IMPICCATO

A cura di Anna Caprari, Abdullah Ijaz e Hadri Cobaj

SUDDIVISIONE DEI RUOLI

1. Fare scegliere la difficoltà del gioco: Hadri Cobaj
2. Generare una parola casuale segreta: Abdullah Ijaz
3. Nascondere la parola: Anna Caprari
4. Aggiornare la parola nascosta: Abdullah Ijaz
5. Far fare i tentativi all'utente: Anna Caprari con l'aiuto di Abdullah Ijaz
6. Fine della partita: Hadri Cobaj

1

```python
def difficoltà():
    difficile, intermedio, facile = (
        "Difficile: massimo 5 tentativi",
        "Intermedio: massimo 8 tentativi",
        "Facile: massimo 10 tentativi")
    print("Scegli la difficoltà: ")
    print(difficile, intermedio, facile, sep='\n')
    scelta = input("Scegli la difficoltà: ")
    print("-----------------------------------")
    while scelta not in ["facile", "intermedio", "difficile"]:
        print("Scelta non valida!!")
        print(difficile, intermedio, facile, sep='\n')
        scelta = input("Scegli la difficoltà: ")
    return scelta
```

2

```python
def quante_parole() -> str:
    scelta_file = int(input("Con quante parole vuoi giocare? 1,000 o 660,000: "))
    while scelta_file not in [1000, 660000]:
        print("Scelta non valida!!")
        scelta_file = int(input("Con quante parole vuoi giocare? 1,000 o 660,000: "))
    if scelta_file == 1000:
        path = r"1000_parole.txt"
    else:
        path = r"660000_parole.txt"
    return path
```

```python
def lunghezza_parole(lunghezza_massima):
    testo = f"Scegli la lunghezza della parola? Minimo 5 massimo {lunghezza_massima} :"
    lunghezza_parola = int(input(testo))
    while lunghezza_parola < 5 or lunghezza_parola > lunghezza_massima:
        print("Scelta non valida!!")
        lunghezza_parola = int(input(testo))
    return lunghezza_parola
```

2

```
def leggi_file(percorso_file: str) -> np.array(str):
    with open(percorso_file, encoding="UTF-8") as file:
        testo = file.readlines()
        return np.array([parola.strip() for parola in testo])
```

```
def scegli_parola(testo: np.array(str), lunghezza_scelta: int):
    func_mask = np.vectorize(lambda x: len(x) == lunghezza_scelta)
    parole_corrette = testo[func_mask(testo)]
    return np.random.choice(parole_corrette, size=1)
```

3

```python
def nascondi(parola: str) -> str:
    vocali = "aeiouàèìòù"
    st = ""
    for carattere in parola:
        if carattere in vocali:
            st = st + "+"
        else:
            st = st + "-"
    return st

```

4

```
1  def aggiorna(inserimento: np.array(str), enigma: str) → str:
2      sr = ''
3      for lettera in enigma:
4          if lettera in inserimento:
5              sr += lettera
6          else:
7              sr += nascondi(lettera)
8      return sr
```

5

```python
def fai_tentativi(scelta: str, parola: str):
    if scelta == 'facile':
        n_tentativi = 10
        inserimenti_utente = np.array((parola[0], parola[-1]))
    elif scelta == 'intermedio':
        n_tentativi = 8
        inserimenti_utente = np.array((parola[0]))
    else:
        n_tentativi = 5
        inserimenti_utente = np.array([])
```

5

```
while n_tentativi > 0:
        print(f"{n_tentativi} tentativi rimasti!")
        if scelta == "facile":
            print("Lettere o parole già inserite: ", end="")
            for i in inserimenti_utente:
                if i != inserimenti_utente[-1]:
                    print(i, end=" | ")
                else:
                    print(i)
        parola_nascosta = aggiorna(inserimenti_utente, parola)
        print(parola_nascosta)
        utente = input("Fai un tentativo: ")
        if utente in inserimenti_utente and scelta in ["facile", "intermedio"]:
            print("-----------------------------------")
            print("Lettera già inserita!")
        elif utente not in parola:
            n_tentativi -= 1
            print("-----------------------------------")
            print("No! Ritenta")
        else:
            print("-----------------------------------")
            print(f"Hai indovinato la lettera {utente}")
        if utente not in inserimenti_utente:
            inserimenti_utente = np.append(inserimenti_utente, utente)
        parola_aggiornata = aggiorna(inserimenti_utente, parola)
        if utente == parola or parola_aggiornata == parola:
            print("-----------------------------------")
            fine(True, parola)
            return
        print("-----------------------------------")
    else:
        fine(False, parola)
```

6

```python
def fine(vittoria: bool, enigma: str):
    if vittoria:
        print("Hai vinto!!")
        print(f"La parola era: {enigma}")
    else:
        print("IMPICCATO!!  :(")
        print(f"La parola era: {enigma}")
```

Programma assemblato

```
1  def main():
2      func_mask = np.vectorize(lambda x: len(x))
3      print("BENVENUTO NEL GIOCO DELL'IMPICCATO!")
4      print("-----------------------------------")
5      percorso_file = parole()
6      file = leggi_file(percorso_file)
7      max_len = np.max(func_mask(file))
8      scelta = difficoltà()
9      len_parola = lunghezza_parole(max_len)
10     parola = scegli_parola(file, len_parola)
11     fai_tentativi(scelta, parola[0])
12
13
14 if __name__ == "__main__":
15     main()
```