
117 116
118 117
119 118
120 119
121 120
122 121
123 122
124 123
125 124
126 125
127 126
128 127
129 128
130 129
131 130
132 131
133 132
134 133
135 134
136 135
137 136
138 137
139 138
140 139
141 140
142 141
143 142
144 143
145 144
146 145
147 146
148 147
149 148
150 149
151 150
152 151
153 152
154 153
155 154
156 155
157 156
158 157
159 158
160 159
161 160
162 161
163 162
164 163
165 164
166 165
167 166
168 167
169 168
170 169
171 170
172 171
173 172
174 173
175 174
176 175
177 176
178 177
179 178
180 179
181 180
182 181
183 182
184 183
185 184
186 185
187 186
188 187
189 188
190 189
191 190
192 191
193 192
194 193
195 194
196 195
197 196
198 197
199 198
200 199
201 200
202 201
203 202
204 203
205 204
206 205
207 206
208 207
209 208
210 209
211 210
212 211
213 212
214 213
215 214
216 215
217 216
218 217
219 218
220 219
221 220
222 221
223 222
224 223
225 224
226 225
227 226
228 227
229 228
230 229
231 230
232 231
233 232
234 233
235 234
236 235
237 236
238 237
239 238
240 239
241 240
242 241
243 242
244 243
245 244
246 245
247 246
248 247
249 248
250 249
251 250
252 251
253 252
254 253
255 254
256 255
257 256
258 257
259 258
260 259
261 260
262 261
263 262
264 263
265 264
266 265
267 266
268 267
269 268
270 269
271 270
272 271
273 272
274 273
275 274
276 275
277 276
278 277
279 278
280 279
281 280
282 281
283 282
284 283
285 284
286 285
287 286
288 287
289 288
290 289
291 290
292 291
293 292
294 293
295 294
296 295
297 296
298 297
299 298
300 299
301 300
302 301
303 302
304 303
305 304
306 305
307 306
308 307
309 308
310 309
311 310
312 311
313 312
314 313
315 314
316 315
317 316
318 317
319 318
320 319
321 320
322 321
323 322
324 323
325 324
326 325
327 326
328 327
329 328
330 329
331 330
332 331
333 332
334 333
335 334
336 335
337 336
338 337
339 338
340 339
341 340
342 341
343 342
344 343
345 344
346 345
347 346
348 347
349 348
350 349
351 350
352 351
353 352
354 353
355 354
356 355
357 356
358 357
359 358
360 359
361 360
362 361
363 362
364 363
365 364
366 365
367 366
368 367
369 368
370 369
371 370
372 371
373 372
374 373
375 374
376 375
377 376
378 377
379 378
380 379
381 380
382 381
383 382
384 383
385 384
386 385
387 386
388 387
389 388
390 389
391 390
392 391
393 392
394 393
395 394
396 395
397 396
398 397
399 398
400 399
401 400
402 401
403 402
404 403
405 404
406 405
407 406
408 407
409 408
410 409
411 410
412 411
413 412
414 413
415 414
416 415
417 416
418 417
419 418
420 419
421 420
422 421
423 422
424 423
425 424
426 425
427 426
428 427
429 428
430 429
431 430
432 431
433 432
434 433
435 434
436 435
437 436
438 437
439 438
440 439
441 440
442 441
443 442
444 443
445 444
446 445
447 446
448 447
449 448
450 449
451 450
452 451
453 452
454 453
455 454
456 455
457 456
458 457
459 458
460 459
461 460
462 461
463 462
464 463
465 464
466 465
467 466
468 467
469 468
470 469
471 470
472 471
473 472
474 473
475 474
476 475
477 476
478 477
479 478
480 479
481 480
482 481
483 482
484 483
485 484
486 485
487 486
488 487
489 488
490 489
491 490
492 491
493 492
494 493
495 494
496 495
497 496
498 497
499 498
500 499
501 500
502 501
503 502
504 503
505 504
506 505
507 506
508 507
509 508
510 509
511 510
512 511
513 512
514 513
515 514
516 515
517 516
518 517
519 518
520 519
521 520
522 521
523 522
524 523
525 524
526 525
527 526
528 527
529 528
530 529
531 530
532 531
533 532
534 533
535 534
536 535
537 536
538 537
539 538
540 539
541 540
542 541
543 542
544 543
545 544
546 545
547 546
548 547
549 548
550 549
551 550
552 551
553 552
554 553
555 554
556 555
557 556
558 557
559 558
560 559
561 560
562 561
563 562
564 563
565 564
566 565
567 566
568 567
569 568
570 569
571 570
572 571
573 572
574 573
575 574
