

Progetto Python

Studente: **Alessandro Busà**



Indice

- Introduzione
- Lista con tutte le ricette.
- Librerie Utilizzate
- Funzione “Aggiungi_Ricetta” + Output
- Funzione “Elimina_Ricetta” + Output
- Funzione “Visualizza_Ricette” + Output
- Funzione “Cerca_Ricetta”
- Funzione “Ricerca_Ricetta” + Output
- Funzione “Ingrediente_Frequenza” + Output
- Funzione “Ricetta_con_più_ingredienti” + Output
- Funzione “Ricetta_con_più_minutaggio” + Output
- Funzione “Statistiche_Ingredienti” + Output
- Funzione “Statistiche_Durata” + Output
- Funzione “Filtraggio_Avanzato” + Output
- Funzione “Filtraggio_Avanzato2” + Output
- GitHub

Introduzione



Il progetto di gestione delle ricette si propone di realizzare una piattaforma intuitiva e versatile, sviluppata in **Python**, che consente agli utenti di organizzare e analizzare una vasta collezione di ricette. Pensato per appassionati di cucina o per chi desidera mantenere un archivio ben strutturato delle proprie preparazioni, il sistema permette di gestire ricette con facilità, rendendo più agevoli le operazioni di aggiunta, modifica, eliminazione e consultazione delle ricette.

Le principali funzionalità includono la possibilità di inserire ricette dettagliate, specificando ingredienti e tempi di preparazione, così come di cercarle utilizzando criteri avanzati come nome, ingredienti e durata. Inoltre, il programma offre strumenti di analisi che permettono di ottenere statistiche utili, come l'ingrediente più comune o la ricetta più lunga da preparare.

Python è stato scelto per lo sviluppo del sistema grazie alla sua flessibilità e alla vasta disponibilità di librerie utili per la gestione e manipolazione dei dati. Questo linguaggio offre un ambiente ottimale per implementare le funzionalità di analisi e ricerca, assicurando prestazioni efficienti e codice leggibile.

Il sistema non si limita a fornire un archivio di ricette, ma diventa anche uno strumento di supporto per l'organizzazione delle proprie abitudini culinarie, offrendo un filtro avanzato per trovare rapidamente ricette con ingredienti specifici e in base al tempo disponibile. Le funzionalità di analisi permettono di valutare la frequenza degli ingredienti e di identificare rapidamente le ricette più complesse o veloci, rendendo l'intero processo di gestione delle ricette più efficiente e piacevole. Grazie alla sua interfaccia intuitiva e alle potenti funzionalità di analisi e filtraggio, questo progetto si configura come uno strumento essenziale per chiunque desideri ottimizzare e personalizzare la gestione delle proprie ricette culinarie.

