

Assessment Task 1 – Planning and Problem Solving

Requirements Definition:

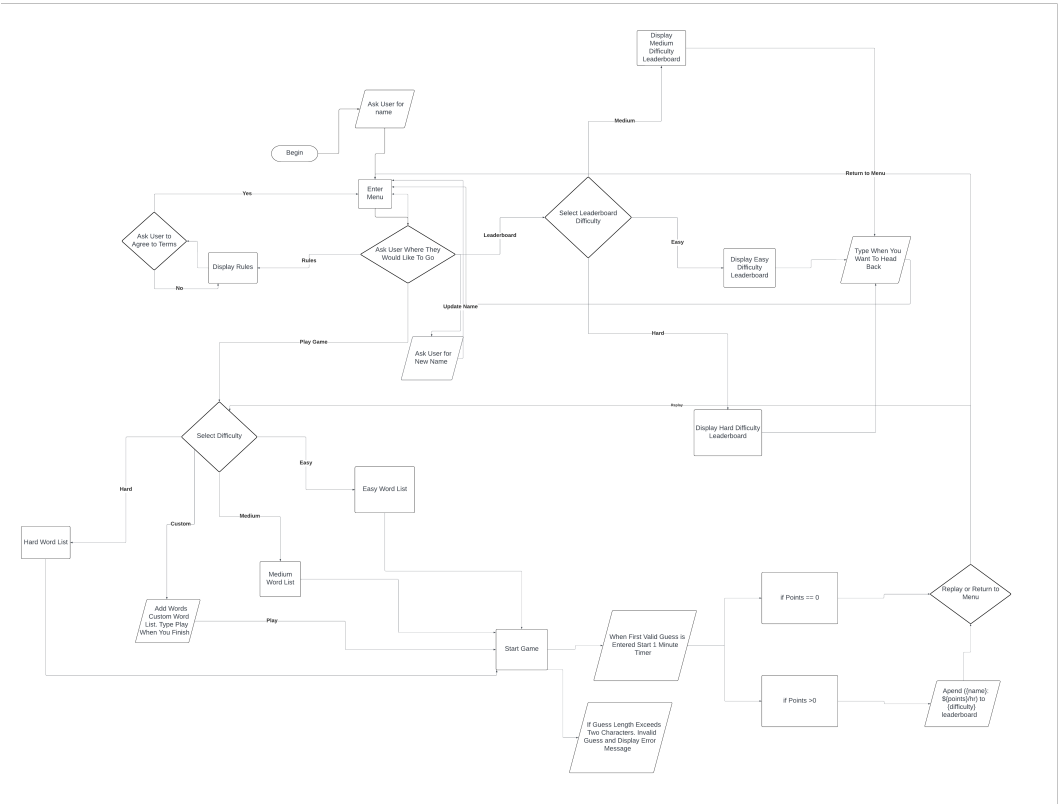
The game must meet various functional and non-functional requirements to ensure a hassle free and enjoyable experience for the player.

The most vital functional requirement is that of the game which must run properly and consistency to ensure fair results. Another functional requirement is to make a main menu system to enable players to navigate through various menus in the game. Additionally, the game folder must contain all necessary external files to ensure the game does not run into errors. Furthermore, the game must incorporate different wordlists for corresponding difficulties that are returned to the wordlist variable once the player has selected their difficulty. Finally, a leader board must be implemented to allow players to track and compare their progress to their peers.

