Pseudocodice del gioco Sudoku Funzioni

Gruppo 1

Funzioni

FUNZIONE inizializzare_griglia
INPUT
impostazioni_selezionate, impostazioni selezionate in precedenza dall'utente, impostazioni
OUTPUT
griglia_sudoku, griglia del sudoku generata con le impostazioni selezionate, griglia
pseudocodice:

```
griglia_sudoku = riempire_griglia(impostazioni_utente)
```

FUNZIONE giocare_partita
INPUT
partita, partita che verrà giocata, partita
OUTPUT
partite_salvate, insieme delle partite salvate dall'utente, array di file_binario
DATI DI LAVORO
file_salvataggio, file che verrà utilizzato per salvare la partita, file_binario
comando_utente, variabile che prenderà il valore letto da tastiera, carattere
pseudocodice:

```
file_salvataggio = partita_leggere_nome(partita)

RIPETI
    stampa_a_video(INTERFACCIA_SCHERMATA_DI_GIOCO)
    comando_utente = invertire_maiuscolo_minuscolo(leggere_da_tastiera())
    SE(comando_utente = 'S')
        ALLORA file_salvataggio = salvare_partita(file_salvataggio, partita)
        stampare_a_video(INTERFACCIA_CARICARE_PARTITA)
        comando_utente = convertire_lettere_in_numeri(leggere_da_tastiera())
        SE(comando_utente < 6 ∧ comando_utente > 0):
            ALLORA elemento di partite_salvate in posizione comando_utente =
   file_salvataggio
        FINE
    FINE
    SE(comando_utente = 'I')
        riga = leggere_da_tastiera()
        colonna = leggere_da_tastiera()
        valore = leggere_da_tastiera()
        partita = partita_scrivere_griglia(aggiornare_griglia(griglia_sudoku,
   valore, riga, colonna), partita)
    FINE
FINCHE (comando_utente ≠ 27)
```

FUNZIONE salvare_partita
INPUT
file_salvataggio, le scelto dall'utente dove verrà salvata la partita in corso, file_binario
partita_da_salvare, partita attualmente in corso da salvare, partita
OUTPUT
file_salvataggio file con al suo interno la partita salvata dall'utente, file_binario
pseudocodice:

```
file_salvataggio = scrivere_elemento(partita_da_salvare)
```

FUNZIONE caricare_partita
INPUT
partite_salvate, elenco delle partite salvate dall'utente, array di file_binario
OUTPUT
partita_caricata, partita da caricare scelta dall'utente, partita
DATI DI LAVORO
file_scelto, file scelto dall'utente, file_binario
comando_utente, comando scelto dall'utente, carattere
pseudocodice:

```
comando_utente = leggere_da_tastiera()
file_scelto = elemento di partite_salvate in posizione comando_utente
SE (leggere_posizione(file_scelto) > 1)
    ALLORA partita_caricata = leggere_elemento(file_scelto)
FINE
```

FUNZIONE selezionare_difficolta
INPUT
difficolta_scelta, difficoltà che verrà scelta, numero naturale >0
OUTPUT
difficolta_scelta, difficoltà scelta dall'utente, numero naturale >0
DATI DI LAVORO
comando_utente, assumerà il valore dei comandi da tastiera, carattere
Pseudocodice:

```
difficolta_scelta = DIFFICOLTA_STANNDARD
stampa_a_video(INTERFACCIA_MENU_DIFFICOLTA)
comando_utente = leggere_da_tastiera()

SE (comando_utente = '1')
    ALLORA difficolta_scelta = DIFFICOLTA_STANNDARD
FINE
SE (comando_utente = '2')
    ALLORA difficolta_scelta = DIFFICOLTA_MEDIA
FINE
SE (comando_utente = '3')
    ALLORA difficolta_scelta = DIFFICOLTA_DIFFICILE
FINE
```

FUNZIONE selezionare_dimensione_griglia
INPUT
dim_griglia_scelta, dimensione che verrà scelta dall'utente, numero naturale >0
OUTPUT
dim_griglia_scelta, dimensione scelta dall'utente, numero naturale >0
Pseudocodice:

```
DATI DI LAVORO
```

```
comando_utente, dato che prender il valore dei valori da tastiera, carattere

stampa_a_video(INTERFACCIA_DIMENSIONE_GRIGLIA)
comando_utente = leggere_da_tastiera()
dim_griglia_scelta = DIM_GRIGLIA_STANNDARD

SE (comando_utente = '1')
    ALLORA dim_griglia_scelta = DIMENSIONE_GRIGLIA_PICCOLA
FINE
SE (comando_utente = '2')
    ALLORA dim_griglia_scelta = DIMENSIONE_GRIGLIA_MEDIA
FINE
SE (comando_utente = '3')
    ALLORA dim_griglia_scelta = DIMENSIONE_GRIGLIA_GR∧E
FINE
```