Lista con tutte le ricette.

```
lista_ricette= [

    # Ogni ricetta è un dizionario con il nome della ricetta, una lista di ingredienti e il minutaggio richiesto.
    {'nome': 'Carbonara', 'ingredienti': ['Pasta', 'Uova', 'Pecorino', 'Parmigiano', 'Pepe Nero', 'Guanciale'], 'minutaggio': 30} ,
    {'nome': 'Matriciana', 'ingredienti': ['Pasta', 'Sugo di Pomodoro', 'Pecorino', 'Pepe Nero', 'Guanciale'], 'minutaggio': 45},
    {'nome': 'Pesto', 'ingredienti': ['Aglio', 'Basilico', 'Pinoli', 'Olio', 'Sale'], 'minutaggio': 10},
    {'nome': 'Polpetta', 'ingredienti': ['Carne Macinata', 'Uova', 'Pane', 'Pepe Nero', 'Sale', 'Parmigiano'], 'minutaggio': 20},
    {'nome': 'Margherita', 'ingredienti': ['Sugo di Pomodoro', 'Mozzarella', 'Basilico', 'Olio'], 'minutaggio': 15},
    {'nome': 'Lasagna', 'ingredienti': ['Pasta', 'Carne Macinata', 'Sugo di Pomodoro', 'Besciamella', 'Mozzarella', 'Parmigiano'], 'minutaggio': 90},
    {'nome': 'Risotto ai Funghi', 'ingredienti': ['Riso', 'Funghi', 'Brodo Vegetale', 'Vino Bianco', 'Cipolla', 'Parmigiano'], 'minutaggio': 40},
    {'nome': 'Tiramisu', 'ingredienti': ['Mascarpone', 'Uova', 'Caffè', 'Savoiardi', 'Zucchero', 'Cacao in Polvere'], 'minutaggio': 30},
    {'nome': 'Cacciatora', 'ingredienti': ['Pollo', 'Pomodoro', 'Cipolla', 'Olive', 'Vino Rosso', 'Rosmarino'], 'minutaggio': 60},
    {'nome': 'Frittata di Patate', 'ingredienti': ['Uova', 'Patate', 'Cipolla', 'Parmigiano', 'Sale', 'Pepe'], 'minutaggio': 30},
    {'nome': 'Caprese', 'ingredienti': ['Mozzarella', 'Pomodoro', 'Basilico', 'Olio d\'Oliva', 'Sale'], 'minutaggio': 10},
    {'nome': 'Zuppa di Legumi', 'ingredienti': ['Legumi Misti', 'Brodo Vegetale', 'Carota', 'Cipolla', 'Sedano', 'Pomodoro'], 'minutaggio': 50},
    {'nome': 'Pollo al Limone', 'ingredienti': ['Pollo', 'Limone', 'Olio d\'Oliva', 'Aglio', 'Rosmarino', 'Sale', 'Pepe'], 'minutaggio': 40},
    {'nome': 'Pancakes', 'ingredienti': ['Farina', 'Latte', 'Uova', 'Zucchero', 'Lievito in Polvere', 'Burro'], 'minutaggio': 20},
    {'nome': 'Couscous alle Verdure', 'ingredienti': ['Couscous', 'Zucchine', 'Peperoni', 'Pomodori', 'Cipolla', 'Olio d\'Oliva'], 'minutaggio': 30},
    {'nome': 'Spaghetti Aglio e Olio', 'ingredienti': ['Spaghetti', 'Aglio', 'Peperoncino', 'Olio d\'Oliva', 'Prezzemolo'], 'minutaggio': 20},
    {'nome': 'Sgombro al Forno', 'ingredienti': ['Sgombro', 'Limone', 'Rosmarino', 'Olio d\'Oliva', 'Sale', 'Pepe'], 'minutaggio': 25},
    {'nome': 'Involtini di Melanzane', 'ingredienti': ['Melanzane', 'Ricotta', 'Pomodoro', 'Mozzarella', 'Basilico'], 'minutaggio': 45},
    {'nome': 'Torta di Mele', 'ingredienti': ['Mele', 'Farina', 'Zucchero', 'Uova', 'Burro', 'Lievito in Polvere'], 'minutaggio': 60},
    {'nome': 'Gnocchi al Pesto', 'ingredienti': ['Gnocchi di Patate', 'Pesto', 'Parmigiano'], 'minutaggio': 20},
    {'nome': 'Boeuf Bourguignon', 'ingredienti': ['Manzo', 'Vino Rosso', 'Carota', 'Cipolla', 'Funghi', 'Bacon', 'Brodo di Carne'], 'minutaggio': 120},
    {'nome': 'Falafel', 'ingredienti': ['Ceci', 'Aglio', 'Cipolla', 'Prezzemolo', 'Coriandolo', 'Cumino', 'Farina'], 'minutaggio': 45},
    {'nome': 'Moussaka', 'ingredienti': ['Melanzane', 'Carne Macinata', 'Pomodoro', 'Cipolla', 'Besciamella', 'Parmigiano'], 'minutaggio': 90},
    {'nome': 'Chili con Carne', 'ingredienti': ['Carne Macinata', 'Fagioli', 'Pomodoro', 'Peperoni', 'Cipolla', 'Spezie'], 'minutaggio': 60},
    {'nome': 'Zuppa di Cipolle', 'ingredienti': ['Cipolla', 'Brodo di Carne', 'Pane', 'Formaggio Gruyère', 'Burro'], 'minutaggio': 50},
    {'nome': 'Insalata di Tonno', 'ingredienti': ['Tonno in scatola', 'Pomodori', 'Cetrioli', 'Olive', 'Cipolla', 'Olio d\'Oliva'], 'minutaggio': 15},
    {'nome': 'Tacos', 'ingredienti': ['Tortillas', 'Carne Macinata', 'Lattuga', 'Pomodoro', 'Formaggio', 'Salsa'], 'minutaggio': 30},
    {'nome': 'Pasta al Pesto di Rucola', 'ingredienti': ['Pasta', 'Rucola', 'Noci', 'Parmigiano', 'Olio d\'Oliva', 'Aglio'], 'minutaggio': 20},
    {'nome': 'Ratatouille', 'ingredienti': ['Melanzane', 'Zucchine', 'Peperoni', 'Pomodoro', 'Cipolla', 'Aglio', 'Olio d\'Oliva'], 'minutaggio': 60},
    {'nome': 'Polpette di Ricotta', 'ingredienti': ['Ricotta', 'Farina', 'Uova', 'Parmigiano', 'Prezzemolo', 'Sale'], 'minutaggio': 30},
    {'nome': 'Pancetta alla Griglia', 'ingredienti': ['Pancetta', 'Sale', 'Pepe', 'Rosmarino', 'Olio d\'Oliva'], 'minutaggio': 20},
    {'nome': 'Frittelle di Zucchine', 'ingredienti': ['Zucchine', 'Farina', 'Uova', 'Parmigiano', 'Aglio', 'Prezzemolo'], 'minutaggio': 25},
    {'nome': 'Crostini al Pomodoro', 'ingredienti': ['Pane', 'Pomodori', 'Aglio', 'Basilico', 'Olio d\'Oliva', 'Sale'], 'minutaggio': 15},
    {'nome': 'Quiche Lorraine', 'ingredienti': ['Pasta Brisè', 'Panna', 'Uova', 'Bacon', 'Formaggio Gruyère', 'Cipolla'], 'minutaggio': 50},
    {'nome': 'Sgombro alla Griglia', 'ingredienti': ['Sgombro', 'Limone', 'Rosmarino', 'Olio d\'Oliva', 'Sale', 'Pepe'], 'minutaggio': 25},
    {'nome': 'Torta Salata con Spinaci e Ricotta', 'ingredienti': ['Pasta Brisè', 'Spinaci', 'Ricotta', 'Parmigiano', 'Uova', 'Noce Moscata'], 'minutaggio': 45}

]
```

Librerie Utilizzate

```
from collections import Counter
```



Importa Counter dal modulo collections, che è utile per contare gli elementi in una sequenza.

```
import pandas as pd
```



Importa la libreria pandas, utile per la manipolazione dei dati.