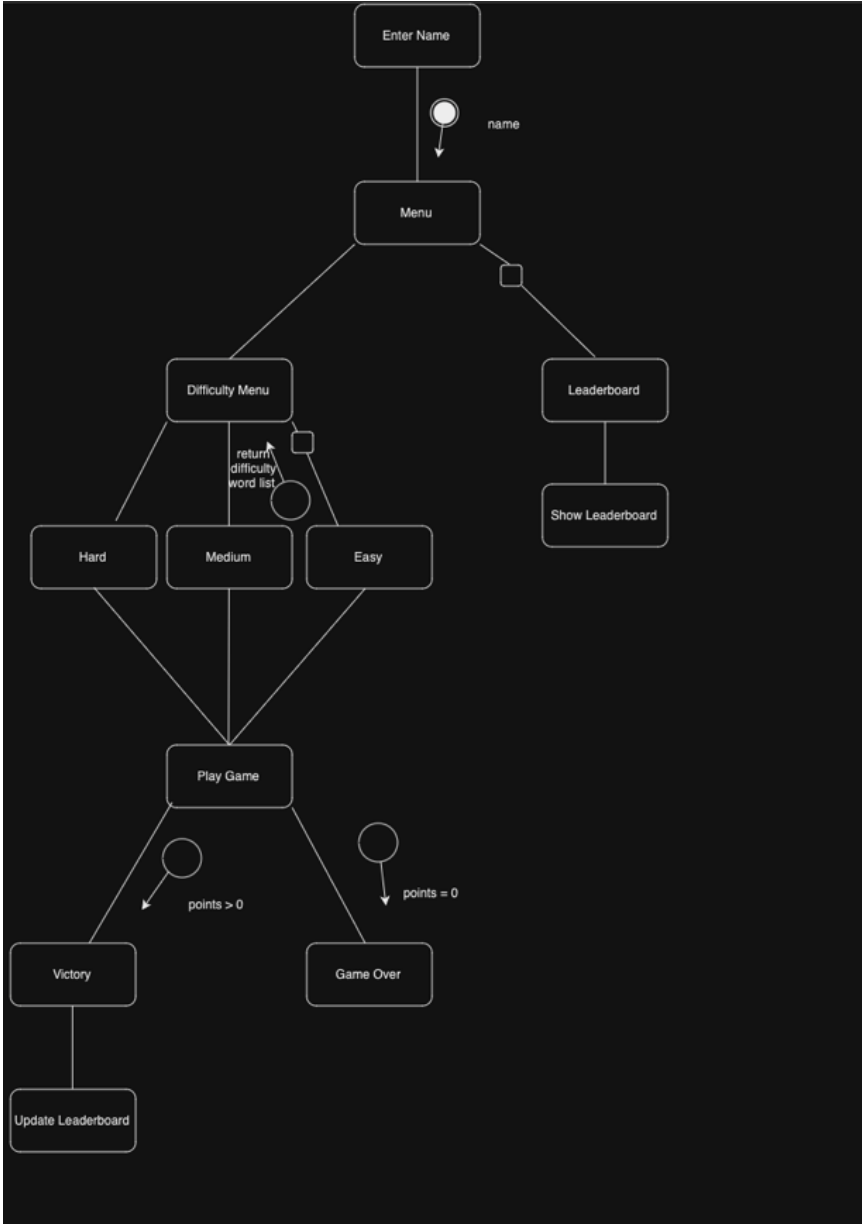
Non-functional requirements are pivotal for ensuring a great experience for the player. A graphical display presenting the number of incorrect guesses in the game helps immerse the player in the game. Additionally, both coloured text and background music help draw the user to game by reducing the bland user interface that would usually be associated with a text-based game. Finally, a visual timer allows the player to keep track of how much time they have remaining after each guess to ensure they take appropriate measures.

In conclusion there are multiple functional and non-functional requirements that should be considered when approaching this project.

