

---

# RUZZLE

---

MANZILLI DAVIDE

IV E a.s. 2021/2022

Prof. Mazzone Diomede

Prof.ssa Tripepi Marina

---

<b>Cos'è Ruzzle</b>	<b>3</b>
<b>Idea di base</b>	<b>4</b>
<b>Le regole del gioco</b>	<b>5</b>
<b>Interfaccia</b>	<b>6</b>
<b>Diagramma di flusso</b>	<b>9</b>
<b>Realizzazione</b>	<b>13</b>

## Cos'è Ruzzle

Ruzzle è un gioco con alla base l'idea di stimolare e allenare la mente, attraverso la ricerca di parole da alcune lettere che vengono generate casualmente. Il suo fondamento si basa quindi sugli anagrammi, i quali sono basati sul concetto di permutazione, un concetto approfondito in ambito matematico attraverso il calcolo combinatorio. La realizzazione di questo gioco è stata possibile grazie ad alcune librerie in ambiente Python.



## Idea di base

Il calcolo combinatorio studia tutti i raggruppamenti ottenibili con un numero  $n$  di oggetti su  $k$  posti con o senza ripetizione degli elementi. In particolare ci sono tre raggruppamenti di cui si occupa il calcolo combinatorio:

- Combinazioni
- Disposizioni
- Permutazioni

In questo documento verranno brevemente descritte le permutazioni in quanto unico ramo del calcolo combinatorio sfruttato nella progettazione del gioco.

Le permutazioni sono la teoria alla base degli anagrammi, infatti consistono in raggruppamenti realizzati quando il numero di elementi  $n$  è uguale al numero di posti  $k$  e conta l'ordine in cui essi si dispongono.

Prendendo in considerazione una parola, “casa”, considerando ogni lettera come un elemento, quindi 4 elementi e  $k$  il numero totale degli elementi, sempre 4, abbiamo che il numero totale di anagrammi, e, quindi, di permutazioni è dato dalla formula  $n!$  (  $n$  fattoriale), cioè  $4 \times 3 \times 2 \times 1 = 24$ , la parola “casa” ha 24 anagrammi possibili!



**CALCOLO COMBINATORIO**  
COSA SONO GLI ANAGRAMMI?

---

## Le regole del gioco

Il gioco del ruzzle è un gioco semplice, come ogni gioco ha però bisogno di regole!

- All'avvio della partita compariranno sullo schermo alcune lettere casuali dell'alfabeto
- Il giocatore avrà a disposizione un minuto dall'avvio della partita
- lo scopo del giocatore sarà quello di trovare più parole possibili con le lettere a disposizione, ogni parola trovata aumenterà il punteggio del giocatore di 1 unità
- Al termine della partita il giocatore visualizzerà il suo punteggio paragonato al punteggio ottenibile



**REGOLE**

## Interfaccia

All'avvio del programma comparirà quindi una schermata così composta:



Questa è la schermata che permette all'utente che avvia il gioco di inserire il suo nome utente per salvare i dati relativi alle sue partite.



La schermata di gioco è, quindi, così composta:

- In alto a destra i classici pulsanti per la chiusura del programma, la modalità finestra e la modalità background.
- Al centro, sotto la scritta ruzzle compariranno le caselle contenenti le lettere dell'alfabeto da anagrammare, il numero delle caselle varia da 3 a 9
- Al centro della schermata vi è la casella di testo nella quale è possibile scrivere le parole.

- In basso vi è un elenco di tutte le parole trovate e ritenute valide dal sistema.

