882713 Cantoni

Generated by Doxygen 1.11.0

Chapter 1

882713_Cantoni

Questo progetto implementa il gioco del Domino

1.1 Author

Cantoni Letizia, matricola: 882713

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Data Structure Documentation

4.1 Tile Struct Reference

```
#include <tile.h>
```

Data Fields

- int **side1**
- int **side2**
- int **occupied**
- int **vertical**
- int **special**

4.1.1 Detailed Description

Tile (p. 7) is the chosen structure to represent the tile in the game side1 is the number left in the tile and side2 is the number right [side1|side2] [2|6] occupied is used to mark a tile not yet played in the player tiles, And in the board game it mark a tile that is the board. vertical is used to see if a tile is vertical [2] or using the graphic in the boardgame: {2} [6] {6} special is used to see if the tile is special, like [0|0] or [11|11] or [12|21]

4.1.2 Field Documentation

4.1.2.1 occupied

```
int occupied
```

4.1.2.2 side1

```
int side1
```

4.1.2.3 side2

```
int side2
```

4.1.2.4 special

```
int special
```

4.1.2.5 vertical

```
int vertical
```

The documentation for this struct was generated from the following file:

- C:/Users/letyc/Desktop/882713_Cantoni/include/ **tile.h**

Chapter 5

File Documentation

5.1 C:/Users/letyc/Desktop/882713_Cantoni/include/checks.h File Reference

This file defines the functions that are used to check the insertion of a tile into the game.

```
#include <stdio.h>
#include <stdlib.h>
#include "tile.h"
#include "gameMove.h"
```

Functions

- int **possibleRows** (**Tile** ∗∗boardGame, int numTiles)
- int **possibleLeftCell** (**Tile** ∗boardGame, int numTiles)

  Finds the possible left cell in which a tile can be add on the game board in a specific row.
- int **possibleRightCell** (**Tile** ∗boardGame, int numTiles)

  Finds the possible right cell in which a tile can be add on the game board in a specific row.
- int **isValid** (int row, int col, **Tile** selectedtile, char pos, **Tile** ∗∗boardGame, int numTiles)

  Checks if a move is valid.
- int **isValidFlat** (int row, int col, **Tile** selectedtile, char pos, **Tile** ∗∗boardGame, int numTiles)

  Checks if a flat move is valid (flat move = the tile to insert is not vertical).
- int **checkEndgame** (**Tile** ∗playersTiles, int numTiles, **Tile** ∗∗boardGame)

  Checks if there are any available moves or the game is finished. is true when the game is finished.

5.1.1 Detailed Description

This file defines the functions that are used to check the insertion of a tile into the game.

Author

Cantoni Letizia 882713

5.1.2 Function Documentation

5.1.2.1 checkEndgame()

```
int checkEndgame (
             Tile * playersTiles,
             int numTiles,
             Tile ** boardGame)
```

Checks if there are any available moves or the game is finished. is true when the game is finished.

Checks if a flat move is valid (flat move = the tile to insert is not vertical).

Parameters

playersTiles	The player's tiles.
numTiles	The number of tiles the player has.
boardGame	The game board.

Returns

int Returns 1 if there are no available moves, 0 otherwise.

Slides all the possible tile of the player

if a tile is occupied it means that it has not yet been played

for every 'possible' row it checks if a certain move is valid right or left

checks if the move is valid right or left

if it finds a possible move it exits

if no moves are possible

5.1.2.2 isValid()

```
int isValid (
             int row,
             int col,
             Tile selectedtile,
             char pos,
             Tile ** boardGame,
             int numTiles)
```

Checks if a move is valid.

Finds the possible right cell in which a tile can be add on the game board in a specific row.

Parameters

row	The row to check.
col	The column to check.
selectedtile	The selected tile.
pos	The position to check.(left 's' or right 'd')
boardGame	The game board.
numTiles	The number of tiles in the game.

Returns

int Returns 1 if the move is valid, 0 otherwise.

I check If where I want to put the tile is not occupied

and if the tile to insert is not vertical

if I want to put the tile vertical I have to look that the cell below is unoccupied

If i want to put the tile right

If the upper part of the tile can be placed

here I look to see if the tile I'm placing it next to is NOT vertical and it's compatible with my card

here I look to see if the tile I'm placing it next to is vertical and it's compatible with my card

If i want to put the tile left

5.1.2.3 isValidFlat()

```
int isValidFlat (
             int row,
             int col,
             Tile selectedtile,
             char pos,
             Tile ** boardGame,
             int numTiles)
```

Checks if a flat move is valid (flat move = the tile to insert is not vertical).

Checks if a move is valid.

Parameters

row	The row to check.
col	The column to check.
selectedtile	The selected tile.
pos	The position to check.
boardGame	The game board.
numTiles	The number of tiles in the game.

Returns

int Returns 1 if the move is valid, 0 otherwise.

If is special i can place it everywhere

If i am trying to put the tile between two other tiles

all the two tile must be compatible with the one i want to insert, in this case i am trying to insert the tile vertically

all the two tile must be compatible with the one i want to insert, in this case i am trying to insert the tile horizontally

If i am trying to put the tile right

If the adjacent right tile is vertical and compatible with mine

If the adjacent right tile is NOT vertical and compatible with mine

If i am trying to put the tile left

If the adjacent left tile is compatible with mine

5.1.2.4 possibleLeftCell()

```
int possibleLeftCell (
             Tile * boardGame,
             int numTiles)
```

Finds the possible left cell in which a tile can be add on the game board in a specific row.

Counts the possible rows in which a tile can be added on the game board.

Parameters

boardGame	The game board.
numTiles	The number of tiles in the game.

Returns

int Returns the index of the possible left cell, or -1 if not found.

5.1.2.5 possibleRightCell()

```
int possibleRightCell (
             Tile * boardGame,
             int numTiles)
```

Finds the possible right cell in which a tile can be add on the game board in a specific row.

Finds the possible left cell in which a tile can be add on the game board in a specific row.

Parameters

boardGame	The game board.
numTiles	The number of tiles in the game.

Returns

int Returns the index of the possible right cell, or -1 if not found.

5.1.2.6 possibleRows()

```
int possibleRows (
             Tile ** boardGame,
             int numTiles)
```

Counts the possible rows in which a tile can be added on the game board. That is, look at which rows are occupied by at least one tile

Parameters

boardGame	The game board.
numTiles	The number of tiles in the game.

Returns

int Returns the number of possible rows.