Design:
Algorithm –



Structure Chart –

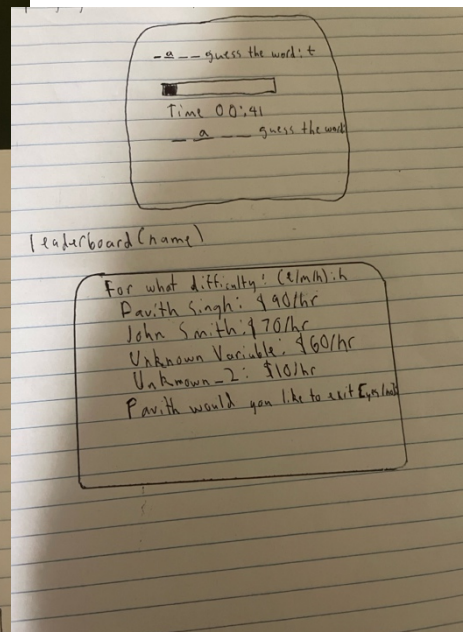
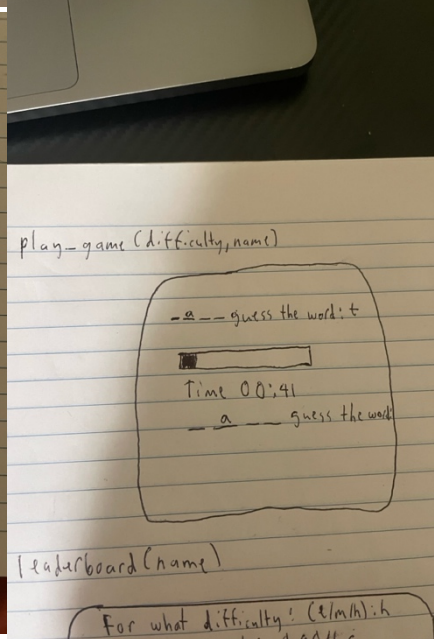
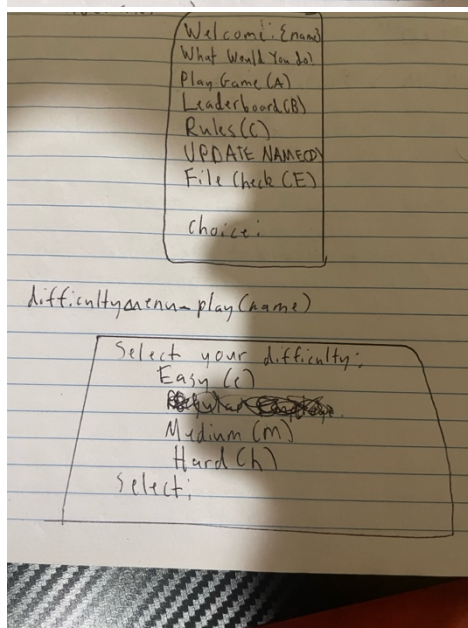
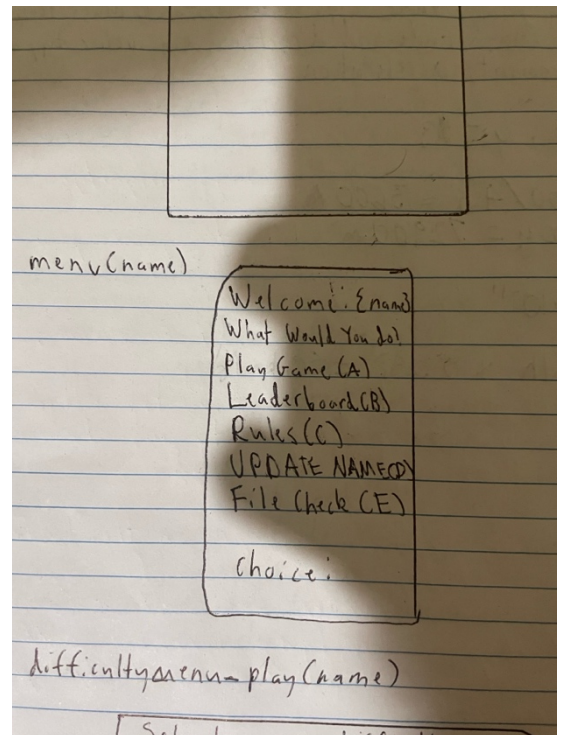
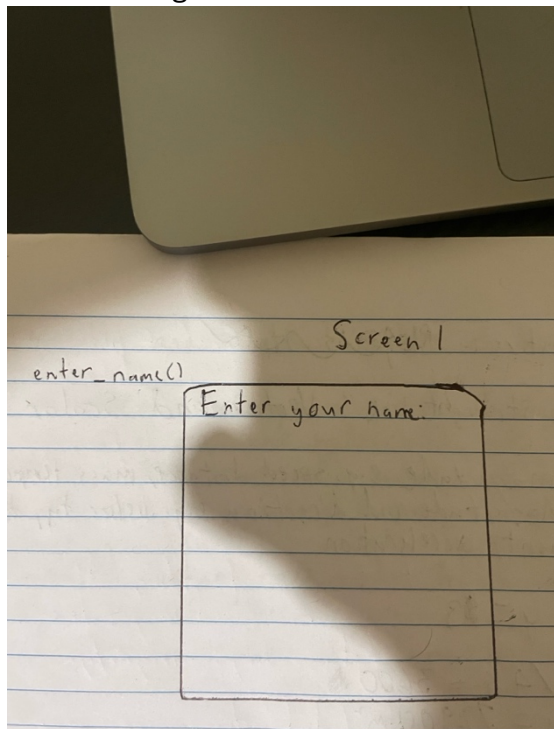