Funzione "Aggiungi_Ricetta"

```
# Definisce una funzione che permette di aggiungere una nuova ricetta nella lista. (Start2impact -> Registrazione di un nuovo elemento)
def aggiungi_ricetta(lista):

    """
    Aggiunge una nuova ricetta alla lista delle ricette se non è già presente.

    Args:
        lista (list): Lista che contiene tutte le ricette.

    Returns:
        Lista aggiornata con la nuova ricetta se presente,
        altrimenti restituisce la lista originale.
    """

    nome = input("Inserisci il nome della ricetta: ") # Chiede all'utente di inserire il nome della ricetta.

    for ricetta in lista: # Controlla se la ricetta esiste già nella lista per evitare duplicati.
        if ricetta['nome'].lower() == nome.lower(): # Se la ricetta è già presente, notifica l'utente e restituisce la lista originale.
            print(f"La ricetta '{nome}' è già presente nella lista.")
            return lista # Restituisce la lista originale se la ricetta è duplicata

    ingredienti = input("Inserisci gli ingredienti separati da virgole: ").split(',') # Chiede all'utente di inserire gli ingredienti.
    while True: # Ciclo Infinito
        try:
            minutaggio = int(input("Inserisci il minutaggio necessario (in minuti): ")) # Chiede all'utente di inserire il minutaggio della ricetta.
            if minutaggio > 0:
                break # Esce dal ciclo solo se il minutaggio è valido (positivo).
            else:
                print("Il minutaggio deve essere un numero maggiore di 0. Riprova.") # Messaggio di errore se il minutaggio è <= 0.
        except ValueError:
            print("Inserisci un numero valido per il minutaggio. Riprova.")

    nuova_ricetta = { # Crea un nuovo dizionario con i dettagli della ricetta
        'nome': nome, # Assegna il nome alla ricetta.
        'ingredienti': [ingrediente.strip() for ingrediente in ingredienti], # Assegna gli ingredienti alla ricetta e rimuove eventuali spazi inutili dagli ingredienti.
        'minutaggio': minutaggio # Assegna il minutaggio alla ricetta.
    }

    lista.append(nuova_ricetta) # Aggiunge il nuovo dizionario alla lista delle ricette
    print(f"La ricetta '{nome}' è stata aggiunta con successo.") # Notifica l'utente che la ricetta è stata aggiunta con successo.

    return lista # Restituisce la lista aggiornata
```

Output "Aggiungi_Ricetta"

```
#Funzione Richiamata per Aggiungere una Nuova Ricetta a quelle già presenti.  
aggiungi_ricetta(lista_ricette)
```

Funzione Richiamata

Output Restituito

```
Inserisci il nome della ricetta: 
```

Inserimento del nome

```
Inserisci il nome della ricetta: Pollo  
Inserisci gli ingredienti separati da virgole: 
```

Inserimento degli ingredienti

```
Inserisci il nome della ricetta: Pollo  
Inserisci gli ingredienti separati da virgole: Pollo, Rosmarino, Pomodoro, Cipolla  
Inserisci il minutaggio necessario (in minuti): 
```

Inserimento del minutaggio

```
Inserisci il nome della ricetta: Pollo  
Inserisci gli ingredienti separati da virgole: Pollo, Rosmarino, Pomodoro, Cipolla  
Inserisci il minutaggio necessario (in minuti): 35  
La ricetta 'Pollo' è stata aggiunta con successo.
```

Output Finale

Funzione "Elimina_Ricetta"

Definisce una funzione che consente di eliminare una ricetta dalla lista.

```
# Definisce una funzione che consente di eliminare una ricetta dalla lista.
def elimina_ricetta(lista):

    """
    Elimina una ricetta dalla lista basata sul nome fornito dall'utente.

    Args:
        lista (list): Lista che contiene tutte le ricette.

    Returns:
        Lista aggiornata senza la ricetta eliminata se presente,
        altrimenti restituisce la lista originale.
    """

    nome = input("Inserisci il nome della ricetta da eliminare: ").lower()
    for ricetta in lista:
        if ricetta['nome'].lower() == nome:
            lista.remove(ricetta)
            print(f"Ricetta '{nome}' eliminata.")
            return lista

    print(f"Ricetta '{nome}' non trovata.")
    return lista

# Chiede all'utente di inserire il nome della ricetta da eliminare, convertendolo in minuscolo per uniformità.
# Cicla attraverso la lista delle ricette per trovare quella con il nome corrispondente.
# Confronta il nome della ricetta in minuscolo con l'input dell'utente.
# Rimuove la ricetta dalla lista
# Notifica l'utente che la ricetta è stata eliminata.
# Restituisce la lista aggiornata dopo l'eliminazione.

# Se la ricetta non viene trovata, notifica l'utente.
# Restituisce la lista originale se la ricetta non è trovata
```


Output "Elimina_Ricetta"

```
#Funzione Richiamata per Eliminare una Ricetta già presente nella lista.  
elimina_ricetta(lista_ricette)
```

Funzione Richiamata

Output Restituito

```
Inserisci il nome della ricetta da eliminare: Tacos  
Ricetta 'tacos' eliminata.
```

Se la ricetta dovesse essere presente nella lista, viene eliminata mostrando questo "output"

```
Inserisci il nome della ricetta da eliminare: Zucchina Ripiena  
Ricetta 'zucchina ripiena' non trovata.
```

Se la ricetta **NON** dovesse essere presente nella lista, verrà mostrato questo "output"

Funzione "Visualizza_Ricette"

Definisce una funzione che permetta di migliorare la visualizzazione delle ricette.

```
# Definisce una funzione che permetta di migliorare la visualizzazione delle ricette. (Start2impact -> Visualizzazione di tutti gli elementi)
def visualizza_ricette(lista):

    """
    Mostra tutte le ricette presenti nella lista, formattando nome, ingredienti e minutaggio.

    Args:
        lista (list): Lista che contiene tutte le ricette.

    Returns:
        None: Stampa le ricette
    """

    for ricetta in lista:
        print(f"Nome: {ricetta['nome']}")
        print(f"Ingredienti: {' '.join(ricetta['ingredienti'])}")
        print(f"Minutaggio: {ricetta['minutaggio']} minuti")
        print("-" * 40)

        # Cicla attraverso ogni ricetta nella lista
        # Stampa il nome della ricetta
        # Stampa gli ingredienti uniti in una stringa, separati da virgole
        # Stampa il minutaggio della ricetta
        # Stampa una linea di separazione per rendere l'output più leggibile
```


