DOCUMENTAZIONE

Descrizione della classe

Per la realizzazione del gioco, abbiamo creato una classe su Python, la quale tiene conto delle regole del calcolo combinatorio. Nella classe abbiamo creato varie funzioni che ci serviranno per realizzare il ruzzle. Ora elencheremo alcune di queste funzioni.

- anagrammi: questa funzione attraverso un ciclo di for realizzerà una lista di anagrammi a partire da una stringa di caratteri data. Il metodo permutazioni genera una lista di tuple, ogni tupla è una permutazione e se scorriamo la lista attraverso un ciclo, possiamo scorrere gli elementi della tupla per ricostruire la stringa.

```
def anagrammi(self):
    lettere = list(parola)

    permutazioni = list(permutations(lettere))

    temp = ''

    anagrammi = []

    for i in permutazioni:

        for carattere in i:
            temp += carattere

        anagrammi.append(temp)

        temp = ''
```

- charRipetuti: questa funzione serve per trovare i caratteri ripetuti e anche questa funzione sfrutta il ciclo di for, e se trova il carattere nel dictionary, incrementa il suo valore; se non lo trova lo aggiunge.

```
def charRipetuti(self):

    word = list(parola)
    caratteriripetuti={}
    nCaratteri = 0
    count = 0

    for i in word:
        if (i in caratteriripetuti):
            caratteriripetuti[str(i)] += 1
    else:
        caratteriripetuti[str(i)] = 1
    for i in caratteriripetuti:
        if caratteriripetuti[i]>1:
            count+=1
            nCaratteri += caratteriripetuti[i]
```

- combUtil: questa funzione controlla la lista delle parole e verifica se esse esistono nel vocabolario italiano

```
def combUtil(self):
    words = 'words.italian.txt'
    f = open(words, 'r')
    for riga in f:

       p=f.readline()

       if self.__stringa in p:
           return("è una parola italiana")
       if parola == p[:-1]:
           return("vero")
```

- fattoriale: questa funzione serve semplicemente per realizzare il fattoriale di un numero: In matematica, il fattoriale di un numero intero positivo n, è il prodotto dei numeri interi da 1 a n

```
def fattoriale(n):
    if n==0:
        return 1
    else:
        return n*fattoriale(n-1)
```

- coeffBinom: questa funzione, invece, serve per realizzare il coefficiente binomiale: Il coefficiente binomiale è un numero naturale definito a partire da una coppia di numeri naturali, solitamente indicati con n e k. In particolare il coefficiente binomiale n su k rappresenta il numero di sottoinsiemi di k elementi che si possono estrarre da un insieme di n elementi.

```
def coeffBinom(n, k):
    x = len(parola)
    y = int(input("Enter a value for y: "))
    if y == 1 or y == x:
       return(1)
    if y > x:
        return(0)
    else:
        a = fattoriale(x)
        b = fattoriale(y)
        div = a // (b*(x-y))
    return(div)
```

Dopo aver creato queste funzioni della classe calcComb (calcolo combinatorio), avremmo dovuto creare altre funzioni sulle permutazioni, disposizioni e ripetizioni, tutte con e senza ripetizioni.

Permutazioni

Una permutazione è il risultato di uno scambio dell'ordine degli elementi di una sequenza, ossia è uno dei possibili modi per ordinare elementi di qualsiasi tipo. Si distinguono due tipi di permutazioni: le permutazioni semplici, le permutazioni con ripetizione.

- Permutazioni semplici: le permutazioni semplici di n elementi distinti sono tutti i gruppi formati dagli n elementi, che differiscono per il loro ordine $P_n = n!$
- Permutazioni con ripetizioni: Le permutazioni con ripetizioni di n elementi di cui h, k,... ripetuti, sono tutti i gruppi formati dagli n elementi, che differiscono per l'ordine in

cui si presentano gli elementi distinti e la posizione che occupano gli elementi ripetuti. $P_n^{(h,k...)} = n! / h! * k! *…$

Disposizioni

Come per le permutazioni anche le disposizioni sono con e senza ripetizioni:

- DIsposizioni semplici: Le disposizioni semplici di n elementi distinti di classe k (con n,k appartenenti a N e k≤n) sono tutti i gruppi che si possono formare con k elementi, presi fra gli n, tali che ognuno è diverso dagli altri per gli elementi contenuti o l'ordine. $D_{n,k}= n* (n-1) * (n-2)*...*(n-k+1)=n!/(n-k)!$
- Disposizioni con ripetizioni: Le disposizioni con ripetizioni di n elementi distinti di classe k(con n,k appartenenti a N) sono tutti i gruppi che si possono formare con k elementi, anche ripetuti, presi fra gli n, tali che ogni gruppo è diverso dagli altri per gli elementi contenuti o per il loro ordine. $D'_{n,k} = n^k$

Combinazioni

Una combinazione è un raggruppamento di k elementi, presi in qualsiasi ordine, formato a partire da n elementi distinti. In termini più rigorosi si dice combinazione ogni sequenza di k elementi estratti tra n elementi distinti, nell'ipotesi che l'ordine di estrazione sia ininfluente. Pure in questi caso esistono combinazioni semplici e con ripetizioni:

- Combinazioni semplici: Le combinazioni semplici di n elementi distinti di classe k (con $0<k\leq n$) sono tutti i gruppi che si possono formare con k elementi, presi fra gli n, e tali che ogni gruppo è diverso dagli altri per almeno un elemento contenuto. $C_{n,k}=D_{n,k}/ P_k=n!/k!(n-k)!$
- Combinazioni con ripetizioni: Le combinazioni con ripetizioni di n elementi distinti di classe k, sono tutti i gruppi che si possono formare con k elementi, presi fra gli n; ogni elemento di un gruppo può essere ripetuto fino a k volte, non interessa l'ordine in cui gli elementi si presentano e in ciascun gruppo è diverso il numero delle volte in cui un elemento compare. $C'_{n,k}=C_{n+k-1,k}=(n+k-1)*(n+k-2)*.../k!$

Descrizione gioco

Il gioco creato prende molto spunto da ruzzle. Lo scopo del gioco è quello di trovare più parole possibili, con le lettere generate in una griglia 3x3, in un intervallo di tempo prestabilito di 1 minuto. Il punteggio aumenta in base a determinate regole:

- più una parola è lunga più punti si guadagnano, le parole hanno punteggi differenti:
  - dalle 2 alle 4 lettere = 2 punti
  - dalle 5 alle 8 = 5 punti
  - 9 lettere = 10 punti
- Esistono delle parole bonus che valgono il doppio di quanto dovrebbero valere. Le parole bonus sono scelte in base ad una categoria che è specificata prima di far partire il tempo.
- Una streak (serie) di parole giuste ti concede un bonus di 5 punti alla fine del livello.

Il gioco è diviso in 2 modalità: giocatore singolo , contro un altro giocatore.

- Giocatore singolo: per arrivare alla vittoria bisogna superare il punteggio necessario per avanzare al livello successivo. Ci sono delle differenze delle varie difficoltà:
  - principiante: non dovrà trovare le parole bonus

    - intermedio: dovrà trovare un minimo di 3 parole bonus
    - difficile: dovrà trovare un minimo di 6 parole bonus
- Contro un altro giocatore:per arrivare alla vittoria bisogna trovare quante più parole possibili prima che si esaurisca il tempo per superare il punteggio dell’avversario.

Interfaccia