FUNZIONE inizializzare_partita
INPUT
impostazioni_utente, impostazioni dell'utente, impostazioni
griglia, griglia generata dalle impostazioni_utente, griglia
nome_partita, nome selezionato dall'utente per la partita, stringa
partita, partita che verrà inizializzata, partita
OUTPUT
partita, partita inizializzata, partita
pseudocodice:

```
partita = partita_scrivere_impostazioni(partita, impostazioni_utente)
partita = partita_scrivere_griglia(partita, griglia)
partita = partita_scrivere_nome(partita, nome_partita)
```

FUNZIONE aggiornare_griglia
INPUT
griglia, griglia che andrà aggiornata, griglia
valore, valore che verrà inserito all'interno della griglia, numero naturale
riga, riga della griglia dove verrà inserito il valore, numero naturale >0
colonna, colonna della griglia dove verrà inserito il valore, numero naturale >0
OUTPUT
griglia, griglia aggiornata con il nuovo valore, griglia
pseudocodice:

```
SE(validare_input_utente(riga, colonna, valore, griglia) = VERO)
    ALLORA griglia = griglia_scrivere_valore(griglia, valore)
FINE
```

FUNZIONE iniziare_partita
INPUT
impostazioni_di_gioco, impostazioni del gioco, impostazioni
difficolta_scelta, difficoltà scelta dall'utente, numero naturale
dim_griglia_scelta, dimensione della griglia scelta dall'utente, numero naturale
nome_partita, nome della partita, stringa
partita, partita che verrà iniziata ed inserita all'interno di partite_salvate, partita

OUTPUT
partite_salvate, array delle partite salvate, array di file_binario
DATI DI LAVORO
griglia_sudoku, griglia della partita, griglia
Pseudocodice:

```
RIPETI
    stampa_a_video(INTERFACCIA_MENU_OPZIONI)
    comando_utente = leggere_da_tastiera()
    SE (comando_utente = '1')
        ALLORA difficolta_scelta = selezionare_difficolta(difficolta_scelta)
    FINE
    SE(comando_utente = '2')
        ALLORA dim_griglia_scelta =
   selezionare_dimensione_griglia(dim_griglia_scelta)
    FINE
    SE (comando_utente = '3')
        ALLORA nome_partita = leggere_da_tastiera()
    FINE
    SE (comando_utente = '4')
        ALLORA
            impostazioni_di_gioco =
   impostare_parametri_di_gioco(impostazioni_di_gioco, difficolta_scelta,
   dim_griglia_scelta)
            griglia_sudoku = inizializzare_griglia(impostazioni_di_gioco)
            partita = inizializzare_partita(impostazioni_di_gioco, griglia_sudoku,
   nome_partita, partita)
            partite_salvate = giocare_partita(partita)
    FINE
FINCHE (comando_utente ≠ '4' OR comando_utente ≠ '5')
```

FUNZIONE menu_principale
INPUT
comando_utente, comando selezionato dall'utente, carattere
OUTPUT
partite_salvate, array di partite salvate, array di file_binario
DATI DI LAVORO
difficolta_scelta, difficoltà scelta dall'utente, numero naturale
dim_griglia_scelta, dimensione della griglia scelta dall'utente, numero naturale
impostazioni_gioco, impostazioni di gioco scelte dall'utente, impostazioni
Pseudocodice:

```
RIPETI
    stampa_a_video(INTERFACCIA_MENU_PRINCIPALE)
    comando_utente = leggere_da_tastiera()
    SE (comando_utente = '1')
        ALLORA difficolta_scelta = DIFFICOLTA_STANNDARD
            dim_griglia_scelta = DIM_GRIGLIA_STANNDARD
            impostazioni_gioco =
   impostazioni_scrivere_dimensione_griglia(impostazioni_gioco,
   DIM_GRIGLIA_STANNDARD)
            impostazioni_gioco =
```

```
    impostazioni_scrivere_difficolta(impostazioni_gioco, DIFFICOLTA_STANNDARD)
            partite_salvate = iniziare_partita(impostazioni_gioco,
    difficolta_scelta, dim_griglia_scelta, nome_partita, partita)
     ALTRIMENTI
         SE(comando_utente = '2')
             ALLORA
                 partita_caricata = caricare_partita(partite_salvate)
                 partite_salvate = giocare_partita(partita_caricata)
         FINE
     FINE
FINCHE(comando_utente ≠ '3')
```