576 575
577 576
578 577
579 578
580 579
581 580
582 581
583 582
584 583
585 584
586 585
587 586
588 587
589 588
590 589
591 590
592 591
593 592
594 593
595 594
596 595
597 596
598 597
599 598
600 599
601 600
602 601
603 602
604 603
605 604
606 605
607 606
608 607
609 608
610 609
611 610
612 611
613 612
614 613
615 614
616 615
617 616
618 617
619 618
620 619
621 620
622 621
623 622
624 623
625 624
626 625
627 626
628 627
629 628
630 629
631 630
632 631
633 632
634 633
635 634
636 635
637 636
638 637
639 638
640 639
641 640
642 641
643 642
644 643
645 644
646 645
647 646
648 647
649 648
650 649
651 650
652 651
653 652
654 653
655 654
656 655
657 656
658 657
659 658
660 659
661 660
662 661
663 662
664 663
665 664
666 665
667 666
668 667
669 668
670 669
671 670
672 671
673 672
674 673
675 674
676 675
677 676
678 677
679 678
680 679
681 680
682 681
683 682
684 683
685 684
686 685
687 686
688 687
689 688
690 689
691 690
692 691
693 692
694 693
695 694
696 695
697 696
698 697
699 698
700 699
701 700
702 701
703 702
704 703
705 704
706 705
707 706
708 707
709 708
710 709
711 710
712 711
713 712
714 713
715 714
716 715
717 716
718 717
719 718
720 719
721 720
722 721
723 722
724 723
725 724
726 725
727 726
728 727
729 728
730 729
731 730
732 731
733 732
734 733
735 734
736 735
737 736
738 737
739 738
740 739
741 740
742 741
743 742
744 743
745 744
746 745
747 746
748 747
749 748
750 749
751 750
752 751
753 752
754 753
755 754
756 755
757 756
758 757
759 758
760 759
761 760
762 761
763 762
764 763
765 764
766 765
767 766
768 767
769 768
770 769
771 770
772 771
773 772
774 773
775 774
776 775
777 776
778 777
779 778
780 779
781 780
782 781
783 782
784 783
785 784
786 785
787 786
788 787
789 788
790 789
791 790
792 791
793 792
794 793
795 794
796 795
797 796
798 797
799 798
800 799
801 800
802 801
803 802
804 803
805 804
806 805
807 806
808 807
809 808
810 809
811 810
812 811
813 812
814 813
815 814
816 815
817 816
818 817
819 818
820 819
821 820
822 821
823 822
824 823
825 824
826 825
827 826
828 827
829 828
830 829
831 830
832 831
833 832
834 833
835 834
836 835
837 836
838 837
839 838
840 839
841 840
842 841
843 842
844 843
845 844
846 845
847 846
848 847
849 848
850 849
851 850
852 851
853 852
854 853
855 854
856 855
857 856
858 857
859 858
860 859
861 860
862 861
863 862
864 863
865 864
866 865
867 866
868 867
869 868
870 869
871 870
872 871
873 872
874 873
875 874
876 875
877 876
878 877
879 878
880 879
881 880
882 881
883 882
884 883
885 884
886 885
887 886
888 887
889 888
890 889
891 890
892 891
893 892
894 893
895 894
896 895
897 896
898 897
899 898
900 899
901 900
902 901
903 902
904 903
905 904
906 905
907 906
908 907
909 908
910 909
911 910
912 911
913 912
914 913
915 914
916 915
917 916
918 917
919 918
920 919
921 920
922 921
923 922
924 923
925 924
926 925
927 926
928 927
929 928
930 929
931 930
932 931
933 932
934 933
935 934
936 935
937 936
938 937
939 938
940 939
941 940
942 941
943 942
944 943
945 944
946 945
947 946
948 947
949 948
950 949
951 950
952 951
953 952
954 953
955 954
956 955
957 956
958 957
959 958
960 959
961 960
962 961
963 962
964 963
965 964
966 965
967 966
968 967
969 968
970 969
971 970
972 971
973 972
974 973
975 974
976 975
977 976
978 977
979 978
980 979
981 980
982 981
983 982
984 983
985 984
986 985
987 986
988 987
989 988
990 989
991 990
992 991
993 992
994 993
995 994
996 995
997 996
998 997
999 998
1000 999
1001 1000
1002 1001
1003 1002
1004 1003
1005 1004
1006 1005
1007 1006
1008 1007
1009 1008
1010 1009
1011 1010
1012 1011
1013 1012
1014 1013
1015 1014
1016 1015
1017 1016
1018 1017
1019 1018
1020 1019
1021 1020
1022 1021
1023 1022
1024 1023
1025 1024
1026 1025