Output "Visualizza_Ricette"

```
#Funzione Richiamata per migliorare la visualizzazione di tutte le ricette.  
visualizza_ricette(lista_ricette)  
|
```

→ *Funzione Richiamata*

Output Restituito

```
Nome: Carbonara  
Ingredienti: Pasta, Uova, Pecorino, Parmigiano, Pepe Nero, Guanciale  
Minutaggio: 30 minuti  
-----  
Nome: Matriciana  
Ingredienti: Pasta, Sugo di Pomodoro, Pecorino, Pepe Nero, Guanciale  
Minutaggio: 45 minuti  
-----  
Nome: Pesto  
Ingredienti: Aglio, Basilico, Pinoli, Olio, Sale  
Minutaggio: 10 minuti  
-----  
Nome: Polpetta  
Ingredienti: Carne Macinata, Uova, Pane, Pepe Nero, Sale, Parmigiano  
Minutaggio: 20 minuti  
-----  
Nome: Margherita  
Ingredienti: Sugo di Pomodoro, Mozzarella, Basilico, Olio  
Minutaggio: 15 minuti  
-----  
Nome: Lasagna  
Ingredienti: Pasta, Carne Macinata, Sugo di Pomodoro, Besciamella, Mozzarella, Parmigiano  
Minutaggio: 90 minuti  
-----
```

→ *Visualizzazione di tutte le ricette
(qui ne vengono mostrare solo alcune)*

Funzione "Cerca_Ricetta"

Definisce una funzione che permetta di cercare ricette basate su uno o piu' attributi

```
# Definisce una funzione che permetta di cercare ricette basate su uno o piu' attributi
def cerca_ricette(lista, nome=None, ingrediente=None, minutaggio=None):

    """
    Cerca ricette nella lista in base a nome, ingrediente o minutaggio fornito.

    Args:
        lista (list): Lista che contiene tutte le ricette.
        nome (str, optional): Nome della ricetta da cercare.
        ingrediente (str, optional): Ingrediente da cercare nelle ricette.
        minutaggio (int, optional): Minutaggio per filtrare le ricette.

    Returns:
        None: Stampa le ricette che soddisfano i criteri di ricerca oppure un messaggio se non ci sono risultati.
    """

    risultati = [] # Crea una lista vuota per memorizzare le ricette che soddisfano i criteri di ricerca.
    for ricetta in lista: # Itera attraverso ogni ricetta nella lista.
        if nome and nome.lower() not in ricetta['nome'].lower(): # Controlla se è specificato un nome e se il nome della ricetta non corrisponde, salta la ricetta.
            continue
        if ingrediente and all(ingrediente.lower() not in ingr.lower() for ingr in ricetta['ingredienti']): # Controlla se è specificato un ingrediente e se non è presente negli ingredienti della ricetta, salta la ricetta.
            continue
        if minutaggio and ricetta['minutaggio'] != minutaggio: # Controlla se è specificato un minutaggio e se non corrisponde a quello della ricetta, salta la ricetta.
            continue
        risultati.append(ricetta) # Se tutte le condizioni sono soddisfatte, aggiunge la ricetta alla lista dei risultati.

    if risultati: # Se ci sono risultati, stampali.
        for ricetta in risultati:
            visualizza_ricette(ricetta) # Visualizza le ricette che corrispondono ai criteri di ricerca.
    else: # Se non ci sono risultati, stampa un messaggio di avviso.
        print("Nessuna ricetta trovata che soddisfi i criteri.")
```


Funzione "Ricerca_Ricetta"

Definisce una funzione che permetta di interagire con l'utente per acquisire criteri di ricerca.

```
# Definisce una funzione che permetta di interagire con l'utente per acquisire criteri di ricerca. (Start2impact -> Ricerca di elementi)
def ricerca_ricetta(lista):

    """
    Consente all'utente di cercare una ricetta per nome, ingrediente o minutaggio.
    Args:
        lista (list): Lista che contiene tutte le ricette.
    Returns:
        None: Stampa i risultati della ricerca o un messaggio di errore se l'input non è valido.
    """

    print("Criteri di ricerca disponibili:")                                # Stampa le opzioni di ricerca disponibili per l'utente.
    print("1. Nome")
    print("2. Ingrediente")
    print("3. Minutaggio")
    scelta = input("Scegli il criterio di ricerca (1/2/3): ")

    if scelta == '1':
        nome = input("Inserisci il nome della ricetta da cercare: ")
        cerca_ricette(lista, nome=nome)
    elif scelta == '2':
        ingrediente = input("Inserisci l'ingrediente da cercare: ")
        cerca_ricette(lista, ingrediente=ingrediente)
    elif scelta == '3': # Ricerca per minutaggio
        while True:
            try:
                minutaggio = int(input("Inserisci il minutaggio della ricetta da cercare: "))
                if minutaggio < 0:
                    print("Il minutaggio deve essere un numero positivo. Riprova.")
                    continue
                break
            except ValueError:
                print("Per favore inserisci un numero valido. Riprova.")
        cerca_ricette(lista, minutaggio=minutaggio)
    else:
        print("Scelta non valida. Per favore, scegli 1, 2 o 3.")

# Chiede all'utente di scegliere un criterio di ricerca tra le opzioni disponibili.
# Se l'utente ha scelto di cercare per nome, richiede il nome della ricetta e chiama la funzione cerca_ricette.
# Input per il nome della ricetta
# Chiamata alla funzione di ricerca con il nome specificato.
# Se l'utente ha scelto di cercare per ingrediente, richiede l'ingrediente e chiama la funzione cerca_ricette.
# Input per l'ingrediente
# Chiamata alla funzione di ricerca con l'ingrediente specificato

# Inserisce un numero da tastiera

# Ripeti il ciclo se il numero è negativo
# Esci dal ciclo se il numero è valido
# Gestisce il caso in cui venga inserita una parola

# Chiamata alla funzione di ricerca con il minutaggio specificato.

# Se l'input non è valido (non è 1, 2 o 3), stampa un messaggio di errore.
```