FUNZIONE n_numeri_di_griglia
INPUT
impostazioni_gioco, impostazioni scelte dall'utente, impostazioni
OUTPUT
numeri_da_inserire_in_griglia, numero di valori da inserire nella griglia, numero naturale >0
DATI DI LAVORO
difficolta, difficoltà scelta dall'utente, numero naturale [0,2] (0 facile, 1 media, 2 difficile)
dimensione_griglia, dimensione della griglia scelta in base alle impostazioni
pseudocodice:

```
difficolta = impostazioni_leggere_difficolta(impostazioni_gioco)
dimensione_griglia = griglia_leggere_dimensione(impostazioni_gioco)

SE(difficolta = 0)
    ALLORA numeri_da_inserire_in_griglia = ((dimensione_griglia)^2
    *PERCENTUALE_DIFFICOLTA_FACILE)
FINE
SE(difficolta = 1)
    numeri_da_inserire_in_griglia = ((dimensione_griglia)^2 *
    PERCENTUALE_DIFFICOLTA_MEDIA)
FINE
SE(difficolta = 2)
    numeri_da_inserire_in_griglia = ((dimensione_griglia)^2
    *PERCENTUALE_DIFFICOLTA_DIFFICILE)
FINE
```

FUNZIONE riempire_griglia
INPUT
impostazioni_utente, impostazioni scelte dall'utente, impostazioni
OUTPUT
griglia, griglia con al suo interno dei numeri posizionati in maniera casuale, griglia
DATI DI LAVORO
numeri_da_inserire, rappresenta il numero di numeri che verranno inseriti all'interno della griglia, selezionati in base alla difficoltà e alla dimensione della griglia, numero naturale >0
coordinata_x, seleziona casualmente la posizione del valore all'interno delle righe, numero naturale >0
coordinate_y = seleziona casualmente la posizione del valore all'interno delle colonne, numero naturale >0
valore, valore che verrà inserito all'interno della griglia, numero naturale >0

pseudocodice:

```
i = 1
numeri_da_inserire = n_numeri_di_griglia(impostazioni_utente)

MENTRE(i <= numeri_da_inserire)
    coordinata_x = generare_n_casuale(1,
    griglia_leggere_dimensione(impostazioni_utente))
    coordinata_y = generare_n_casuale(1,
    griglia_leggere_dimensione(impostazioni_utente))
    SE(verificare_coordinate_e_valore(coordinata_x, coordinata_y) = VERO)
        ALLORA valore = generare_n_casuale(1,
    leggere_dimensione_griglia(impostazioni_utente))
        SE(verificare_numero_da_inserire(griglia, valore, coordinata_x,
    coordinata_y)= VERO)
            ALLORA
            SE(valore<= 9)
                ALLORA griglia = griglia_scrivere_valore(coordinata_x, coordinata_y,
    valore)
            FINE
            ALTRIMENTI convertire_numeri_in_lettere(valore)
            griglia = griglia_scrivere_valore(coordinata_x, coordinata_y, valore)
        FINE
    FINE
    i = i + 1
FINE
```

FUNZIONE convertire_numeri_in_lettere
INPUT
numero, numero che verrà convertito in lettera, numero naturale
OUTPUT
lettera, lettera che rappresenta il numero riscritto, carattere
pseudocodice:

```
lettera = numero + 55
```

FUNZIONE convertire_lettere_in_numeri
INPUT
lettere, lettera che verrà convertita in numero, carattere
OUTPUT
numero, numero che sarà trasformato dal carattere, numero naturale >0
pseudocodice:

```
SE(lettera >= 'A' ∧ lettera <= 'F')
    ALLORA numero = lettera - 55
FINE ALTRIMENTI numero = lettera - 48
```

FUNZIONE verificare_coordinate_e_valore
INPUT
griglia, griglia corrente con i valori già inseriti, griglia
coordinata_x, posizione riga che si vuole controllare, numero naturale >0

coordinata_y, posizione colonna che si vuole controllare, numero naturale >0
valore, valore che si vuole controllare, numero naturale
OUTPUT
validità, indica con falso se la cella è libera, booleano
DATI DI LAVORO
dimensione_griglia, rappresenta la dimensione della griglia, numero naturale >0
i, contatore per scorrere le celle della griglia, numero naturale >0
j, contatore per scorrere le celle della griglia, numero naturale >0
valore_cella, valore presente in una cella della griglia, numero naturale
pseudocodice:

```
validit = VERO
dimensione_griglia = griglia_leggere_dimensione(impostazioni_utente)

SE(griglia_leggere_valore(coordinata_x, coordinata_y) ≠ '')
    ALLORA validit = FALSO
FINE
```