Move to the next row as soon as we find an occupied cell

5.2 checks.h

Go to the documentation of this file.

```
#ifndef CHECKS_H
#define CHECKS_H
#include <stdio.h>
#include <stdlib.h>
#include "tile.h"
#include "gameMove.h"

int possibleRows(Tile **boardGame,int numTiles);
int possibleLeftCell(Tile *boardGame, int numTiles);
int possibleRightCell(Tile *boardGame, int numTiles);
int isValid (int row,int col,Tile selectedtile,char pos,Tile **boardGame,int numTiles);
int isValidFlat (int row,int col,Tile selectedtile,char pos,Tile **boardGame,int numTiles);
int checkEndgame(Tile *playersTiles, int numTiles, Tile **boardGame);
#endif
```

5.3 C:/Users/letyc/Desktop/882713_Cantoni/include/gameAi.h File Reference

This file defines the functions that are used to play the game in AI mode.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include "gameMove.h"
#include "checks.h"
#include "printGraphic.h"
```

Functions

- void **sortTiles** (**Tile** ∗tiles, int numTiles)

  Function to sort the player tiles based on their score (es.[3|2] has a score=5).
- int **MoveAi** (int choice, **Tile** selectedTiles, **Tile** ∗playersTiles, int numTiles, **Tile** ∗∗boardGame)

  Function to make a move for the AI.
- void **gameStartAi** (**Tile** ∗playersTiles, int numTiles, **Tile** ∗∗boardGame)

  Initial function for the AI to choose a tile.

5.3.1 Detailed Description

This file defines the functions that are used to play the game in AI mode.

Author

Cantoni Letizia 882713

5.3.2 Function Documentation

5.3.2.1 gameStartAi()

```
void gameStartAi (
             Tile * playersTiles,
             int numTiles,
             Tile ** boardGame)
```

Initial function for the AI to choose a tile.

Function to make a move for the AI.

Parameters

playersTiles	The array of tiles available to the AI.
numTiles	The number of tiles available to the AI.
boardGame	The current state of the game board.

it sorts the tiles so the first match you find is the one with the highest score

if the tile is not already played

if the tile was not inserted i carry on

else i wait just for the user to see the move and then i start checking the tiles from the start

if the tile has already been played i go to the next

if there are no avaiable moves exit

if all the tiles have been played print the phrase below else print the remaining tiles

5.3.2.2 MoveAi()

```
int MoveAi (
             int choice,
             Tile selectedTiles,
             Tile * playersTiles,
             int numTiles,
             Tile ** boardGame)
```

Function to make a move for the AI.

Function to sort the player tiles based on their score (es.[3|2] has a score=5).

Parameters

choice	The index of the tile chosen by the AI.
selectedTiles	The tile chosen by the AI.
playersTiles	The array of tiles available to the AI.
numTiles	The number of tiles available to the AI.
boardGame	The current state of the game board.

Returns

int Returns 1 if the move was successful, 0 otherwise.

Initialize the random number generator with the current time

Generates a random number between 0 and 4 that give us a probability of 80% that is != 0

Possible row where to insert the tile

I will try for every row if there is a match

Possible right column where to insert the tile

Possible left column where to insert the tile

If there is a possible right cell

If the randon number is !=0 the vertical is true, so we try to place it vertical

If the moveis not valid with the vertical tile, we try with the horizonal tile

if the move is valid we add it

else we try with the flipped tile and if the move is valid we add it

If there is a possible left cell

If the randon number is !=0 the vertical is true, so we try to place it vertical

If the move is not valid with the vertical tile, we try with the horizonal tile

if the move is valid we add it

else we try with the flipped tile and if the move is valid we add it

If there is a possible right AND left cell

If the move is not valid on either left or right with the vertical tile, we try with the horizonal tile

if the move is valid left we add it

else if the move is valid with the flipped tile left we add it

if the move is valid right we add it

else if the move is valid with the flipped tile right we add it

5.3.2.3 sortTiles()

```
void sortTiles (
              Tile * tiles,
             int numTiles)
```

Function to sort the player tiles based on their score (es.[3|2] has a score=5).

Parameters

tiles	The array of tiles to sort(the player tiles).
numTiles	The number of tiles available to the AI.

Move special tiles to the second position so it's likely it mirrores the tile with the highest score

5.4 gameAi.h

Go to the documentation of this file.

```

#ifndef GAME_AI_H
#define GAME_AI_H
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include "gameMove.h"
#include "checks.h"
#include "printGraphic.h"

void sortTiles(Tile *tiles, int numTiles);
int MoveAi(int choice,Tile selectedTiles,Tile *playersTiles,int numTiles,Tile **boardGame);
void gameStartAi(Tile *playersTiles,int numTiles,Tile **boardGame);
#endif
```

5.5 C:/Users/letyc/Desktop/882713_Cantoni/include/gameInteractive.h File Reference

This file defines all the functions that are used to play in the Interactive game mode.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include "tile.h"
#include "printGraphic.h"
#include "userInputs.h"
#include "checks.h"
```

Functions

- void **makeMove** (**Tile** selectedTiles, int numTiles, **Tile** ∗∗boardGame, int choice, **Tile** ∗playersTiles)

  Function to get the input from the player to where insert the tile.
- void **tileOrientation** (int choice, **Tile** selectedTiles, **Tile** ∗playersTiles, int numTiles, **Tile** ∗∗boardGame, int first)

  Given a tile, take the input of the player to whether to flip it, put it vertical,confirm or exit.
- void **gameStartInteractive** (**Tile** ∗playersTiles, int numTiles, **Tile** ∗∗boardGame)

  Initial function to choose the tile (interactive mode).

5.5.1 Detailed Description

This file defines all the functions that are used to play in the Interactive game mode.

Author

Cantoni Letizia 882713

5.5.2 Function Documentation

5.5.2.1 gameStartInteractive()

```
void gameStartInteractive (
             Tile * playersTiles,
             int numTiles,
             Tile ** boardGame)
```

Initial function to choose the tile (interactive mode).

Given a tile, take the input of the player to whether to flip it, put it vertical,confirm or exit.

Parameters

playersTiles	The player's tiles.
numTiles	The number of tiles in the game.
boardGame	The game board.