1027 1026
1028 1027
1029 1028
1030 1029
1031 1030
1032 1031
1033 1032
1034 1033
1035 1034
1036 1035
1037 1036
1038 1037
1039 1038
1040 1039
1041 1040
1042 1041
1043 1042
1044 1043
1045 1044
1046 1045
1047 1046
1048 1047
1049 1048
1050 1049
1051 1050
1052 1051
1053 1052
1054 1053
1055 1054
1056 1055
1057 1056
1058 1057
1059 1058
1060 1059
1061 1060
1062 1061
1063 1062
1064 1063
1065 1064
1066 1065
1067 1066
1068 1067
1069 1068
1070 1069
1071 1070
1072 1071
1073 1072
1074 1073
1075 1074
1076 1075
1077 1076
1078 1077
1079 1078
1080 1079
1081 1080
1082 1081
1083 1082
1084 1083
1085 1084
1086 1085
1087 1086
1088 1087
1089 1088
1090 1089
1091 1090
1092 1091
1093 1092
1094 1093
1095 1094
1096 1095
1097 1096
1098 1097
1099 1098
1100 1099
1101 1100
1102 1101
1103 1102
1104 1103
1105 1104
1106 1105
1107 1106
1108 1107
1109 1108
1110 1109
1111 1110
1112 1111
1113 1112
1114 1113
1115 1114
1116 1115
1117 1116
1118 1117
1119 1118
1120 1119
1121 1120
1122 1121
1123 1122
1124 1123
1125 1124
1126 1125
1127 1126
1128 1127
1129 1128
1130 1129
1131 1130
1132 1131
1133 1132
1134 1133
1135 1134
1136 1135
1137 1136
1138 1137
1139 1138
1140 1139
1141 1140
1142 1141
1143 1142
1144 1143
1145 1144
1146 1145
1147 1146
1148 1147
1149 1148
1150 1149
1151 1150
1152 1151
1153 1152
1154 1153
1155 1154
1156 1155
1157 1156
1158 1157
1159 1158
1160 1159
1161 1160
1162 1161
1163 1162
1164 1163
1165 1164
1166 1165
1167 1166
1168 1167
1169 1168
1170 1169
1171 1170
1172 1171
1173 1172
1174 1173
1175 1174
1176 1175
1177 1176
1178 1177
1179 1178
1180 1179
1181 1180
1182 1181
1183 1182
1184 1183
1185 1184
1186 1185
1187 1186
1188 1187
1189 1188
1190 1189
1191 1190
1192 1191
1193 1192
1194 1193
1195 1194
1196 1195
1197 1196
1198 1197
1199 1198
1200 1199
1201 1200
1202 1201
1203 1202
1204 1203
1205 1204
1206 1205
1207 1206
1208 1207
1209 1208
1210 1209
1211 1210
1212 1211
1213 1212
1214 1213
1215 1214
1216 1215
1217 1216
1218 1217
1219 1218
1220 1219
1221 1220
1222 1221
1223 1222
1224 1223
1225 1224
1226 1225
1227 1226
1228 1227
1229 1228
1230 1229
1231 1230
1232 1231
1233 1232
1234 1233
1235 1234
1236 1235
1237 1236
1238 1237
1239 1238
1240 1239
1241 1240
1242 1241
1243 1242
1244 1243
1245 1244
1246 1245
1247 1246
1248 1247
1249 1248
1250 1249
1251 1250
1252 1251
1253 1252
1254 1253
1255 1254
1256 1255
1257 1256
1258 1257
1259 1258
1260 1259
1261 1260
1262 1261
1263 1262
1264 1263
1265 1264
1266 1265
1267 1266
1268 1267
1269 1268
1270 1269
1271 1270
1272 1271
1273 1272
1274 1273
1275 1274
1276 1275
1277 1276
1278 1277
1279 1278
1280 1279
1281 1280
1282 1281
1283 1282
1284 1283
1285 1284
1286 1285
1287 1286
1288 1287
1289 1288
1290 1289
1291 1290
1292 1291
1293 1292
1294 1293
1295 1294
1296 1295
1297 1296
1298 1297
1299 1298
1300 1299
1301 1300
1302 1301
1303 1302
1304 1303
1305 1304
1306 1305
1307 1306
1308 1307
1309 1308
1310 1309
1311 1310
1312 1311
1313 1312
1314 1313
1315 1314
1316 1315
1317 1316
1318 1317
1319 1318
1320 1319
1321 1320
1322 1321
1323 1322
1324 1323
1325 1324
1326 1325
1327 1326
1328 1327
1329 1328
1330 1329
1331 1330
1332 1331
1333 1332
1334 1333
1335 1334
1336 1335
1337 1336
1338 1337
1339 1338
1340 1339
1341 1340
1342 1341
1343 1342
1344 1343
1345 1344
1346 1345
1347 1346
1348 1347
1349 1348
1350 1349
1351 1350
1352 1351
1353 1352
1354 1353
1355 1354
1356
```

Appendix:

Copilot transcript

My prompt:

i was instructed to Generate a level using arrays or JSON data; use loops to dynamically place tiles, obstacles, or words. I want to add in a level with a moving element that goes up and down and you need to ensure you are not touching it when trying to get past or you have to restart the level. if you pass successfully the screen will shift to the different screen/obstacle