Al termine del gioco verrà visualizzata la schermata con il punteggio finale

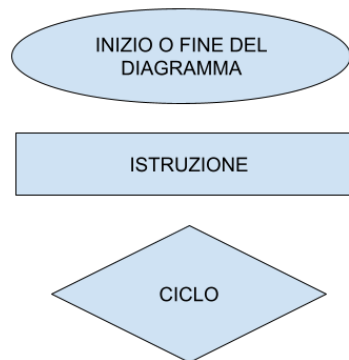


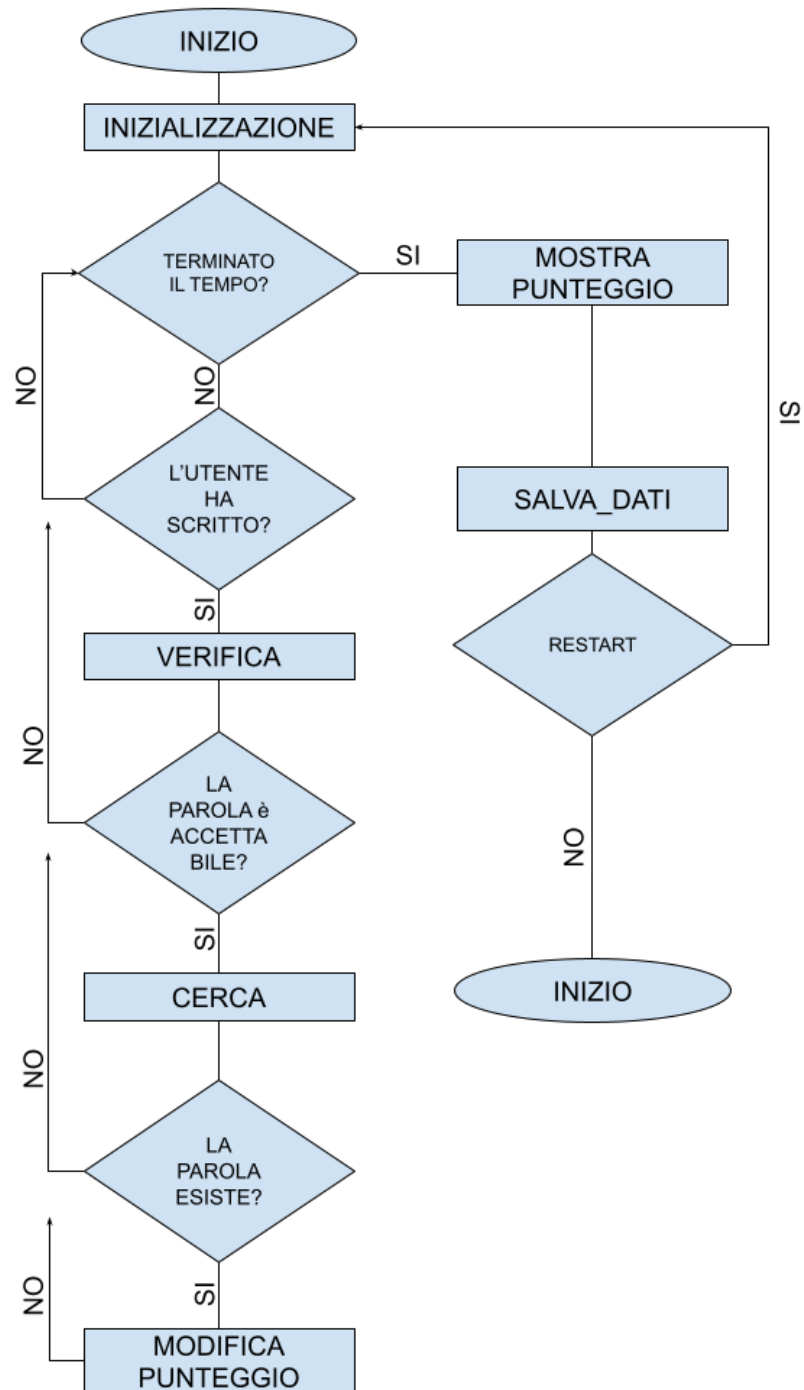


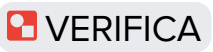
## Diagramma di flusso




Nella pagina successiva sarà possibile visualizzare il diagramma di flusso alla base dell'intero algoritmo del gioco. Come ogni progetto, anche la realizzazione di un gioco apparentemente semplice ha bisogno di una progettazione prima di iniziare a scrivere il codice. In particolare il diagramma è semplificato per poter essere facilmente comprensibile.

Al fine della comprensione si riportano le regole per la realizzazione di un diagramma di flusso.





- **INIZIALIZZAZIONE:** prevede l'avvio della schermata del gioco così come impostata nel paragrafo precedente, si avvia il tempo, compaiono le lettere nelle caselle.
- **CICLO TEMPO:** dá la condizione per la quale il gioco è finito o può continuare.
- **L'UTENTE HA SCRITTO:** in questo momento il programma è in esecuzione in attesa dell'input dell'utente che farà proseguire il programma una volta scritta la parola nell'apposita casella di testo.
- **VERIFICA:** una volta inserita la parola il sistema verifica che nella parola siano presenti solo le lettere generate nelle caselle.  