Output "Ricerca_Ricetta"

```
#Funzione Richiamata per ricercare una determinata ricetta in base a determinati criteri.  
ricerca_ricetta(lista_ricette)
```

Funzione Richiamata

Output Restituito

```
Criteri di ricerca disponibili:  
1. Nome  
2. Ingrediente  
3. Minutaggio  
Scegli il criterio di ricerca (1/2/3): 
```

```
Scegli il criterio di ricerca (1/2/3): 1  
Inserisci il nome della ricetta da cercare: carbonara  
Nome: Carbonara  
Ingredienti: Pasta, Uova, Pecorino, Parmigiano, Pepe Nero, Guanciale  
Minutaggio: 30 minuti
```

```
Scegli il criterio di ricerca (1/2/3): 1  
Inserisci il nome della ricetta da cercare: Polipo Bollito  
Nessuna ricetta trovata che soddisfi i criteri.
```

```
Scegli il criterio di ricerca (1/2/3): 2  
Inserisci l'ingrediente da cercare: guanciale  
Nome: Carbonara  
Ingredienti: Pasta, Uova, Pecorino, Parmigiano, Pepe Nero, Guanciale  
Minutaggio: 30 minuti  
-----  
Nome: Matriciana  
Ingredienti: Pasta, Sugo di Pomodoro, Pecorino, Pepe Nero, Guanciale  
Minutaggio: 45 minuti  
-----
```

```
Scegli il criterio di ricerca (1/2/3): 3  
Inserisci il minutaggio della ricetta da cercare: 15  
Nome: Margherita  
Ingredienti: Sugo di Pomodoro, Mozzarella, Basilico, Olio  
Minutaggio: 15 minuti  
-----  
Nome: Insalata di Tonno  
Ingredienti: Tonno in scatola, Pomodori, Cetrioli, Olive, Cipolla, Olio d'Oliva  
Minutaggio: 15 minuti  
-----  
Nome: Crostini al Pomodoro  
Ingredienti: Pane, Pomodori, Aglio, Basilico, Olio d'Oliva, Sale  
Minutaggio: 15 minuti  
-----
```

```
Inserisci il minutaggio della ricetta da cercare: pollo  
Per favore inserisci un numero valido. Riprova.  
Inserisci il minutaggio della ricetta da cercare: -2  
Il minutaggio deve essere un numero positivo. Riprova.  
Inserisci il minutaggio della ricetta da cercare: 15
```

Questo controllo è disponibile
in tutti e 3 i criteri di ricerca

Funzione "Ingrediente_Frequenza"

Definisce una funzione che permetta di visualizzare con quale frequenza si presenta un determinato ingrediente

```
# Definisce una funzione che permetta di visualizzare con quale frequenza si presenta un determinato ingrediente (Start2impact -> Statistiche sugli elementi)
def ingrediente_frequenza(lista):

    """
    Chiede all'utente di inserire un ingrediente e determina quante volte appare tra tutte le ricette.

    Args:
    | lista (list): Lista che contiene tutte le ricette.

    Returns:
    | None: Stampa i risultati della ricerca o un messaggio se l'ingrediente non è stata trovato.
    """

    if not lista:
        print("La lista delle ricette è vuota.")
        return None

    tutti_ingredienti = []

    for ricetta in lista:
        tutti_ingredienti.extend([ingrediente.lower() for ingrediente in ricetta['ingredienti']])

    ingrediente_cercato = input("Inserisci l'ingrediente di cui vuoi conoscere la frequenza: ").strip().lower()
    conteggi = Counter(tutti_ingredienti)
    frequenza_ingrediente = conteggi.get(ingrediente_cercato, 0)

    if frequenza_ingrediente > 0:
        print(f"L'ingrediente '{ingrediente_cercato}' appare {frequenza_ingrediente} volte nelle ricette.")
    else:
        print(f"L'ingrediente '{ingrediente_cercato}' non è presente in nessuna ricetta.")
```

Controlla se la lista è vuota
Se sì, stampa un messaggio.
La funzione termina se non ci sono ricette.

Inizializza una lista vuota per raccogliere tutti gli ingredienti.

Itera attraverso ogni ricetta nella lista
Converte tutti gli ingredienti in minuscolo per la ricerca

Richiede all'utente di inserire l'ingrediente da cercare e lo converte in minuscolo
Conta la frequenza di ogni ingrediente (tutti in minuscolo).
Ottiene la frequenza dell'ingrediente cercato (0 se non trovato)

Stampa il risultato.

Se l'input non è valido, stampa un messaggio.

Output "Ingrediente_Frequenza"

```
#Funzione Richiamata per visualizzare la frequenza con la quale si presenta un determinato ingrediente  
risultato= ingrediente_frequenza(lista_ricette)
```

→ *Funzione Richiamata*

Output Restituito

Inserisci l'ingrediente di cui vuoi conoscere la frequenza: aglio
L'ingrediente 'aglio' appare 8 volte nelle ricette.