FUNZIONE controllare_riga
INPUT
griglia, griglia del sudoku in cui verificare se il numero può essere inserito nella riga, griglia
riga, riga in cui controllare, numero naturale
numero_da_inserire, numero da verificare che non sia presente nella riga, numero naturale
dimensione_sudoku, dimensione di un lato del sudoku, numero naturale
OUTPUT
corretto, indica se il numero può essere inserito, booleano
DATI DI LAVORO
j, indice delle colonne della griglia del sudoku, numero naturale >0
pseudocodice:

```
corretto = VERO
j = 1
MENTRE(j<= dimensione_sudoku(griglia))
    SE(griglia_leggere_valore(sudoku,riga,j) = numero_da_inserire)
        corretto = FALSO
    FINE
    j = j + 1
FINE
```

FUNZIONE controllare_colonna
INPUT
griglia, griglia del sudoku in cui verificare se il numero può essere inserito nella colonna, griglia
colonna, colonna in cui controllare, numero naturale
numero_da_inserire, numero da verificare che non sia presente nella colonna, numero naturale
dimensione_sudoku, dimensione di un lato del sudoku, numero naturale
OUTPUT
corretto, indica se il numero può essere inserito, booleano
DATI DI LAVORO
i, indice delle righe della griglia del sudoku, numero naturale >0
pseudocodice:

```
corretto = VERO
i = 1
MENTRE(i<= dimensione_sudoku(griglia))
    SE(griglia_leggere_valore(sudoku,i,colonna) = numero_da_inserire)
        corretto = FALSO
    FINE
    i = i + 1
FINE
```

FUNZIONE controllare_regione

INPUT

griglia, griglia del sudoku in cui verificare se il numero può essere inserito, griglia
riga, indica l'inizio dell'indice della regione, numero naturale
colonna, indica l'inizio dell'indice della regione, numero naturale
numero_da_inserire, numero da verificare che non sia presente nella regione, numero naturale
dimensione_sudoku, dimensione di un lato del sudoku, numero naturale

OUTPUT

corretto, indica se il numero può essere inserito, booleano

DATI DI LAVORO

i, indice delle righe della regione del sudoku, numero naturale
j, indice delle colonne della griglia del sudoku, numero naturale >0

pseudocodice:

```
corretto = VERO
i = riga
MENTRE(i<= riga + dimensione_regione)
    j = colonna
    MENTRE(j < colonna + dimensione_regione)
        SE(griglia_leggere_valore(sudoku, i, j) = numero_da_inserire)
            ALLORA corretto = FALSO
        FINE
        j = j + 1
    FINE
    i = i + 1
FINE
```

FUNZIONE verificare_numero_da_inserire

INPUT

griglia, griglia in cui verificare se il numero inserito è coretto, griglia
numero_da_inserire, numero che si vuole inserire, numero intero
riga, indice delle righe in cui si vuole inserire il numero, numero intero
colonna, indice delle colonne in cui si vuole inserire il numero, numero intero

OUTPUT

corretto, indica se è possibile inserire il numero, booleano

DATI DI LAVORO

i, indice delle righe del sudoku, numero naturale >0
j, indice delle colonne del sudoku, numero naturale >0
dimensione_griglia, dimensione del lato del sudoku, numero intero

pseudocodice:

```
corretto = VERO
dimensione_sudoku = griglia_leggere_dimensione(griglia)

corretto = controllare_riga(sudoku, riga, numero_da_inserire, dimensione_griglia)

SE(corretto = VERO)
    ALLORA corretto = controllare_colonna(sudoku, colonna, numero_da_inserire,
    dimensione_sudoku)
FINE

SE(corretto = VERO)
    ALLORA riga = riga - calcolare_resto_intero(riga, dimensione_regione)
    colonna = colonna - calcolare_resto_intero(colonna, dimensione_regione)
    corretto = controllare_regione(sudoku, riga, colonna, numero_da_inserire,
    calcolare_radice_quadrata(dimensione_sudoku))
FINE
```

FUNZIONE calcolare_resto_intero
INPUT
numeratore, numeratore dell'operazione, numero intero
denominatore, denominatore dell'operazione, numero intero
OUTPUT
r, resto della divisione, numero intero
DATI DI LAVORO
q, quoziente della divisione, numero intero
pseudocodice:

```
q = 0
MENTRE(a >= b)
    q = q + 1
    a = a - b
FINE
r = a
```