- **LA PAROLA è ACCETTABILE?:** una volta verificata la validità della parola dal programma, il gioco prosegue. Altrimenti verrà richiesto di inserire un'altra parola.

- **CERCA:** la funzione cerca sfoglia il file contenente tutto il dizionario italiano per verificare che la parola inserita dall'utente sia valida al fine di assegnare il punteggio.  CERCA
- **LA PAROLA ESISTE?:** Se la parola inserita risulta presente nel dizionario il gioco prosegue altrimenti verrà richiesto di inserire un'altra parola.
- **MODIFICA\_PUNTEGGIO:** Verificata la parola viene assegnato il punteggio che incrementa di 1 punto.  MODIFICA\_PUNTEGGIO
- **MOSTRA\_PUNTEGGIO:** Terminato il tempo viene mostrato a schermo il punteggio ottenuto dall'utente.
- **SALVA\_DATI:** Questa funzione richiama il nome utente inserito all'avvio del programma, infatti salverà sul database le informazioni relative alla partita, cioè nome utente inserito, lettere disponibili da anagrammare e le parole inserite e inseribili.  SALVA\_DATI
- **RESTART:** Viene data all'utente la possibilità di giocare un'altra partita o chiudere il programma.

## Realizzazione

La realizzazione del progetto prevede alla base la realizzazione di una classe in Python con tutti i metodi del calcolo combinatorio. In particolare una classe in python consiste in un insieme di elementi con caratteristiche comuni a cui applicare dei metodi e, quindi, delle funzioni.

I metodi che permettono la realizzazione del gioco sono:

Il costruttore:

```
class calcComb():  
    def __init__(self, stringa):  
        self.__N = len(stringa)  
        self.__stringa = stringa  
        self.__listStringa = list(stringa)
```

Questo metodo alla base di tutte le classi in python permette di costruire gli oggetti della classe, in questo caso ci permette di costruire la nostra stringa al quale applicare i metodi.

Cerca:

```
def cerca(self): #verificare se la STRINGA è presente nel file word.italian.txt
    f = open("class\words.italian.txt", 'r')
    #print(self.__stringa)
    for riga in f:
        righe = f.readline()
        #print(righe)
        #print(self.__stringa)
        if self.__stringa == righe[:-1]:
            return True
        else:
            False
```

Questo metodo consente di capire se la stringa è una parola della lingua italiana cercandola in un file contenente tutte le parole italiane. Nel caso la parola esistesse il risultato della ricerca sarà True, altrimenti False.

nPermutSenzaRip:

```
def nPermutSenzaRip(self):
    return math.factorial(self.__N)
```

Questo metodo consente di calcolare attraverso la formula vista in precedenza quante permutazioni e, quindi, anagrammi sono possibili dalla stringa.

## permutSenzaRip:


```
def permutSenzaRip(self):
    import itertools
    listapermutazioni = list(itertools.permutations(self.__stringa))
    temp = ''
    anagrammi = []
    for i in listapermutazioni:
        for carattere in i:
            # in temp concatenano tutti gli elementi della tupla così da
            # ottenere i singoli anagrammi della stringa iniziale
            temp += carattere

        # aggiungo la parola ricostruita dalla tupla alla lista finale degli anagrammi
        anagrammi.append(temp)
        # "svuoto" la variabile temp così da ricostruire un secondo anagramma
        temp = ''
    return anagrammi
```

Questo è il metodo principale del gioco, permette di visualizzare tutti gli anagrammi della stringa presa in considerazione, anche quelli che non hanno senso compiuto!

## Confutil:

```
def ConfUtil(self):
    for x in range(len(calcComb.permutSenzaRip(self))):
        calcComb.permutSenzaRip(self)[x] = self.__stringa
        calcComb.cerca(self)
    return calcComb.cerca(self)
```



Una volta stabiliti tutti gli anagrammi della stringa occorre, al fine di implementare nel gioco il punteggio, prendere in considerazioni solo quelli che hanno senso compiuto. Il metodo `confutil` confronta tutti gli anagrammi con tutte le parole della lingua italiana.