Quando l'ingrediente viene
trovato restituisce il numero di
volte che appare

Inserisci l'ingrediente di cui vuoi conoscere la frequenza: Sale Rosa
L'ingrediente 'sale rosa' non è presente in nessuna ricetta.

Quando l'ingrediente **NON**
viene trovato restituisce un
messaggio di output

Funzione " Ricetta_con_più_ingredienti"

Definisce una funzione che permetta di visualizzare la ricetta con più ingredienti

```
# Definisce una funzione che permetta di visualizzare la ricetta con più ingredienti (Start2impact -> Statistiche sugli elementi)
def ricetta_con_piu_ingredienti(lista):

    """
    Determina e restituisce la ricetta con il maggior numero di ingredienti.

    Args:
    | lista (list): Lista che contiene tutte le ricette.

    Returns:
    | dict or None: Restituisce un dizionario contenente la ricetta con il maggior numero di ingredienti,
    | | | | | o None se la lista è vuota.
    """

    if not lista:
        print("La lista delle ricette è vuota.")
        return None

    ricetta_max_ingredienti = None
    max_ingredienti = 0
    for ricetta in lista:
        numero_ingredienti = len(ricetta['ingredienti'])
        if numero_ingredienti > max_ingredienti:
            max_ingredienti = numero_ingredienti
            ricetta_max_ingredienti = ricetta

    return ricetta_max_ingredienti
```

Output "Ricetta_con_più_ingredienti"

```
#Funzione Richiamata per mostrare la ricetta che contiene più ingredienti.  
ricetta_max_ingredienti = ricetta_con_piu_ingredienti(lista_ricette)  
  
if ricetta_max_ingredienti:  
    print(f"La ricetta con il maggior numero di ingredienti è '{ricetta_max_ingredienti['nome']}' con {len(ricetta_max_ingredienti['ingredienti'])} ingredienti.")  
  
print("-" * 40)
```

→ *Funzione Richiamata*

Output Restituito

La ricetta con il maggior numero di ingredienti è 'Pollo al Limone' con 7 ingredienti.

Funzione " Ricetta_con_più_minutaggio"

Definisce una funzione che permetta di visualizzare la ricetta che richiede maggior minutaggio per la preparazione

```
# Definisce una funzione che permetta di visualizzare la ricetta che richiede maggior minutaggio per la preparazione (Start2impact -> Statistiche sugli elementi)
def ricetta_con_piu_minutaggio(lista):

    """
    Determina e visualizza la ricetta con il maggior minutaggio.

    Args:
        lista (list): Lista che contiene tutte le ricette, dove ogni ricetta è rappresentata come un dizionario
        con chiavi 'nome', 'ingredienti' e 'minutaggio'.

    Returns:
        dict or None: Restituisce un dizionario contenente la ricetta con il maggior minutaggio,
        o None se la lista è vuota.
    """

    if not lista:
        print("La lista delle ricette è vuota.")
        return None

    ricetta_max_minutaggio = None
    max_minutaggio = 0
    for ricetta in lista:
        minutaggio = ricetta['minutaggio']
        if minutaggio > max_minutaggio:
            max_minutaggio = minutaggio
            ricetta_max_minutaggio = ricetta

    return ricetta_max_minutaggio
```

```
# Controlla se la lista è vuota
# Se sì, stampa un messaggio.
# La funzione restituisce None se non ci sono ricette.

# Inizializza una variabile per tenere traccia della ricetta con il maggior minutaggio.
# Inizializza una variabile per tenere traccia del minutaggio massimo.
# Scorre ogni ricetta nella lista fornita.
# Estrae il minutaggio della ricetta corrente.
# Se il minutaggio della ricetta corrente è maggiore di quello attualmente massimo
# Aggiorna la variabile massimo
# Aggiorna la ricetta corrispondente.

# Restituisce la ricetta con il maggior minutaggio.
```


Output "Ricetta_con_più_minutaggio"

```
#Funzione Richiamata per mostrare la ricetta che richiede maggior Minutaggio.  
ricetta_max_minutaggio = ricetta_con_piu_minutaggio(lista_ricette)  
  
if ricetta_max_minutaggio:  
    print(f"La ricetta con il maggior minutaggio è '{ricetta_max_minutaggio['nome']}' con {ricetta_max_minutaggio['minutaggio']} minuti.")  
  
print("-" * 40)
```

→ *Funzione Richiamata*

Output Restituito

La ricetta con il maggior minutaggio è 'Boeuf Bourguignon' con 120 minuti.

Funzione "Statistiche_Ingredienti"

Definisce una funzione che permetta di visualizzare gli ingredienti più e meno usati

```
# Definisce una funzione che permetta di visualizzare gli ingredienti più e meno usati (Start2Impact -> Statistiche sugli elementi)
def statistiche_ingredienti(lista):

    """
    Analizza e stampa statistiche sugli ingredienti delle ricette.

    Args:
        lista (list): Lista che contiene tutte le ricette,
                       dove ogni ricetta è rappresentata come un dizionario
                       con chiavi 'nome', 'ingredienti' e 'minutaggio'.

    Returns:
        None: Stampa i risultati della ricerca o un messaggio di errore se la lista è vuota.
    """

    if not lista:
        print("La lista delle ricette è vuota.")
        return None

    tutti_ingredienti = []

    for ricetta in lista:
        tutti_ingredienti.extend(ricetta['ingredienti'])

    conteggi = Counter(tutti_ingredienti)
    ingredienti_comuni = conteggi.most_common(5)
    frequenza_minima = min(conteggi.values())
    ingredienti_meno_comuni = [ingrediente for ingrediente, frequenza in conteggi.items() if frequenza == frequenza_minima] # Crea una lista di tutti gli ingredienti che hanno la frequenza minima.

    print("Ingredienti più comuni:")
    for ingrediente, frequenza in ingredienti_comuni:
        print(f"{ingrediente}: {frequenza} occorrenze")

    print("-" * 40)

    print(f"Ingredienti meno comuni: {'', '.join(ingredienti_meno_comuni)} (frequenza: {frequenza_minima})") # Stampa gli ingredienti meno comuni e la loro frequenza minima.
```