Data Dictionary –

Variable	Data Type	Data Format	Description	Example
name	String	'aaaa'	The name associated with the player	'Pavith Singh'
difficulty	String	'a', 'b', 'c',...	The difficulty chosen by the player	'Easy', 'Medium', 'Hard', 'Custom'
wordlist	List	['aaaa', 'aaab', 'aaba',...]	A list of words for the difficulty level	['truck', 'ducks', 'frogs,...]
randomword	String	'aaaa'	A randomly selected word from wordlist	'truck'
guesses	string	'a', 'b', 'c'...	A string of characters the player has guessed	't', 'r', 'u', 'c', 'k'
guess	String	'a'	A character that the player has guessed	't'
points	Integer	nnnn	The player's score	90
badguesses	Integer	nnnn	The number of incorrect guesses made in a game in increments of 10	10
start_time	Float	n:nnn	The time at which the player typed the first letter	0
end_time	Float	n:nnn	The time at which the game will end which is exactly 1 minute after the start_time	1:00
timecount	Float	n:nnn	A count of the remaining time after each guess	0:39
pay_rate	String	\$nnnn/hr	The player's scored displayed as a an hourly wage	\$90/hr
leaderboard_position	String	'a', 'b', 'c'	The difficulty for which the player	'easy', 'medium', 'hard'

			wants to view the leaderboard for	
data	List	['aaaa: \$nnnn/hr', 'aaab: \$nnno/hr',...]	The information in the text file of each difficulty	['Pavith Singh: \$90/hr', 'Steve John: \$100/hr']
pay_rates	List	['\$nnnn/hr', '\$nnno/hr',...]	The pay rates extracted from the text file	['\$90/hr', '\$100/hr']
sorted_pay_rates	List	['\$nnnn/hr', '\$nnno/hr',...]	The pay rates sorted in descending order	['\$100/hr', '\$90/hr']
rate	List	'aaaa: \$nnnn/hr' 'aaab: \$nnno/hr'...	The information in the text file in descending order in terms of pay rates.	'Steve John: \$100/hr' Pavith Singh: \$90/hr'
back	String	'Y' or 'N'	The player's response to the rules	'Yes' or 'No'
customlist	List	['aaaa', 'aaab', 'aaba',...]	A version of the wordlist, where the player chooses the word	['twentythreehundred', 'generational', 'two']
game_over	String	'Y' or 'N'	The player deciding where to go after the game over	'Yes' or 'No'
game_over1	String	'Y' or 'N'	The player deciding where to go after the game over	'Yes' or 'No'
goback	String	'Y' or 'N'	A response to whether the player wants to return to the menu	'Yes' or 'No'
menu_choice	String	'a', 'b', 'c'...	A choice of where the player wants to go from the main menu	'a', 'b', 'c', 'd', 'e', 'f'
new_name	String	'aaaa'	A new name for the player replacing the old one	'Steve John'

Screen Designs –



Testing/Errors:

Difficulty Selection –

Input	Output
i	Return Wordlist(Intern) Call play_game(difficulty, name)
l	Return Wordlist(Intern) Call play_game(difficulty, name)
r	Return Wordlist(Regular Employee) Call play_game(difficulty, name)
R	Return Wordlist(Regular Employee) Call play_game(difficulty, name)
\$	Return Wordlist(Executive) Call play_game(difficulty, name)
c	Return Customlist Call play_game(difficulty, name)
C	Return Customlist Call play_game(difficulty, name)
a	Call difficultymenu_play(name)
0	Call difficultymenu_play(name)

Encountered Error

```
116 def play_game(difficulty, name):
117     ...
203     menu(name)
204     if game_over == 'yes':
205         difficultymenu_play(name)
206
207 def difficultymenu_play(name):
208     print('\n\033[30m Select your position in the compnay')
209     print('\033[2m Intern (I)')
210     print('\033[3m Regular Employee (R)')
211     print('\033[32m Executive ($)')
212     print('\033[33m Custom List (C)')
213     difficulty = input('\033[0m\033[93mSelect: ').upper()
214     if difficulty == 'I' or 'R' or '$' or 'C':
215         play_game(difficulty, name)
216     else:
217         difficultymenu_play(name)
218
219 def leaderboard(name):
220     leaderboard_positon = input('For what position would you like to view the leaderboard: (I/R/$): ').upper()
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

choice: a

Select your position in the compnay
Intern (I)
Regular Employee (R)
Executive (\$)
Custom List (C)

Select: 0

Traceback (most recent call last):
File "/Users/pavithpreetsingh/Desktop/Year 11/Software Engineering/CodingStuff/ye remix.py", line 313, in <module>
enter_name()
File "/Users/pavithpreetsingh/Desktop/Year 11/Software Engineering/CodingStuff/ye remix.py", line 9, in enter_name
menu(name)
File "/Users/pavithpreetsingh/Desktop/Year 11/Software Engineering/CodingStuff/ye remix.py", line 256, in menu
difficultymenu_play(name)
File "/Users/pavithpreetsingh/Desktop/Year 11/Software Engineering/CodingStuff/ye remix.py", line 215, in difficultymenu_play
play_game(difficulty, name)
File "/Users/pavithpreetsingh/Desktop/Year 11/Software Engineering/CodingStuff/ye remix.py", line 118, in play_game
randomword = random.choice(wordlist)
~~~~~  
File "/Library/Frameworks/Python.framework/Versions/3.11/Lib/python3.11/random.py", line 372, in choice  
if not len(seq):  
TypeError: object of type 'NoneType' has no len()

○ pavithpreetsingh@Tim-Cooks-MacBook-Pro CodingStuff %

This error occurs when you enter a character other than the four that are assigned to the wordlist. To fix this I instead changed the code to check if the input was not in a list containing the four characters.