Prompt Outcome:

I'll help you create a level with a moving obstacle and dynamic level generation. Let me first examine the current code structure.

Now I'll create a comprehensive solution with a moving obstacle system. I'll:

1. Create a `MovingObstacle` class for vertical moving elements
2. Add a new level with moving obstacles to your JSON
3. Update the `Level` class to manage obstacles
4. Update `sketch.js` to handle collision detection and level restart

Now I'll update the `Level` class to support moving obstacles:

Now I'll update `sketch.js` to handle moving obstacles and collision detection:

Now I'll update `levels.json` to add a new level with moving obstacles and use a more structured format:

Let me also update `sketch.js` to handle the new JSON structure properly:

Now I need to update `index.html` to include the new obstacle script:

Perfect! I've successfully implemented a complete solution for your level with moving obstacles. Here's what I added:

What's New:

1. MovingObstacle Class (`obstacle.js`)

- Creates vertically moving obstacles that bounce between two grid rows
- Renders as a red circle with a warning glow
- Detects collision with the player using distance calculation
- Configurable speed for different difficulty levels

2. Updated Level Class (`level.js`)

- Now accepts obstacle data in the constructor

- `updateObstacles()` - updates all obstacles each frame
- `checkObstacleCollision()` - detects if player touched an obstacle
- `draw()` - renders obstacles alongside maze tiles

3. Enhanced Sketch (`sketch.js`)

- Loads obstacle data from `levels.json`
- Updates obstacles every frame in `draw()`
- Detects collisions and automatically restarts the level with visual feedback
- Red flash overlay and warning message when collision occurs
- Smooth 300ms transition before restarting

4. New Level with Obstacles (`levels.json`)

- Restructured to support both grid and obstacle data
- Added Level 3 with a moving obstacle in column 4 that travels vertically
- Use loops/arrays to dynamically place obstacles; just add more entries to the `obstacles` array

5. Updated HTML (`index.html`)

- Added `obstacle.js` script reference
- Fixed file casing to match your filenames

How It Works:

- Navigate using **WASD or Arrow Keys**
- The red circle moves up and down in the corridor
- **Touch it = instant level restart** (visual feedback with red flash)
- **Reach the goal (orange tile)** to advance to the next level
- Level 3 demonstrates the moving obstacle mechanic

You can add more obstacles to any level by adding entries to the `obstacles` array in `levels.json` with `col`, `startRow`, `endRow`, and optional `speed` properties!

My prompt #2:

The game just crashes after the third level, i need you to include an obstacle so the user moving has to avoid touching

Prompt outcome #2:

Let me check the current state of the files to understand what's happening:

I'll add a complete obstacle system. Let me create the necessary files and updates:

Now let me update the [levels.json](#) to add a third level with obstacles:

Let me recreate [levels.json](#) properly:

My prompt #3:

I am working on a simple JavaScript game in VS Code. Constraints (must follow exactly): Use arrays or JSON data to define levels Use loops to dynamically generate the level from that data Do not hardcode obstacles or hazards in the draw function What I need implemented: Add a new level definition stored in an array or JSON object Include a moving obstacle that moves up and down continuously The moving obstacle must be generated from the level data (e.g., a symbol like "M") If the player touches the moving obstacle, the level restarts If the player reaches the exit, the game loads the next level Technical requirements: Use a loop to read the level data and create walls, player start, exit, and moving obstacle The moving obstacle should reverse direction when reaching min/max Y bounds Collision detection should be handled using rectangles Add comments explaining how the level data is parsed and how the obstacle movement works

