|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requirement** | | **Name** | **Location** | | **Notes** |
| **File** | **Line** |
| List | | *word\_list* | main.py | *17* | .splitlines() method returns an array |
| Tuple | | *self.\_\_secret\_word* | game.py | *12* | Secret word set as tuple so it doesn’t accidentally change |
| Set | | *self.guessed\_letters* | game.py | *15* | Tracks every unique letter guessed |
| Dictionary | | *feedback* | game.py | *98* | Maintains two keys for ‘perfect’ and ‘partial’ matches |
| Iteration | | --- | game.py | *73* | For-loop used to compare words, one char at a time |
| Conditional | | --- | game.py | *22* | If-statement checks if game is still ongoing |
| Try/Else | | --- | main.py | *13* | Try to read in data file. If exception, then exit |
| Input/Output File | | *word\_list.txt* | main.py | *15* | Read in text file containing all possible 4-letter words |
| User Defined Function | | *print\_feedback* | main.py | *33* | Translate *feedback* dictionary into meaningful text |
| User Defined Class | | *Game* | game.py | *7* | Game class contains secret word and game methods |
|  | init | *\_\_init\_\_* | game.py | *11* | Initializes secret word randomly from *word\_list* parameter |
|  | Private Attribute | *self.\_\_secret\_word* | game.py | *12* |  |
|  | Public Attribute #1 | *self.guesses* | game.py | *13* |  |
|  | Public Attribute #2 | *self.won\_game* | game.py | *14* |  |
|  | Private Method | *\_\_print\_win* | game.py | *111* |  |
|  | Public Method #1 | *validate* | game.py | *35* |  |
|  | Public Method #2 | *evaluate* | game.py | *59* |  |
|  | str() or repr() | *\_\_str\_\_* | game.py | *20* | Used by main.py on line 114 |
|  | Magic Class Method | *\_\_bool\_\_* | game.py | *18* | Used by main.py on line 113 |
| Unit Test #1 | | *test\_validate* | test\_game.py | *14* |  |
| Unit Test #2 | | *test\_evaluate* | test\_game.py | *36* |  |