Get the tile the player wants to play

This happens only for the first move

The special tiles [11|11] and [12|21] cannot be placed in the first position

Get the tile the player wants to play

If all the tiles have been inserted print

else print the remaining tiles

5.5.2.2 makeMove()

```
void makeMove (
             Tile selectedTiles,
             int numTiles,
             Tile ** boardGame,
             int choice,
             Tile * playersTiles)
```

Function to get the input from the player to where insert the tile.

Parameters

selectedTiles	The tile selected ti insert.
numTiles	The number of tiles in the game.
boardGame	The game board.
choice	The chosen tile.
playersTiles	The player's tiles.

the number of the row a tile can be placed

if the number of rows>1 asks the player which row he wants to put it in

Ask the player where he wants to put the tile, left or right

retrieve the right column where to put the tile

retrieve the left column where to put the tile

if there is a column for the move

if the move is valid

makes the move

5.5.2.3 tileOrientation()

```
void tileOrientation (
              int choice,
               Tile selectedTiles,
               Tile * playersTiles,
              int numTiles,
               Tile ** boardGame,
              int first)
```

Given a tile, take the input of the player to whether to flip it, put it vertical,confirm or exit.

Function to get the input from the player to where insert the tile.

Parameters

choice	The chosen tile.
selectedTiles	The selected tile.
playersTiles	The player's tiles.
numTiles	The number of tiles in the game.
boardGame	The game board.
first	Whether it's the first move.

choose what to do with the tile: flip it, put it vertical, confirm or exit

if confirmed go to makeMove

5.6 gameInteractive.h

Go to the documentation of this file.

```
#ifndef GAME_INTERACTIVE_H
#define GAME_INTERACTIVE_H

#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include "tile.h"
#include "printGraphic.h"
#include "userInputs.h"
#include "checks.h"

void makeMove(Tile selectedTiles,int numTiles,Tile **boardGame, int choice,Tile *playersTiles);
void tileOrientation(int choice,Tile selectedTiles,Tile *playersTiles,int numTiles,Tile **boardGame,
    int first);
void gameStartInteractive(Tile *playersTiles,int numTiles,Tile **boardGame);
#endif
```

5.7 C:/Users/letyc/Desktop/882713_Cantoni/include/gameMove.h File Reference

This file defines all the functions that are used to move and insert the tiles in the game.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
```

```
#include "tile.h"
#include "printGraphic.h"
#include "userInputs.h"
#include "checks.h"
#include "gameInteractive.h"
#include "gameAi.h"
```

Functions

- void **swapTiles** (**Tile** ∗tile1, **Tile** ∗tile2)
  Function to swap two tiles.
- void **removePlayerTile** (**Tile** ∗playersTiles, int numTiles, int choice)
  Removes a choosen tile from the player's tiles.
- **Tile flipTile** (**Tile** d)
  Flips a tile: flips the number of side1 with the number of side2.
- int **scoreTile** (**Tile** tile)
  Calculate the score of a tile which is the sum of side1+side2 of the tile.
- int **countScore** (**Tile** ∗∗boardGame, int numTiles)
  Counts the total score of the tiles on the game board.
- int **countTiles** (**Tile** ∗playersTiles, int numTiles)
  Counts the number of tiles the player has left to play.
- void **addTile** (**Tile** d, int row, int col, **Tile** ∗∗boardGame, int numTiles)
  Adds the chosen tile to the game board.
- void **gameStart** (**Tile** ∗playersTiles, int numTiles, **Tile** ∗∗boardGame)
  Initial function to choose the game mode and start the game.
- void **free_game** (**Tile** ∗∗boardGame, int numTiles)
  Frees the memory allocated for the game board.

5.7.1 Detailed Description

This file defines all the functions that are used to move and insert the tiles in the game.

Author

Cantoni Letizia 882713

5.7.2 Function Documentation

5.7.2.1 addTile()

```
void addTile (
             Tile d,
             int row,
             int col,
             Tile ** boardGame,
             int numTiles)
```

Adds the chosen tile to the game board.

Counts the number of tiles the player has left to play.

Parameters

d	The chosen tile.
row	The row to insert the tile.
col	The column to insert the tile.
boardGame	The game board.
numTiles	The number of tiles in the game.

Add the tile to the board

Handle special tiles

Check if the tile is adjacent to another tile left

Insert the tile mirroring the left tile

Check if the tile to insert is vertical and if the tile left below is free

Insert the tile below(vertical) copying the left tile up

Insert the tile below(vertical) copying the adjacent left tile

if the adjacent tile is vertical i need to flip the tile, otherwhise it will be: [2]{3} instead of [2]{2} [3]{2} [3]{3}

Check if the tile is adjacent to another tile right

Insert the tile mirroring the right tile

Check if the tile to insert is vertical and if the tile right below is free

Insert the tile below(vertical) copying the right tile up

Insert the tile below(vertical) copying the adjacent right tile

if the adjacent tile is vertical i need to flip the tile, otherwhise it will be: {3}[2] instead of [2]{2} {2}[3] [3]{3}

Special tile [11|11] Increment digits of all tiles by 1, except 6 which becomes 1

if the tile is not special and is vertical Add the tile also to the next row

5.7.2.2 countScore()

```
int countScore (
              Tile ** boardGame,
             int numTiles)
```

Counts the total score of the tiles on the game board.

Calculate the score of a tile which is the sum of side1+side2 of the tile

Parameters

boardGame	The game board.
numTiles	The number of tiles in the game.

Returns

int Returns the total score in the board.

5.7.2.3 countTiles()

```
int countTiles (
             Tile * playersTiles,
            int numTiles)
```

Counts the number of tiles the player has left to play.

Counts the total score of the tiles on the game board.

Parameters

playersTiles	The player's tiles.
numTiles	The number of tiles the player has.

Returns

int Returns the number of tiles left.

5.7.2.4 flipTile()

```
 Tile flipTile (
             Tile d)
```

Flips a tile: flips the number of side1 with the number of side2.

Removes a choosen tile from the player's tiles.

Parameters

d	The tile to flip.

Returns

Tile (p. 7) Returns the flipped tile.

5.7.2.5 free_game()

```
void free_game (
             Tile ** boardGame,
            int numTiles)
```

Frees the memory allocated for the game board.

Initial function to start the game.

Parameters

boardGame	The game board.
numTiles	The number of tiles in the game.

5.7.2.6 gameStart()