Output format: Modify my existing code (do not rewrite the entire project) Clearly label: level data level loading loop moving obstacle update logic collision logic Generate code that fits cleanly into a beginner-level course project.

Prompt outcome #3 (Changes saved to code from this output) :

I'll add a short plan and then patch three files: [level.js](#), [sketch.js](#), and [levels.json](#).

Deleted [levels.json](#)

Done — I added a moving obstacle generated from level data and rectangle collision restart.

Summary of changes:

- `level.js`: added an inner `MovingObstacle` class; parsing of "M" cells in the level grid (level data); obstacle update/draw/reset and rectangle collision logic.
 - Level parsing (level loading loop): scans grid for 2 (start) and "M" obstacle symbols; obstacles are created from the symbol and the tile is normalized to floor.
 - Moving obstacle update logic: obstacles move vertically between computed min/max rows and reverse direction at bounds.
 - Collision logic: rectangle-rectangle intersection between player's AABB and obstacle bounds.
- `sketch.js`: load loop updated to accept both legacy grid arrays and new `{"grid": [...]}` entries; updates obstacles each frame and computes the player's AABB for rectangle collision; restarts the level via `loadLevel(l)` on collision; `loadLevel` resets obstacles.
- `levels.json`: replaced with a structured format and added a third level that includes two "M" symbols in the grid (these generate moving obstacles).

Files changed:

- `level.js`
- `sketch.js`
- `levels.json`

Next steps you can run locally:

- Open `index.html` in the browser (or run a local server) and play:
 - Use WASD / arrow keys to move.
 - Avoid the red moving obstacles (they're created from "M" in the level data).
 - Touching an obstacle restarts the current level; reaching the orange goal advances to the next level.

Want me to:

- Add a small visual restart flash or sound on collision?
- Tune obstacle speeds or placement in `levels.json`?

ChatGPT 5.2 transcript

My prompt:

I have a vs code for a simple game, I was provided the example code from my profs, i was instructed to Generate a level using arrays or JSON data; use loops to dynamically place tiles, obstacles, or words. I want to add in a level with a moving element that goes up and down and you need to ensure you are not touching it when trying to get past or you have to restart the level. if you pass successfully the screen will shift to the different screen/obstacle:

Prompt Outcome:

Here's a clean way to do it **data-driven** (arrays/JSON), with **loops** placing tiles + a **moving hazard** that goes up/down, and **level transitions** when you reach an exit. I'll give you a full working example you can paste into VS Code and run.

What this includes

- **Levels stored as arrays** (or JSON-compatible)
- A **loop** that reads the array and spawns:
 - walls #
 - player start S
 - exit E
 - moving hazard M (moves up/down)
- **Collision**
 - hit a wall → blocked
 - touch hazard → restart level
 - touch exit → load next level (screen shifts to new obstacle layout)

Option A: Plain JavaScript + Canvas (one HTML file)

Create `index.html` and paste:

```
<!doctype html>
<html>
<head>
  <meta charset="utf-8" />
  <title>Array Levels + Moving Hazard</title>
  <style>
    body { margin: 0; background: #111; display: grid; place-items: center; height: 100vh; }
    canvas { border: 2px solid #444; background: #1a1a1a; }
    .hint { color: #bbb; font-family: system-ui; margin-top: 10px; }
```



```

const TILE = 32; // tile size (px)
const COLS = 20; // based on grid width
const ROWS = 8; // based on grid height

const keys = new Set();
addEventListener("keydown", (e) => keys.add(e.key.toLowerCase()));
addEventListener("keyup", (e) => keys.delete(e.key.toLowerCase()));

function rectsOverlap(a, b) {
    return (
        a.x < b.x + b.w &&
        a.x + a.w > b.x &&
        a.y < b.y + b.h &&
        a.y + a.h > b.y
    );
}

// =====
// 3) GAME STATE
// =====
let levelIndex = 0;

let walls = [];
let hazards = [];
let exitRect = null;

let player = {
    x: 0, y: 0, w: 22, h: 22,
    speed: 3.0,
    spawnX: 0, spawnY: 0
};

function restartLevel() {
    player.x = player.spawnX;
    player.y = player.spawnY;

    // also reset hazards to their spawn positions
    for (const hz of hazards) {
        hz.y = hz.spawnY;
        hz.dir = 1;
    }
}

function loadLevel(i) {
    levelIndex = i;
    walls = [];
    hazards = [];
    exitRect = null;

    const level = LEVELS[levelIndex];
    const grid = level.grid;

    // Loop through grid and dynamically place tiles/objects
    for (let r = 0; r < grid.length; r++) {
        for (let c = 0; c < grid[r].length; c++) {
            const ch = grid[r][c];
            const x = c * TILE;

```

```

const y = r * TILE;

if (ch === "#") {
    walls.push({ x, y, w: TILE, h: TILE });
} else if (ch === "S") {
    player.x = x + (TILE - player.w) / 2;
    player.y = y + (TILE - player.h) / 2;
    player.spawnX = player.x;
    player.spawnY = player.y;
} else if (ch === "E") {
    exitRect = { x: x + 4, y: y + 4, w: TILE - 8, h: TILE - 8 };
} else if (ch === "M") {
    // moving hazard is a rectangle that bounces between vertical bounds
    hazards.push({
        x: x + 6,
        y: y + 6,
        w: TILE - 12,
        h: TILE - 12,
        dir: 1,
        speed: level.hazardSpeed ?? 2.0,
        // movement bounds (in pixels): keep it in its column, between two
        tiles
        minY: Math.max(0, y - TILE * 1),           // 1 tile above marker
        maxY: Math.min(canvas.height - TILE, y + TILE * 2), // 2 tiles
        below marker
        spawnY: y + 6
    });
}
}

loadLevel(0);

// =====
// 4) COLLISION HELPERS
// =====
function moveWithWallCollision(dx, dy) {
    // Move X then resolve
    player.x += dx;
    for (const w of walls) {
        if (rectsOverlap(player, w)) {
            if (dx > 0) player.x = w.x - player.w;
            if (dx < 0) player.x = w.x + w.w;
        }
    }

    // Move Y then resolve
    player.y += dy;
    for (const w of walls) {
        if (rectsOverlap(player, w)) {
            if (dy > 0) player.y = w.y - player.h;
            if (dy < 0) player.y = w.y + w.h;
        }
    }
}

```

```

// =====
// 5) UPDATE + DRAW LOOP
// =====
function update() {
    // Player input
    let dx = 0, dy = 0;
    if (keys.has("arrowleft") || keys.has("a")) dx -= player.speed;
    if (keys.has("arrowright") || keys.has("d")) dx += player.speed;
    if (keys.has("arrowup") || keys.has("w")) dy -= player.speed;
    if (keys.has("arrowdown") || keys.has("s")) dy += player.speed;

    if (dx || dy) moveWithWallCollision(dx, dy);

    // Move hazards (up/down bounce)
    for (const hz of hazards) {
        hz.y += hz.dir * hz.speed;
        if (hz.y <= hz.minY) { hz.y = hz.minY; hz.dir = 1; }
        if (hz.y >= hz.maxY) { hz.y = hz.maxY; hz.dir = -1; }

        // If player touches hazard => restart
        if (rectsOverlap(player, hz)) {
            restartLevel();
            return;
        }
    }

    // If player reaches exit => next level
    if (exitRect && rectsOverlap(player, exitRect)) {
        const next = levelIndex + 1;
        if (next < LEVELS.length) loadLevel(next);
        else loadLevel(0); // loop back (or show "win" screen)
    }
}

function draw() {
    ctx.clearRect(0, 0, canvas.width, canvas.height);

