PANORAMICA SULLE CLASSI

BOARD

- La classe Board è nata con lo scopo di creare una finestra generica e ridefinire all'interno di essa le funzioni principali della libreria ncurses che andremo a usare per tutte le nostre finestre.
- Al suo interno troviamo i seguenti attributi
  - `WINDOW *win`
  - `int win_x, int win_y`
- E i seguenti metodi
  - `WINDOW getWin()`
    - Restituisce l'attributo di tipo finestra `win`
  - `void init()`
    - Inizializza la finestra
  - `void addBorder()`
    - Aggiunge un bordo alla finestra
  - `void clear()`
    - Elimina tutto il contenuto della finestra
  - `void addCharAt(Position pos, chtype ch)`
    - Aggiunge un carattere `ch` in una posizione `pos`
  - `void addCharAt(int x, int y, chtype ch)`
    - Stesso metodo ma con la possibilità di passare le coordinate `x y` come posizione anziché una posizione
  - `void addStringAt(Position pos, char str[])`
    - Aggiunge una stringa alla posizione data
  - `void addStringAt(int x, int y, char str[])`
    - Stesso metodo ma con le coordinate `x y` come parametro

- `int getInput()`
  - Ritorna il codice del tasto appena premuto
- `char getCharAt(Position pos)`
  - Ritorna il carattere presente nella posizione `pos`
- `void rmCharAt(Position pos)`
  - Rimuove il carattere presente alla posizione `pos`
- `bool isEmpty(Position pos)`
  - Ritorna true se la posizione `pos` è vuota

MENU

- La classe Menu eredita Board ed è la classe che si occupa dei menu che vediamo all'interno del gioco, quello iniziale e quello 'di morte'.
- All'interno di questa classe si trova una struttura `Levels` che serve a rappresentare i livelli mediante una lista bidirezionale e i seguenti attributi:
  - `Board menuBoard`
  - `int choice`
  - `int level`
- E i seguenti metodi:
  - `void initLevels()`
    - Inizializza i livelli, ne aggiunge 3 mediante la funzione `addLevel`
  - `Board getMenuBoard`
    - Ritorna l'attributo `menuBoard`
  - `void showOptions()`
    - Mostra sulla `menuBoard` le opzioni disponibili all'interno del menu principale
  - `void showDeathOptions()`
    - Mostra sulla `menuBoard` le opzioni disponibili all'interno del menu che si apre alla fine di una partita

    - `void open()`
        - Apre il menu iniziale richiamando la funzione `showOptions()`
    - `void openDeath()`
        - Apre il menu di fine partita richiamando la funzione `showDeathOptions()`
    - `void getChoice()`
        - Ritorna l'attributo `choice`
    - `void resetChoice()`
        - Setta l'attributo `choice` al suo valore di default -1
    - `void showLevels()`
        - Mostra i livello selezionabili all'interno della `menuBoard`

SNAKE

- La classe che si occupa dell'oggetto serpente, rappresentato mediante una matrice booleana.
- Al suo interno troviamo i seguenti attributi:
    - `bool body[WIDTH][HEIGHT]`
    - `Position headPosition`
    - `Position tailPosition`
    - `Direction dirHistory`
        - Un array circolare utilizzato per ricordare le ultime direzioni del serpente
    - `int indexCircular`
        - Necessario per la logica di `dirHistory`
    - `char headIcon`
    - `char bodyIcon`
- E i seguenti metodi:
    - `Position getHead()`

    - `Position getTail()`
    - `char getHeadIcon()`
    - `char getBodyIcon()`
    - `bool move(Direction inputDirection)`
        - Si occupa del movimento del serpente, ritorna true se il movimento ha avuto successo
    - `bool snakeIsHere(Position p)`
        - Ritorna true se il serpente si trova in questa posizione
    - `void reset()`
        - Resetta la posizione del serpente al centro della griglia

SCRIBA

- La classe che si occupa di scrivere e leggere i dati da un file di testo `classifica.txt` e mostrarli all'utente inserendoli all'interno di una `Board`
- Al suo interno troviamo come unico attributo:
    - `Board scoreBoard`
        - La Board attraverso la quale l'utente può vedere la classifica
- E i metodi:
    - `void initBoard()`
        - Inizializza l'attributo `scoreBoard`
    - `void cleanPodium()`
        - Rimuove l'intero contenuto del file
    - `void insertRec(char righe[][RAW_LEN], int pos, int pts, const char level[])`
        - Funzione ausiliaria ricorsiva per inserire un punteggio nella classifica
    - `void insert(int pts, const char level[]`

        - Funzione che con l'ausilio di `insertRec()` inserisce all'interno della classifica un nuovo punteggio nella posizione corretta
    - `void showScore()`
        - Funzione che mediante la `scoreBoard` mostra all'utente la classifica

APPLE

- La classe che si occupa dell'oggetto `Apple` avente due attributi:
    - `Position position`
    - `char icon`
- E due metodi getter che non fanno altro che restituire gli attributi (protetti):
    - `char getIcon()`
    - `Position getPosition()`

GAME

- La classe che gestisce tutta la logica del gioco, con al suo interno i seguenti attributi:
    - `Board board`
        - La board sulla quale viene eseguito il gioco vero e proprio
    - `Menu menu`
    - `Apple apple`
    - `Snake snake`
    - `Scriba scriba`
    - `GameState gameState`
    - `int score`
    - `Direction currentDirection`
    - `Position center`

    - `time_t startTime`
    - `int timeLimit`
    - `time_t pauseTime`
    - `int timePaused`
- E i seguenti metodi:
    - `Scriba getScriba()`
    - `Board getBoard()`
    - `Menu getMenu()`
    - `GameState getGameState()`
    - `void printScore()`
        - Stampa a schermo il punteggio attuale in `score`
    - `void startGame()`
        - Si occupa delle operazioni che precedono l'inizio della partita
    - `void exitGame()`
        - Fa uscire l'utente dal programma
    - `void openMenu()`
        - Apre il menu di gioco
    - `void processInput()`
        - Processa l'input fornito dall'utente sul menu di gioco
    - `void processInputDeath()`
        - Processa l'input fornito dall'utente sul menu di morte
    - `void openDeathMenu()`
        - Apre il menu di fine partita
    - `Position randomPosition()`
        - Ritorna una posizione casuale, utile per creare le mele
    - `void printApple(Position p)`
        - Stampa a schermo le mele in `apple[]`
    - `void removeApple(Position p)`

    - Rimuove una mela quando viene mangiata
- `void spawnApples()`
    - Crea le mele nel gioco grazie a `randomPosition()` e `printApple()`
- `void initPrintSnake()`
    - Stampa il serpente al centro della griglia
- `void updateSnake(Direction inputDirection)`
    - Aggiorna la posizione del serpente e ne regola il movimento
- `void printTimer()`
    - Stampa a schermo il timer di gioco