```
void gameStart (
            Tile * playersTiles,
            int numTiles,
            Tile ** boardGame)
```

Initial function to choose the game mode and start the game.

Adds the chosen tile to the game board.

Parameters

playersTiles	The player's tiles.
numTiles	The number of tiles in the game.
boardGame	The game board.

Loop to decide the game mode, it's a loop because that way if the user decides to read the rules then he can always go back and decide what mode to play.

Start of the game in single-player mode

Start of the game in AI mode

Printing of the Information-Rules

5.7.2.7 removePlayerTile()

```
void removePlayerTile (
            Tile * playersTiles,
            int numTiles,
            int choice)
```

Removes a choosen tile from the player's tiles.

Function to swap two tiles.

Parameters

playersTiles	The player's tiles.
numTiles	The number of tiles the player has.
choice	The index of the tile to remove.

5.7.2.8 scoreTile()

```
int scoreTile (
            Tile tile)
```

Calculate the score of a tile which is the sum of side1+side2 of the tile.

Flips a tile: flips the number of side1 with the number of side2.

Parameters

tile	The tile to calculate the score.

Returns

int Returns the score of the tile.

5.7.2.9 swapTiles()

```
void swapTiles (
              Tile * tile1,
              Tile * tile2)
```

Function to swap two tiles.

Parameters

tile1	The tile to swap.
tile2	The other tile to swap.

5.8 gameMove.h

Go to the documentation of this file.

```
00001
00006 #ifndef GAME_MOVE_H
00007 #define GAME_MOVE_H
00008
00009 #include <stdio.h>
00010 #include <stdlib.h>
00011 #include <time.h>
00012 #include "tile.h"
00013 #include "printGraphic.h"
00014 #include "userInputs.h"
00015 #include "checks.h"
00016 #include "gameInteractive.h"
00017 #include "gameAi.h"
00018
00019
00020 void swapTiles(Tile *tile1, Tile *tile2);
00021 void removePlayerTile(Tile *playersTiles,int numTiles,int choice);
00022 Tile flipTile(Tile d);
00023 int scoreTile(Tile tile);
00024 int countScore(Tile **boardGame, int numTiles);
00025 int countTiles(Tile *playersTiles, int numTiles);
00026 void addTile(Tile d, int row, int col,Tile **boardGame,int numTiles);
00027 void gameStart(Tile *playersTiles,int numTiles,Tile **boardGame);
00028 void free_game(Tile **boardGame, int numTiles);
00030 #endif
```

5.9 C:/Users/letyc/Desktop/882713_Cantoni/include/printGraphic.h File Reference

This file defines functions that are used to display the game implementation.

```
#include <stdio.h>
#include <stdlib.h>
#include "tile.h"
#include "gameMove.h"
```

Functions

- void **delay** (int seconds)

  Delays the execution for a specified number of seconds.
- void **printFirstScreen** ()

  Prints the first screen of the game. Just the title of the game.
- int **endScreen** ()

  Prints this screen when the game is over. and asks if the player wants to play again.
- void **show_rules** ()

  Displays the game rules.
- void **printTiles** (**Tile** d)

  Prints a tile. [side1|dide2].
- void **printSelectedTiles** (**Tile** d)

  Prints the tile selected by the player. in this function the vertical tile is printed differently from printTiles.
- void **printTilesPlayer** (**Tile** ∗playersTiles, int numTiles)

  Prints the player's tiles.
- void **printBoardGame** (**Tile** ∗∗boardGame, int numTiles)

  Prints the game board with all the tiles in the game.

5.9.1 Detailed Description

This file defines functions that are used to display the game implementation.

Author

Cantoni Letizia 882713

5.9.2 Function Documentation

5.9.2.1 delay()

```
void delay (
            int seconds)
```

Delays the execution for a specified number of seconds.

Parameters

seconds	The number of seconds to delay.

5.9.2.2 endScreen()

```
int endScreen ()
```

Prints this screen when the game is over. and asks if the player wants to play again.

Prints the first screen of the game. Just the title of the game

Returns

int Returns 1 if the player wants to play another game.

5.9.2.3 printBoardGame()

```
void printBoardGame (
              Tile ** boardGame,
             int numTiles)
```

Prints the game board with all the tiles in the game.

Prints the player's tiles.

Parameters

boardGame	The game board.
numTiles	The number of tiles in the game.

this is just for graphic

this is just for graphic

prints the number of the rows

if a cell of the boardgame is not occupied it print the equivalent of the tile but in spaces

else if the cell is ocuupied it prints the tile who is in it

this is just for graphic

here it prints the score= the sum of all the tile score in the game

this is just for graphic

5.9.2.4 printFirstScreen()

```
void printFirstScreen ()
```

Prints the first screen of the game. Just the title of the game.

Delays the execution for a specified number of seconds. Color green

Color blue

reset color

cursive

reset color

5.9.2.5 printSelectedTiles()

```
void printSelectedTiles (
              Tile d)
```

Prints the tile selected by the player. in this function the vertical tile is printed differently from printTiles.

Prints a tile. [side1|dide2]

Parameters

d	The selected tile.

5.9.2.6 printTiles()

```
void printTiles (
              Tile d)
```

Prints a tile. [side1|dide2].

Displays the game rules.

Parameters

d	The tile to print.

5.9.2.7 printTilesPlayer()

```
void printTilesPlayer (
              Tile * playersTiles,
             int numTiles)
```

Prints the player's tiles.

Prints a tile but in this function the vertical tile is printed differently from printTiles

Parameters

playersTiles	The player's tiles.
numTiles	The number of tiles the player has.

here if a tile is occupied(not already played) it is displayed

5.9.2.8 show_rules()

```
void show_rules ()
```

Displays the game rules.

Prints the end screen of the game and returns if the player wants to play again. wait for the player to press a button

clean input buffer

5.10 printGraphic.h

Go to the documentation of this file.

```

#ifndef PRINT_GRAPHIC_H
#define PRINT_GRAPHIC_H
#include <stdio.h>
#include <stdlib.h>
#include "tile.h"
#include "gameMove.h"

void delay(int seconds);
void printFirstScreen();
int endScreen();
void show_rules();
void printTiles(Tile d);
void printSelectedTiles(Tile d);
void printTilesPlayer(Tile *playersTiles, int numTiles);
void printBoardGame(Tile **boardGame, int numTiles);
#endif
```

5.11 C:/Users/letyc/Desktop/882713_Cantoni/include/tile.h File Reference

This file defines the game structures and the functions that initialize the player tiles and the boardgame.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
```

Data Structures

- struct **Tile**

Functions

- **Tile** ∗∗ **initializationBoard** (int numTiles)
  Initializes the game board.
- **Tile** ∗ **initializationPlayerTiles** (int numTiles, **Tile** ∗specialTiles)
  Initializes the player's tiles.
- void **initializationSpecialTiles** (**Tile** ∗specialTiles, int numSpecial)
  Initializes the special tiles.