Fixed version

```

123 def play_game(difficulty, name):
220     if game_over1 == 'yes':
221         difficultymenu_play(name)
222
223 def difficultymenu_play(name):
224     # allows player to select which level of difficulty they want. This will mean different word lists depending on player choice
225     print('\n\033[30m Select your position in the companay')
226     print('\033[2m Intern (I)')
227     print('\033[3m Regular Employee (R)')
228     print('\033[32m Executive ($)')
229     print('\033[33m Custom List (C) ')
230     difficulty = input('\033[0m\033[93mSelect: ').upper()
231     if difficulty not in ['I', 'R', '$', 'C']:
232         difficultymenu_play(name)
233     else:
234         play_game(difficulty, name)
235
236 def leaderboard(name):
237     # sorts the pay rates from the text files in descending order and then displays them
238     leaderboard_positon = input('For what position would you like to view the leaderboard: (I/R/$): ').upper()

```

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS

COMMENTS

Python

Play game (A)

Leaderboard (B)

Rules (C)

Update name (D)

File Check (E)

choice: a

Select your position in the companay

Intern (I)

Regular Employee (R)

Executive (\$)

Custom List (C)

Select: 0

Select your position in the companay

Intern (I)

Regular Employee (R)

Executive (\$)

Custom List (C)

Select:

Leader board –

| Input | Output                                                   |
|-------|----------------------------------------------------------|
| I     | Print Sorted Leaderboard for Intern Difficulty           |
| i     | Print Sorted Leaderboard for Intern Difficulty           |
| R     | Print Sorted Leaderboard for Regular Employee Difficulty |
| r     | Print Sorted Leaderboard for Regular Employee Difficulty |
| E     | Print Sorted Leaderboard for Executive Difficulty        |
| e     | Print Sorted Leaderboard for Executive Difficulty        |

Encountered Error



## Input

**Word**

## Output

|      |      |                                             |
|------|------|---------------------------------------------|
| a    | bird | __                                          |
| i    | bird | _i_                                         |
| R    | bird | __r_                                        |
| bird | bird | Error: guess can only contain one character |

## Encountered Problem

```
10         return ['differentiation', 'coordination', 'humongous', 'environmentally', 'sustainability', 'penultimate', 'destabilization']
11
12     Codium: Refactor | Explain | Generate Docstring | X
13     def play_game(dingle):
14         wordlist = get_word_list(dingle)
15         randomword = random.choice(wordlist)
16         guesses = ''
17         points = 100
18         while points > 0:
19             badguesses = 0
20             for character in randomword:
21                 if character in guesses:
22                     print(f'{character}')
23                 else:
24                     # Thanks to Sevan for assisting me in this
25                     badguesses += 10
26             print('_', end = ' ')
27
28             if badguesses == 0:
29                 print(f'You saved everyone from the bomb you have been awarded ${points} for the job. I hope you enjoyed it.')
30                 print(f'Word: {randomword}')
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
100
```

### Very Early Version of the game

The problem being encountered here is that the all the lines after the guessed letter are down one line. To fix this issue I added `end = ' '` to the end of `print(f'{character}')` to replace the new line with a space.

Fixed Version:

```
10     return ['differentiation', 'coordination', 'humongous', 'environmentally', 'sustainability', 'penultimate', 'destab  
11  
Codeium: Refactor | Explain | Generate Docstring | X  
12 def play_game(dingle):  
13     wordlist = get_word_list(dingle)  
14     randomword = random.choice(wordlist)  
15     guesses = ''  
16     points = 100  
17     while points > 0:  
18         badguesses = 0  
19         for character in randomword:  
20             if character in guesses:  
21                 print(f'{character} ', end = ' ')  
22             else:  
23                 # Thanks to Sevan for assisting me in this  
24                 badguesses += 1  
25                 print('_', end = ' ')  
26  
27         if badguesses == 0:  
28             print(f'You saved everyone from the bomb you have been awarded ${points} for the job. I hope you enjoyed it.')  
29             print(f'Word: {randomword}')  
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  

```