Output "Statistiche_Ingredienti"

```
#Funzione Richiamata per mostrare le ricette con piu'/meno minutaggio.  
statistiche_ingredienti(lista_ricette)
```

Funzione Richiamata

Output Restituito

```
Ingredienti più comuni:  
Cipolla: 13 occorrenze  
Parmigiano: 11 occorrenze  
Olio d'Oliva: 11 occorrenze  
Uova: 10 occorrenze  
Sale: 10 occorrenze
```

Ingredienti **PIU'** comuni

```
Ingredienti meno comuni: Pinoli, Olio, Olio, Riso, Vino Bianco, Mascarpone, Caffè, Savoiardi, Cacao in Polvere, Patate, Legumi Misti, Sedano, Latte, Couscous, Pomodorini, Spaghetti, Peperoncino, Mele, Gnocchi di Patate, Pesto, Manzo, Ceci  
, Coriandolo, Cumino, Fagioli, Spezie, Tonno in scatola, Cetrioli, Tortillas, Lattuga, Formaggio, Salsa, Rucola, Noci, Pancetta, Panna, Spinaci, Noce Moscata (frequenza: 1)
```

Ingredienti **MENO** comuni

Funzione "Statistiche_Durata"

Definisce una funzione che permetta di visualizzare il minutaggio minimo, massimo e la media sul totale

```
# Definisce una funzione che permetta di visualizzare il minutaggio minimo, massimo e la media sul totale (Start2impact -> Statistiche sugli elementi)
def statistiche_durata(lista):

    """
    Analizza e stampa statistiche sulla durata delle ricette.

    Args:
        lista (list): Lista che contiene tutte le ricette,
                       dove ogni ricetta è rappresentata come un dizionario
                       con chiavi 'nome', 'ingredienti' e 'minutaggio'.

    Returns:
        None: Stampa i risultati della ricerca o un messaggio di errore se la lista è vuota.
    """

    if not lista:
        print("La lista delle ricette è vuota.")
        return None

    df = pd.DataFrame(lista)

    min_durata = df['minutaggio'].min()
    media_durata = df['minutaggio'].mean()
    max_durata = df['minutaggio'].max()

    print(f"Durata minima: {min_durata} minuti")
    print(f"Durata media: {media_durata:.2f} minuti")
    print(f"Durata massima: {max_durata} minuti")

    # Controlla se la lista è vuota
    # Se sì, stampa un messaggio.
    # La funzione restituisce None se non ci sono ricette.

    # Crea un DataFrame da pandas usando la lista di ricette.
    # Estrae le statistiche sul minutaggio delle ricette.
    # Trova la durata minima.
    # Calcola la durata media.
    # Trova la durata massima.

    # Stampa i risultati delle statistiche sulla durata.
    # Stampa la media formattata a due decimali.
    # Stampa i risultati delle statistiche sulla durata.
```

Output "Statistiche_Durata"

```
#Funzione Richiamata per mostrare le ricette con il minutaggio minore, maggiore e la media sul totale  
statistiche_durata(lista_ricette)
```

→ *Funzione Richiamata*

Output Restituito

```
Durata minima: 10 minuti  
Durata media: 38.61 minuti  
Durata massima: 120 minuti
```

Ogni qualvolta andremo ad Aggiungere o Eliminare una ricetta questo dato si aggiornerà automaticamente

Funzione "Filtraggio_Avanzato"

Definisce una funzione che permetta di avere Doppio Filtro: 1) Per minutaggio. 2) Per Ingrediente

```
# Definisce una funzione che permetta di avere Doppio Filtro: 1) Per minutaggio. 2) Per Ingrediente (Start2impact -> Filtraggio Avanzato)
def filtraggio_avanzato(lista):

    """
    Filtra e visualizza le ricette in base a un doppio criterio:
    minutaggio massimo e presenza di un ingrediente specifico.

    Args:
        lista (list): Lista che contiene tutte le ricette,
                       dove ogni ricetta è rappresentata come un dizionario
                       con chiavi 'nome', 'ingredienti' e 'minutaggio'.

    Returns:
        None: Stampa i risultati della ricerca o un messaggio se la ricetta non è stata trovata.
    """

    while True:
        try:
            max_minutaggio = int(input("Inserisci il massimo minutaggio (in minuti): "))
            if max_minutaggio <= 0:
                raise ValueError("Il minutaggio non può essere negativo o uguale a 0.")
            break
        except ValueError as e:
            print(f"Errore: {e}. Per favore, inserisci un numero intero valido.")

        # Ciclo infinito per ottenere un input valido per il minutaggio massimo.

        # Richiede all'utente di inserire il massimo minutaggio e converte l'input in un intero.
        # Verifica se il minutaggio è un valore positivo(>0).

        # Esce dal ciclo se l'input è valido

        # Stampa un messaggio di errore se l'input non è valido.

    ingrediente = input("Inserisci l'ingrediente da cercare: ").lower()
    ricette_filtrate = []
    for ricetta in lista:
        if ricetta['minutaggio'] <= max_minutaggio:
            ingredienti_lower = [ingrediente_item.lower() for ingrediente_item in ricetta['ingredienti']]
            if ingrediente in ingredienti_lower:
                ricette_filtrate.append(ricetta)

        # Richiede all'utente di inserire un ingrediente da cercare, convertendolo in minuscolo per uniformità.
        # Inizializza una lista per memorizzare le ricette che soddisfano i criteri.
        # Scorre ogni ricetta nella lista fornita.
        # Verifica se il minutaggio della ricetta è inferiore o uguale al massimo specificato.
        # Crea una lista di ingredienti in minuscolo per la ricetta corrente convertiti in minuscolo.
        # Verifica se l'ingrediente (in minuscolo) è presente nella lista di ingredienti (anch'essa in minuscolo).
        # Aggiunge la ricetta alla lista dei risultati filtrati se soddisfa i criteri.

    if ricette_filtrate:
        for ricetta in ricette_filtrate:
            visualizza_ricette(ricetta_filtrate)

        # Controlla se ci sono ricette filtrate da mostrare.

        # Stampa le ricette filtrate chiamando la funzione di visualizzazione.
    else:
        print(f"Nessuna ricetta trovata con meno di {max_minutaggio} minuti e contenente '{ingrediente}'.") # Stampa un messaggio se non ci sono ricette che soddisfano i criteri di filtraggio.
```