5.11.1 Detailed Description

This file defines the game structures and the functions that initialize the player tiles and the boardgame.

Author

Cantoni Letizia 882713

5.11.2 Function Documentation

5.11.2.1 initializationBoard()

```
Tile ** initializationBoard (
            int numTiles)
```

Initializes the game board.

Parameters

numTiles	The number of tiles in the game. used to create the 'matrix' of boardgame

Returns

Tile∗∗ Returns the initialized board game.

if a vertical tile is added on the last row

Allocate memory for rows array

Allocate memory for each column in the current row

5.11.2.2 initializationPlayerTiles()

```
Tile * initializationPlayerTiles (
            int numTiles,
             Tile * specialTiles)
```

Initializes the player's tiles.

Initializes the game board.

Parameters

numTiles	The number of tiles the player has.
specialTiles	The array of special tiles in the game.

Returns

Tile∗ Returns the initialized player's Tiles.

Initialize the random number generator with the current time

Generates a random number between 0 and 9

90% chance of randVar being different from 1

Generates a random index between 0 and 2

Generate a casual number from 1 to 6

5.11.2.3 initializationSpecialTiles()

```
void initializationSpecialTiles (
             Tile * specialTiles,
            int numSpecial)
```

Initializes the special tiles.

Initializes the player's tiles.

Parameters

specialTiles	The special tiles to initialize. **Tile** (p. 7) ∗
numSpecial	The number of special tiles.

Returns

Tile∗ Returns the initialized special tiles.

5.12 tile.h

Go to the documentation of this file.

```

#ifndef TILE_H
#define TILE_H
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

typedef struct {
    int side1;
    int side2;
    int occupied;
    int vertical;
    int special;
} Tile;

Tile **initializationBoard(int numTiles);
Tile *initializationPlayerTiles(int numTiles,Tile *specialTiles);
void initializationSpecialTiles(Tile *specialTiles, int numSpecial);
#endif
```

5.13 C:/Users/letyc/Desktop/882713_Cantoni/include/userInputs.h File Reference

This file defines the functions that are used to get the user's input to make the moves in the game.

```
#include <stdio.h>
#include <stdlib.h>
```

Functions

- int **getTilesNumber** ()

    Asks the player with how many tiles he wants to play with.
- int **getMenuMode** ()

    Displays the game mode menu and returns the user's choice of game mode.(Singleplayer,AI or Game Rules)
- int **getMoveTiles** (int currenTiles)

    Asks the player and then returns the tile to play chosen by the player.
- char **getOrientation** ()

    Asks and returns the action that the player wants to do with the chosen tile. for example flip it, put it vertical, confirm or exit.
- int **getRow** (int numRows)

    Asks and returns the row where the player wants to insert the tile.
- char **getPosition** (int numTiles)

    Asks and returns the position where the player wants to insert the tile. (right or left)

5.13.1 Detailed Description

This file defines the functions that are used to get the user's input to make the moves in the game.

Author

Cantoni Letizia 882713

5.13.2 Function Documentation

5.13.2.1 getMenuMode()

```
int getMenuMode ()
```

Displays the game mode menu and returns the user's choice of game mode.(Singleplayer,AI or Game Rules)

Returns the player's choice of tiles number he wants to play with.

Returns

int return the user's choice of game mode.

clean the input buffer

Reset the choice

5.13.2.2 getMoveTiles()

```
int getMoveTiles (
             int currenTiles)
```

Asks the player and then returns the tile to play chosen by the player.

Returns the player's choice of game mode he want to play in Singleplayer,AI or view the Game Rules

Parameters

currenTiles	The current number of tiles.

Returns

int Returns the chosen tile.

clean the input buffer

5.13.2.3 getOrientation()

```
char getOrientation ()
```

Asks and returns the action that the player wants to do with the chosen tile. for example flip it, put it vertical, confirm or exit.

Returns the tile to play chosen by the player.

Returns

char Returns the chosen action.

clean the input buffer

5.13.2.4 getPosition()

```
char getPosition (
             int numTiles)
```

Asks and returns the position where the player wants to insert the tile. (right or left)

Returns the row where the player wants to insert the tile.

Parameters

numTiles	The number of tiles.

Returns

char Returns the chosen position.

clean the input buffer

5.13.2.5 getRow()

```
int getRow (
             int numRows)
```

Asks and returns the row where the player wants to insert the tile.

Returns the action that the player wants to do with the chosen tile. flip it, put it vertical,confirm or exit

Parameters

numRows	The number of rows.

Returns

int Returns the chosen row.

clean the input buffer

5.13.2.6 getTilesNumber()

```
int getTilesNumber ()
```

Asks the player with how many tiles he wants to play with.

Returns

int return the user's choice of tiles number he wants to play with.

5.14 userInputs.h

Go to the documentation of this file.

```

#ifndef USERINPUTS_H
#define USERINPUTS_H
#include <stdio.h>
#include <stdlib.h>

int getTilesNumber();
int getMenuMode();
int getMoveTiles(int currenTiles);
char getOrientation();
int getRow(int numRows);
char getPosition(int numTiles);
#endif
```

5.15 C:/Users/letyc/Desktop/882713_Cantoni/main.c File Reference

This project implements the Domino Game with it's own set of rules.

```
#include <stdio.h>
#include "include/tile.h"
#include "include/userInputs.h"
#include "include/printGraphic.h"
#include "include/gameMove.h"
#include "include/gameInteractive.h"
#include "include/gameAi.h"
#include "include/checks.h"
```

Functions

- int **main** ()

5.15.1 Detailed Description

This project implements the Domino Game with it's own set of rules.

