Python projekt

re module = regulare ausdruck

random um random zahlen zu erstellen

10-14 interbuild von python alle dir man benötigt

nicht extra installieren

ulrlib2 für lok/text datei

os mod für clean screen

terminal bleibt immer sauber

time modul für pause und play

erste funktion: initial game

1. funktion intialGame
   1. haupt eingang , es sollen keine klassen oder instanzen verwendet werden, alles basiert auf funkt.
   2. (klassen sind besser für große projekte, für kleine reichen funktionen)
   3. zuerst : ruft diese eine funktion make word list auf welche eine liste von wörtern wiedergibt, die man erraten muss
2. make word list erzeugt die liste an wörtern die das ganze programm benutzen soll
   1. zeile 26-43
   2. akzeptiert 3 argumente:
      * 1. ein URL, die zeigt auf eine DATEIN auf, maybe online datein, mayby locale
           1. default datei “alice in wonderland” aus Gutenberg.org =defaultinput
        2. welche sprache die datei hat = englisch als default
        3. min länge
3. zeile 29 try except block
   1. module NLTK , nicht jeder pc hat die ,also erst try
   2. aus NLTK tool regexp importieren
   3. nltk.corpus.stopword s.words(language) von stopwörtern in englisch , für wörter wie „the“ und hello = hier nicht von interesse
   4. language argument ist nun default argument = englisch, falls deutscher text 🡪 change this to german = alles stopwords in german
   5. nach erzeugung ergebnis in stopwords speichern
   6. RegexpTokenizer() funktion, um eine saubere menge von Worter zu bekommen.
   7. nach import, lessen wir eine datei input.txt
   8. finally =filter mit 3 argumenten (min length) list operation benutzen um alle wörter mit der min länge zu sammeln dann speichern in variable “results” (um wörterdatenbank zu erzeugen)
   9. in unsere Haupteingan intialGame(), nach erfolgreich Words Datenbank zu bekommen, in zeile 20
   10. wir stellen eine Frage, mit eine Funktion heisst aksToStart() 🡪 true =spiel beginnt
4. def ask to start
   1. zeile 122 erst welcom print to screen
   2. dann frage stellen „press anykeye“
   3. rückmeldung vom user in variable start speichern
   4. dann test : user : exit dann print to screen XXXYYY
      1. exit() python build in funktion = schließt python
   5. wenn was anderes eingegeben =user will spielen: dann wird true returned
      1. zeile 20 wenn er spielen möchte def startGame(wordsList)wird aufgerufen
5. start game (funktion die wordlist als argument akzeptiert) z 45
6. jedes Mal funkt start game aufrufen, dann wählt programm ein wort als target word aus und ruft die funktion guess auf, da es unterschiedl wörter aufrufen soll
7. funkt pick target , um random zu erzeugen, word list als argument, random modul was wir am anfang importiert haben verwenden, zufalls zahl erzeugen, wenn es einmal erzeugt wurde wird es nicht mehr verwendet, nach erfolgriecher zahl , return das wort und dann in funktion start game
8. start game, ruf pick target aufdann zufallswort da also word target speichern dann zu guess
9. guess funk, 58 ist wichtig
   1. jedes Mal nur ein buchstaben, max versuche an zahlen danach fertig
   2. 59 target word kleinbuchstaben
   3. zeile 60 targetword als liste
   4. return word mit \* „hello“ --- return word ist „\*\*\*\*\*“
   5. 62 bingotime, nicht benutzen
   6. 63 retryTime 11
   7. 64 history, als leere liste zum speichern von falschen buchstaben
   8. show answer= toggle wenn true dann kann der user das richtige wort sehendefault ist false, aber testen, hier ist es true
   9. 66 while true = unendl loop, zuerst 67 (clear aufraufen) um screen sauber zu machen ---zeile 151 clear funk
   10. das sieht komische aus, os.system('cls' if os.name=='nt' else 'clear')—so funkt auf jedem pc , cls für linux/clear füe windows
   11. jedes mal sauber machen, und dann print info on screen
   12. 67 -🡪 68 funktion make tui
10. make tui : einfacher output für den screen zeile 92 ist die funkt
    1. 93 bis 97 argumente einfach string kombinieren
    2. string.format tools
    3. die funk bekommen argument, dann erzeugt eine output to screen
11. 69 nach erst TUI, dann stellt wir fragen fur user,um ein buchstaben zu verlanger
    1. ask letter aufrufen
12. ask letter zuerst eine tip "[888] to exit, [222] to showAnswer"
    1. frage stellen, und ergebnis bekommen, dann speichern in inputletter und dann test
    2. wenn user input 888 dann exit
    3. wenn 222 dann return 222
    4. andere dinge die user eingibt in 117 prüfen 🡪re modul regular ausdruck zum prüfen, nur buchstaben von a bis z
    5. wenn usrer was anderes eingibt, dann „not match“, solange not match wird die funkt askletter() aelbst angerufen bis der user ein buchstaben oder 888 oder 222 eingibt
    6. 69 ask letter
13. ask letter input letter kleinschreiben, speichern in variabler input letter
    1. wenn input letter 22 dann show answer ist true und dann make tui noch mal
    2. dann ask for more time 74-76= vergleichen 3 dinge: inputletter, targetletter, return word
    3. wir haben schon einge targetWord als list zb. [ a,b,c]

nd eine returnWord ist gleiche lange wie target word , hier ist [x, x, x] wenn das inputLetter ein buchstaben ist,

dann vergleichen wir welche position das inputLetter in targetWord auftritt

* 1. wenn user a eingibt dann durch targetword finde position 0 ist a, dann verändern das return word[0]zum a
  2. nach vergleich ist returnword [a,x,x], nach dem vergleich mit dem inputletter
  3. ganze targetwort vergleichenund return word in zeile 78
  4. if beide gleich dann prüfenwenn nicht gleich dann falscher inputund falscher buchstabe in history minus die retrytime
  5. retrytime=0 user hat schon 11 mal falsch eingegebendann ask for more time aufrufen, ask for more time nummer 2

1. ask for more time 83 prüfen ob return word gleich wie target word ist , wenn gleich dann noch mal ask for more time aufrufen mit argument 1
   1. in 132 if ask for more time bekommt 1 =user hat erfolgreich gespielt, wenn user 2 bekommt=user gibt auf und möchte anderes wort
   2. zeile 143 time modul um 2 sek pause im programm zu haben, nach 2 sek wird dann wieder start game aufgerufen und das mit der gl wordlist
2. ohne nltk schwierig saubere liste zu bekommen , sonst default input