FUNZIONE calcolare_radice_quadrata
INPUT
radicando, numero di da operare con la radice, numero naturale
OUTPUT
radice, risultato della radice del radicando, numero naturale
pseudocodice:

```
trovato = FALSO
radice = 2
radice_quad = radice * radice
MENTRE(radicando >= radice_quad ∧ trovato = FALSO)
    SE(radice_quad = radicando)
        ALLORA trovato = VERO
        radice = radice
    FINE
    radice = radice + 1
    radice_quad = radice * radice
FINE
```

```
SE(trovato = FALSO)
    ALLORA radice = 0
FINE
```

FUNZIONE convertire_minuscolo_maiuscolo
INPUT
lettera, lettera da convertire dal minuscolo al maiuscolo, carattere
OUTPUT
lettera_convertita, lettera convertita da minuscolo a maiuscolo, carattere
Pseudocodice:

```
lettera_convertita = lettera
SE ((lettera_convertita >= 97) ∧ (lettera_convertita <= 122))
    ALLORA lettera_convertita = lettera - 32
FINE
```

FUNZIONE validare_input_utente
INPUT
riga, valore riga di cui controllare la correttezza, naturale > 0
colonna, valore colonna di cui controllare la correttezza, naturale > 0
valore, valore inserito di cui controllare la correttezza, carattere
griglia_gioco, griglia a cui si riferiscono riga colonna e valore, griglia
OUTPUT
validato, indica se tutti e 3 gli input sono validi o meno, booleano
DATI DI LAVORO
dim_griglia, indica la dimensione della griglia, naturale > 0
pseudocodice:

```
dim_griglia = griglia_leggere_dimensione(griglia_gioco)
valore = convertire_minuscolo_maiuscolo(valore)
validato = FALSO

SE((validare_riga_input(riga, dim_griglia)= VERO) ∧ (validare_colonna_input(colonna,
    dim_griglia) = VERO) ∧ (validare_valore_input(valore, dim_griglia) = VERO))
    ALLORA validato = VERO
FINE
```

FUNZIONE validare_valore_input
INPUT
valore, valore inserito di cui controllare la correttezza, carattere
dim_griglia, dimensione griglia in cui verrebbe inserito il valore, naturale > 0
OUTPUT
validato, indica se la riga è valida o meno, booleano
DATI DI LAVORO
dim_griglia_carattere, indica la dimensione della griglia convertita in lettera
pseudocodice:

```
validato = VERO
dim_griglia_carattere = Convertire_numeri_in_lettere(dim_griglia)
```

```
SE(dim_griglia >= 16 ∧ (valore < 48 OR (valore > 57 ∧ valore < 65) OR valore >
    dim_griglia_carattere ))
    ALLORA validato = FALSO
    ALTRIMENTI
        SE(dim_griglia < 16 ∧ (valore < 48 OR valore > 57))
            ALLORA validato = FALSO
        FINE
FINE
```

FUNZIONE validare_riga_input
INPUT
riga, valore riga di cui controllare la correttezza, naturale > 0
dim_griglia, dimensione griglia a cui si riferisce riga, naturale > 0
OUTPUT
validato, indica se la riga è valida o meno, booleano
psuedocodice:

```
validato = VERO
SE(riga < 0 OR riga > dim_griglia)
    ALLORA validato = FALSO
FINE
```

FUNZIONE validare_colonna_input
INPUT
colonna, valore colonna di cui controllare la correttezza, naturale > 0
dim_griglia, dimensione griglia a cui si riferisce colonna, naturale > 0
OUTPUT
validato, indica se la riga è valida o meno, booleano
pseudocodice:

```
validato = VERO
SE(colonna < 0 OR colonna > dim_griglia)
    ALLORA validato = FALSO
FINE
```

FUNZIONE impostare_parametri_di_gioco
INPUT
impostazioni_gioco, impostazioni del gioco, impostazioni
difficolta_scelta, difficoltà selezionata dall'utente, numero naturale [0,2]
dim_griglia_scelta, dimensione selezionata dall'utente, numero naturale [1, DIMENSIONE_GRIGLIA].
OUTPUT
impostazioni_gioco, parametri di gioco selezionati dall'utente, impostazioni
Pseudocodice:

```
difficolta_scelta = leggere_da_tastiera()
dim_griglia_scelta = leggere_da_tastiera()
impostazioni_gioco = scrivere_difficolta(impostazioni_gioco, difficolta_scelta)
impostazioni_gioco = scrivere_dimensione_griglia(impostazioni_gioco,
    dim_griglia_scelta)
```