Author

Cantoni Letizia 882713

Date

24/06/2024

5.15.2 Function Documentation

5.15.2.1 main()

```
int main ()
```

Initial function to start the game, in which the game structures are defined and initialized

Returns

int

print the first screen of the game, just for graphic

Wait 2 seconds

Initialization of special tiles

Initialization of player tiles

Board game initialization

Function to start the game

Print the 'game over' screen and return if the player wants to play again

Frees the memory allocated

5.16 C:/Users/letyc/Desktop/882713_Cantoni/README.md File Reference

5.17 C:/Users/letyc/Desktop/882713_Cantoni/src/checks.c File Reference

This file implements all the functions that are used to check the insertion of a tile into the game.

```
#include "../include/checks.h"
#include "../include/printGraphic.h"
#include "../include/gameMove.h"
```

Functions

- int **possibleRows** (**Tile** ∗∗boardGame, int numTiles)
- int **possibleLeftCell** (**Tile** ∗boardGame, int numTiles)

  Finds the possible left cell in which a tile can be add on the game board in a specific row.
- int **possibleRightCell** (**Tile** ∗boardGame, int numTiles)

  Finds the possible right cell in which a tile can be add on the game board in a specific row.
- int **isValid** (int row, int col, **Tile** selectedtile, char pos, **Tile** ∗∗boardGame, int numTiles)

  Checks if a move is valid.
- int **isValidFlat** (int row, int col, **Tile** selectedtile, char pos, **Tile** ∗∗boardGame, int numTiles)

  Checks if a flat move is valid (flat move = the tile to insert is not vertical).
- int **checkEndgame** (**Tile** ∗playersTiles, int numTiles, **Tile** ∗∗boardGame)

  Checks if there are any available moves or the game is finished. is true when the game is finished.

5.17.1 Detailed Description

This file implements all the functions that are used to check the insertion of a tile into the game.

Author

Cantoni Letizia 882713

5.17.2 Function Documentation

5.17.2.1 checkEndgame()

```
int checkEndgame (
             Tile * playersTiles,
             int numTiles,
             Tile ** boardGame)
```

Checks if there are any available moves or the game is finished. is true when the game is finished.

Parameters

playersTiles	The player's tiles.
numTiles	The number of tiles the player has.
boardGame	The game board.

Returns

int Returns 1 if there are no available moves, 0 otherwise.

Slides all the possible tile of the player

if a tile is occupied it means that it has not yet been played

for every 'possible' row it checks if a certain move is valid right or left

checks if the move is valid right or left

if it finds a possible move it exits

if no moves are possible

5.17.2.2 isValid()

```
int isValid (
             int row,
             int col,
             Tile selectedtile,
             char pos,
             Tile ** boardGame,
             int numTiles)
```

Checks if a move is valid.

Parameters

row	The row to check.
col	The column to check.
selectedtile	The selected tile.
pos	The position to check.(left 's' or right 'd')
boardGame	The game board.
numTiles	The number of tiles in the game.

Returns

int Returns 1 if the move is valid, 0 otherwise.

I check If where I want to put the tile is not occupied

and if the tile to insert is not vertical

if I want to put the tile vertical I have to look that the cell below is unoccupied

If i want to put the tile right

If the upper part of the tile can be placed

here I look to see if the tile I'm placing it next to is NOT vertical and it's compatible with my card

here I look to see if the tile I'm placing it next to is vertical and it's compatible with my card

If i want to put the tile left

5.17.2.3 isValidFlat()

```
int isValidFlat (
             int row,
             int col,
              Tile selectedtile,
             char pos,
              Tile ** boardGame,
             int numTiles)
```

Checks if a flat move is valid (flat move = the tile to insert is not vertical).

Parameters

row	The row to check.
col	The column to check.
selectedtile	The selected tile.
pos	The position to check.
boardGame	The game board.
numTiles	The number of tiles in the game.

Returns

int Returns 1 if the move is valid, 0 otherwise.

If is special i can place it everywhere

If i am trying to put the tile between two other tiles

all the two tile must be compatible with the one i want to insert, in this case i am trying to insert the tile vertically

all the two tile must be compatible with the one i want to insert, in this case i am trying to insert the tile horizontally

If i am trying to put the tile right

If the adjacent right tile is vertical and compatible with mine

If the adjacent right tile is NOT vertical and compatible with mine

If i am trying to put the tile left

If the adjacent left tile is compatible with mine

5.17.2.4 possibleLeftCell()

```
int possibleLeftCell (
              Tile * boardGame,
             int numTiles)
```

Finds the possible left cell in which a tile can be add on the game board in a specific row.

Parameters

boardGame	The game board.
numTiles	The number of tiles in the game.

Returns

int Returns the index of the possible left cell, or -1 if not found.

5.17.2.5 possibleRightCell()

```
int possibleRightCell (
              Tile * boardGame,
             int numTiles)
```

Finds the possible right cell in which a tile can be add on the game board in a specific row.

Parameters

boardGame	The game board.
numTiles	The number of tiles in the game.

Returns

int Returns the index of the possible right cell, or -1 if not found.

5.17.2.6 possibleRows()

```
int possibleRows (
              Tile ** boardGame,
             int numTiles)
```

Counts the possible rows in which a tile can be added on the game board. That is, look at which rows are occupied by at least one tile

Parameters

boardGame	The game board.
numTiles	The number of tiles in the game.

Returns

int Returns the number of possible rows.

Move to the next row as soon as we find an occupied cell

5.18 C:/Users/letyc/Desktop/882713_Cantoni/src/gameAi.c File Reference

This file implements all the functions that are used to play the game in AI mode.

```
#include "../include/gameAi.h"
#include "../include/checks.h"
#include "../include/gameMove.h"
#include "../include/printGraphic.h"
```

Functions

- void **sortTiles** (**Tile** ∗tiles, int numTiles)
  Function to sort the player tiles based on their score (es.[3|2] has a score=5).
- int **MoveAi** (int choice, **Tile** selectedTiles, **Tile** ∗playersTiles, int numTiles, **Tile** ∗∗boardGame)
  Function to make a move for the AI.
- void **gameStartAi** (**Tile** ∗playersTiles, int numTiles, **Tile** ∗∗boardGame)
  Initial function for the AI to choose a tile.

5.18.1 Detailed Description

This file implements all the functions that are used to play the game in AI mode.

Author

Cantoni Letizia 882713

5.18.2 Function Documentation

5.18.2.1 gameStartAi()

```
void gameStartAi (
             Tile * playersTiles,
            int numTiles,
             Tile ** boardGame)
```

Initial function for the AI to choose a tile.

Parameters

playersTiles	The array of tiles available to the AI.
numTiles	The number of tiles available to the AI.
boardGame	The current state of the game board.

it sorts the tiles so the first match you find is the one with the highest score

if the tile is not already played

if the tile was not inserted i carry on

else i wait just for the user to see the move and then i start checking the tiles from the start

if the tile has already been played i go to the next

if there are no avaiable moves exit

if all the tiles have been played print the phrase below else print the remaining tiles

5.18.2.2 MoveAi()

```
int MoveAi (
             int choice,
              Tile selectedTiles,
              Tile * playersTiles,
             int numTiles,
              Tile ** boardGame)
```

Function to make a move for the AI.