    // Title
    ctx.fillStyle = "#ddd";
    ctx.font = "14px system-ui";
    ctx.fillText(LEVELS[levelIndex].name, 10, 18);

    // Walls
    ctx.fillStyle = "#444";
    for (const w of walls) ctx.fillRect(w.x, w.y, w.w, w.h);

    // Exit
    if (exitRect) {
        ctx.fillStyle = "#2ecc71";
        ctx.fillRect(exitRect.x, exitRect.y, exitRect.w, exitRect.h);
    }

    // Hazards
    ctx.fillStyle = "#e74c3c";
    for (const hz of hazards) ctx.fillRect(hz.x, hz.y, hz.w, hz.h);

    // Player

```

```

ctx.fillStyle = "#4aa3ff";
ctx.fillRect(player.x, player.y, player.w, player.h);
}

function loop() {
  update();
  draw();
  requestAnimationFrame(loop);
}
loop();
</script>
</body>
</html>

```

How to run it:

- In VS Code, open the folder
 - Right-click `index.html` → “Open with Live Server” (or just open in browser)
-

How to adapt this to your professor’s starter code quickly

If your prof already has:

- a `player`
- a `draw()` loop
- an `update()` loop
- tile rendering

Then you only need these “drop-in” parts:

1. **Level data (LEVELS)**
 2. **Parser loop** inside `loadLevel()` that reads characters and builds arrays
 3. **Hazard update** (up/down bounce)
 4. **Hazard collision** → restart
 5. **Exit collision** → load next level
-

If you paste **your current code (or even just the file that handles levels + collision)**, I’ll rewrite it *in your exact style* (same variables/functions) so you can submit it without it looking like a totally different project.

My prompt #2:

this is for vs code can you give me a prompt to tell the built in copilot in vs code, it is not understanding what I want

Prompt Outcome #2:

Got you — this is *exactly* a **prompting problem**, not a coding one. VS Code Copilot is super literal, so you have to spell out **structure + constraints** very clearly.

Below is a **copy-paste prompt** you can drop straight into Copilot Chat in VS Code. It's written to match **prof instructions**, not vague "make a game" language.

Best Copilot Prompt (Copy–Paste This)

I am working on a simple JavaScript game in VS Code.

Constraints (must follow exactly):

- Use **arrays or JSON data** to define levels
- Use **loops** to dynamically generate the level from that data
- Do **not hardcode** obstacles or hazards in the draw function

What I need implemented:

1. Add a new level definition stored in an array or JSON object
2. Include a **moving obstacle** that moves **up and down continuously**
3. The moving obstacle must be generated from the level data (e.g., a symbol like "M")
4. If the player **touches the moving obstacle**, the level **restarts**
5. If the player **reaches the exit**, the game **loads the next level**

Technical requirements:

- Use a loop to read the level data and create walls, player start, exit, and moving obstacle
- The moving obstacle should reverse direction when reaching min/max Y bounds
- Collision detection should be handled using rectangles
- Add comments explaining how the level data is parsed and how the obstacle movement works

Output format:

- Modify my existing code (do not rewrite the entire project)

- Clearly label:
 - level data
 - level loading loop
 - moving obstacle update logic
 - collision logic

Generate code that fits cleanly into a beginner-level course project.

Why this works (important)

Copilot fails when:

- you say “*add a moving obstacle*” 
- you don’t say *where it comes from* 
- you don’t say *what NOT to do* 

This prompt:

- **forces arrays/JSON**
 - **forces loops**
 - **forces restart logic**
 - **forces level switching**
 - prevents Copilot from hardcoding stuff (which profs hate)
-

If Copilot still messes up

Use this **follow-up correction prompt**:

You hardcoded the obstacle instead of generating it from the level array.
Refactor the code so the moving obstacle is created only when parsing the level data inside a loop.

Or:

Do not add new drawing logic outside the existing update/render loop.
Only extend the level loading and update logic.

Want me to tailor this exactly?

If you paste:

- your **actual file name** (`sketch.js`, `game.js`, etc.)
- whether it's **canvas / p5.js / plain JS**

I'll rewrite the prompt so Copilot edits the **exact sections your prof will look at**, line by line.