Output "Filtraggio_Avanzato"

```
#Funzione Richiamata per mostrare le ricette con al suo interno un Doppio Filtro, ovvero il primo in base al minutaggio  
filtraggio_avanzato(lista_ricette)
```

→ Funzione Richiamata

Output Restituito

```
Inserisci il massimo minutaggio (in minuti): 15  
Inserisci l'ingrediente da cercare: Pomodoro  
Nome: Caprese  
Ingredienti: Mozzarella, Pomodoro, Basilico, Olio d'Olive, Sale  
Minutaggio: 10 minuti  
-----
```

Restituisce tutte le ricette con un minutaggio \leq "al valore inserito" e l'ingrediente da cercare

Funzione "Filtraggio_Avanzato2"

Definisce una funzione che permetta di visualizzare la/e ricetta/e attraverso il filtro di due Ingredienti

```
# Definisce una funzione che permetta di visualizzare la/e ricetta/e attraverso il filtro di due Ingredienti (Start2impact -> Filtraggio Avanzato)
def filtraggio_avanzato2(lista):

    """
    Filtra e visualizza le ricette che contengono due ingredienti specifici.

    Args:
        lista (list): Lista che contiene tutte le ricette,
                     dove ogni ricetta è rappresentata come un dizionario
                     con chiavi 'nome', 'ingredienti' e 'minutaggio'.

    Returns:
        None: Stampa i risultati della ricerca o un messaggio se la ricetta non è stata trovata.
    """

    ingrediente1 = input("Inserisci il primo ingrediente da cercare: ").strip().lower()
    ingrediente2 = input("Inserisci il secondo ingrediente da cercare: ").strip().lower()
    ricette_filtrate = []
    for ricetta in lista:
        ingredienti_lower = [ingrediente_item.lower() for ingrediente_item in ricetta['ingredienti']]
        if ingrediente1 in ingredienti_lower and ingrediente2 in ingredienti_lower:
            ricette_filtrate.append(ricetta)

    if ricette_filtrate:
        for ricetta in ricette_filtrate:
            visualizza_ricetta(ricetta_filtrate)
    else:
        print(f"Nessuna ricetta trovata contenente entrambi '{ingrediente1}' e '{ingrediente2}'")
```

Richiede all'utente di inserire il primo ingrediente e lo converte in minuscolo.
Richiede all'utente di inserire il secondo ingrediente e lo converte in minuscolo.
Inizializza una lista per memorizzare le ricette che soddisfano i criteri.
Scorre ogni ricetta nella lista fornita
Converte gli ingredienti della ricetta in minuscolo per il confronto
Verifica se entrambe le condizioni siano soddisfatte
Aggiunge la ricetta alla lista dei risultati filtrati se soddisfa i criteri.

Controlla se ci sono ricette filtrate da mostrare.

Stampa le ricette filtrate chiamando la funzione di visualizzazione.

Stampa un messaggio se non ci sono ricette che soddisfano i criteri di filtraggio.

Output "Filtraggio_Avanzato2"

```
#Funzione Richiamata per mostrare le ricette con al suo interno un Doppio Filtro, ovvero due Ingredienti  
filtraggio_avanzato2(lista_ricette)
```

→ Funzione Richiamata

Output Restituito

```
Inserisci il primo ingrediente da cercare: Aglio  
Inserisci il secondo ingrediente da cercare: Cipolla  
Nome: Falafel  
Ingredienti: Ceci, Aglio, Cipolla, Prezzemolo, Coriandolo, Cumino, Farina  
Minutaggio: 45 minuti  
-----  
Nome: Ratatouille  
Ingredienti: Melanzane, Zucchine, Peperoni, Pomodoro, Cipolla, Aglio, Olio d'Oliva  
Minutaggio: 60 minuti  
-----  
Nome: Falafel  
Ingredienti: Ceci, Aglio, Cipolla, Prezzemolo, Coriandolo, Cumino, Farina  
Minutaggio: 45 minuti  
-----  
Nome: Ratatouille  
Ingredienti: Melanzane, Zucchine, Peperoni, Pomodoro, Cipolla, Aglio, Olio d'Oliva  
Minutaggio: 60 minuti  
-----
```

→ Restituisce tutte le ricette con i due ingredienti specificati

GitHub



*Clicca qui per
visualizzare il Codice*