Parameters

choice	The index of the tile chosen by the AI.
selectedTiles	The tile chosen by the AI.
playersTiles	The array of tiles available to the AI.
numTiles	The number of tiles available to the AI.
boardGame	The current state of the game board.

Returns

int Returns 1 if the move was successful, 0 otherwise.

Initialize the random number generator with the current time

Generates a random number between 0 and 4 that give us a probability of 80% that is != 0

Possible row where to insert the tile

I will try for every row if there is a match

Possible right column where to insert the tile

Possible left column where to insert the tile

If there is a possible right cell

If the randon number is !=0 the vertical is true, so we try to place it vertical

If the moveis not valid with the vertical tile, we try with the horizonal tile

if the move is valid we add it

else we try with the flipped tile and if the move is valid we add it

If there is a possible left cell

If the randon number is !=0 the vertical is true, so we try to place it vertical

If the move is not valid with the vertical tile, we try with the horizonal tile

if the move is valid we add it

else we try with the flipped tile and if the move is valid we add it

If there is a possible right AND left cell

If the move is not valid on either left or right with the vertical tile, we try with the horizonal tile

if the move is valid left we add it

else if the move is valid with the flipped tile left we add it

if the move is valid right we add it

else if the move is valid with the flipped tile right we add it

5.18.2.3 sortTiles()

```
void sortTiles (
              Tile * tiles,
             int numTiles)
```

Function to sort the player tiles based on their score (es.[3|2] has a score=5).

Parameters

tiles	The array of tiles to sort(the player tiles).
numTiles	The number of tiles available to the AI.

Move special tiles to the second position so it's likely it mirrores the tile with the highest score

5.19 C:/Users/letyc/Desktop/882713_Cantoni/src/gameInteractive.c File Reference

This file implements the functions used to play the game in the Interactive Mode.

```
#include "../include/gameInteractive.h"
#include "../include/tile.h"
#include "../include/gameMove.h"
#include "../include/printGraphic.h"
#include "../include/userInputs.h"
#include "../include/checks.h"
```

Functions

- void **makeMove** (**Tile** selectedTiles, int numTiles, **Tile** ∗∗boardGame, int choice, **Tile** ∗playersTiles)

  Function to get the input from the player to where insert the tile.
- void **tileOrientation** (int choice, **Tile** selectedTiles, **Tile** ∗playersTiles, int numTiles, **Tile** ∗∗boardGame, int first)

  Given a tile, take the input of the player to whether to flip it, put it vertical,confirm or exit.
- void **gameStartInteractive** (**Tile** ∗playersTiles, int numTiles, **Tile** ∗∗boardGame)

  Initial function to choose the tile (interactive mode).

5.19.1 Detailed Description

This file implements the functions used to play the game in the Interactive Mode.

Author

Cantoni Letizia 882713

5.19.2 Function Documentation

5.19.2.1 gameStartInteractive()

```
void gameStartInteractive (
              Tile ∗ playersTiles,
             int numTiles,
              Tile ∗∗ boardGame)
```

Initial function to choose the tile (interactive mode).

Parameters

playersTiles	The player's tiles.
numTiles	The number of tiles in the game.
boardGame	The game board.

Get the tile the player wants to play

This happens only for the first move

The special tiles [11|11] and [12|21] cannot be placed in the first position

Get the tile the player wants to play

If all the tiles have been inserted print

else print the remaining tiles

5.19.2.2 makeMove()

```
void makeMove (
             Tile selectedTiles,
             int numTiles,
             Tile ** boardGame,
             int choice,
             Tile * playersTiles)
```

Function to get the input from the player to where insert the tile.

Parameters

selectedTiles	The tile selected ti insert.
numTiles	The number of tiles in the game.
boardGame	The game board.
choice	The chosen tile.
playersTiles	The player's tiles.

the number of the row a tile can be placed

if the number of rows>1 asks the player which row he wants to put it in

Ask the player where he wants to put the tile, left or right

retrieve the right column where to put the tile

retrieve the left column where to put the tile

if there is a column for the move

if the move is valid

makes the move

5.19.2.3 tileOrientation()

```
void tileOrientation (
             int choice,
             Tile selectedTiles,
             Tile * playersTiles,
             int numTiles,
             Tile ** boardGame,
             int first)
```

Given a tile, take the input of the player to whether to flip it, put it vertical,confirm or exit.

Parameters

choice	The chosen tile.
selectedTiles	The selected tile.
playersTiles	The player's tiles.
numTiles	The number of tiles in the game.
boardGame	The game board.
first	Whether it's the first move.

choose what to do with the tile: flip it, put it vertical, confirm or exit

if confirmed go to makeMove

5.20 C:/Users/letyc/Desktop/882713_Cantoni/src/gameMove.c File Reference

This file implements the functions used to move and insert the tiles in the game.

```
#include "../include/gameMove.h"
#include "../include/tile.h"
#include "../include/printGraphic.h"
#include "../include/userInputs.h"
#include "../include/checks.h"
```

Functions

- void **swapTiles** (**Tile** ∗tile1, **Tile** ∗tile2)
  Function to swap two tiles.
- void **removePlayerTile** (**Tile** ∗playersTiles, int numTiles, int choice)
  Removes a choosen tile from the player's tiles.
- **Tile** **flipTile** (**Tile** d)
  Flips a tile: flips the number of side1 with the number of side2.
- int **scoreTile** (**Tile** tile)
  Calculate the score of a tile which is the sum of side1+side2 of the tile.
- int **countScore** (**Tile** ∗∗boardGame, int numTiles)
  Counts the total score of the tiles on the game board.
- int **countTiles** (**Tile** ∗playersTiles, int numTiles)
  Counts the number of tiles the player has left to play.
- void **addTile** (**Tile** d, int row, int col, **Tile** ∗∗boardGame, int numTiles)
  Adds the chosen tile to the game board.
- void **gameStart** (**Tile** ∗playersTiles, int numTiles, **Tile** ∗∗boardGame)
  Initial function to choose the game mode and start the game.
- void **free_game** (**Tile** ∗∗boardGame, int numTiles)
  Frees the memory allocated for the game board.

5.20.1 Detailed Description

This file implements the functions used to move and insert the tiles in the game.

Author

Cantoni Letizia 882713

5.20.2 Function Documentation

5.20.2.1 addTile()

```
void addTile (
             Tile d,
            int row,
            int col,
             Tile ** boardGame,
            int numTiles)
```

Adds the chosen tile to the game board.

Parameters

d	The chosen tile.
row	The row to insert the tile.
col	The column to insert the tile.
boardGame	The game board.
numTiles	The number of tiles in the game.

Add the tile to the board

Handle special tiles

Check if the tile is adjacent to another tile left

Insert the tile mirroring the left tile

Check if the tile to insert is vertical and if the tile left below is free

Insert the tile below(vertical) copying the left tile up

Insert the tile below(vertical) copying the adjacent left tile

if the adjacent tile is vertical i need to flip the tile, otherwhise it will be: [2]{3} instead of [2]{2} [3]{2} [3]{3}

Check if the tile is adjacent to another tile right

Insert the tile mirroring the right tile

Check if the tile to insert is vertical and if the tile right below is free

Insert the tile below(vertical) copying the right tile up

Insert the tile below(vertical) copying the adjacent right tile

if the adjacent tile is vertical i need to flip the tile, otherwhise it will be: {3}[2] instead of [2]{2} {2}[3] [3]{3}

Special tile [11|11] Increment digits of all tiles by 1, except 6 which becomes 1

if the tile is not special and is vertical Add the tile also to the next row

5.20.2.2 countScore()

```
int countScore (
              Tile ** boardGame,
             int numTiles)
```

Counts the total score of the tiles on the game board.

Parameters

boardGame	The game board.
numTiles	The number of tiles in the game.

Returns

int Returns the total score in the board.

5.20.2.3 countTiles()

```
int countTiles (
              Tile * playersTiles,
             int numTiles)
```

Counts the number of tiles the player has left to play.

Parameters

playersTiles	The player's tiles.
numTiles	The number of tiles the player has.

Returns

int Returns the number of tiles left.

5.20.2.4 flipTile()

```
Tile flipTile (
            Tile d)
```

Flips a tile: flips the number of side1 with the number of side2.

Parameters

d	The tile to flip.

Returns

Tile (p. 7) Returns the flipped tile.

5.20.2.5 free_game()

```
void free_game (
            Tile ** boardGame,
            int numTiles)
```

Frees the memory allocated for the game board.

Parameters

boardGame	The game board.
numTiles	The number of tiles in the game.

5.20.2.6 gameStart()

```
void gameStart (
            Tile * playersTiles,
            int numTiles,
            Tile ** boardGame)
```

Initial function to choose the game mode and start the game.

Parameters

playersTiles	The player's tiles.
numTiles	The number of tiles in the game.
boardGame	The game board.

Loop to decide the game mode, it's a loop because that way if the user decides to read the rules then he can always go back and decide what mode to play.

Start of the game in single-player mode

Start of the game in AI mode

Printing of the Information-Rules

5.20.2.7 removePlayerTile()

```
void removePlayerTile (
              Tile * playersTiles,
             int numTiles,
             int choice)
```

Removes a choosen tile from the player's tiles.

Parameters

playersTiles	The player's tiles.
numTiles	The number of tiles the player has.
choice	The index of the tile to remove.

5.20.2.8 scoreTile()

```
int scoreTile (
              Tile tile)
```

Calculate the score of a tile which is the sum of side1+side2 of the tile.

Parameters

tile	The tile to calculate the score.

Returns

int Returns the score of the tile.

5.20.2.9 swapTiles()

```
void swapTiles (
              Tile * tile1,
              Tile * tile2)
```

Function to swap two tiles.

Parameters

tile1	The tile to swap.
tile2	The other tile to swap.

5.21 C:/Users/letyc/Desktop/882713_Cantoni/src/printGraphic.c File Reference

This file implements all the functions that are used to display the game implementation.

```
#include "../include/printGraphic.h"
#include "../include/gameMove.h"
```

Functions

- void **delay** (int seconds)

    Delays the execution for a specified number of seconds.
- void **printFirstScreen** ()

    Prints the first screen of the game. Just the title of the game.
- int **endScreen** ()

    Prints this screen when the game is over. and asks if the player wants to play again.
- void **show_rules** ()

    Displays the game rules.
- void **printTiles** (**Tile** d)

    Prints a tile. [side1|dide2].
- void **printSelectedTiles** (**Tile** d)

    Prints the tile selected by the player. in this function the vertical tile is printed differently from printTiles.
- void **printTilesPlayer** (**Tile** ∗playersTiles, int numTiles)

    Prints the player's tiles.
- void **printBoardGame** (**Tile** ∗∗boardGame, int numTiles)

    Prints the game board with all the tiles in the game.

5.21.1 Detailed Description

This file implements all the functions that are used to display the game implementation.

Author

Cantoni Letizia 882713

5.21.2 Function Documentation

5.21.2.1 delay()

```
void delay (
            int seconds)
```

Delays the execution for a specified number of seconds.

Parameters

seconds	The number of seconds to delay.

5.21.2.2 endScreen()

```
int endScreen ()
```

Prints this screen when the game is over. and asks if the player wants to play again.

Returns

int Returns 1 if the player wants to play another game.

5.21.2.3 printBoardGame()

```
void printBoardGame (
             Tile ** boardGame,
             int numTiles)
```

Prints the game board with all the tiles in the game.

Parameters

boardGame	The game board.
numTiles	The number of tiles in the game.

this is just for graphic

this is just for graphic

prints the number of the rows

if a cell of the boardgame is not occupied it print the equivalent of the tile but in spaces

else if the cell is ocuupied it prints the tile who is in it

this is just for graphic

here it prints the score= the sum of all the tile score in the game

this is just for graphic

5.21.2.4 printFirstScreen()

```
void printFirstScreen ()
```

Prints the first screen of the game. Just the title of the game.

Delays the execution for a specified number of seconds. Color green

Color blue

reset color

cursive

reset color

5.21.2.5 printSelectedTiles()

```
void printSelectedTiles (
             Tile d)
```

Prints the tile selected by the player. in this function the vertical tile is printed differently from printTiles.

Parameters

d	The selected tile.

5.21.2.6 printTiles()

```
void printTiles (
              Tile d)
```

Prints a tile. [side1|dide2].

Parameters

d	The tile to print.

5.21.2.7 printTilesPlayer()

```
void printTilesPlayer (
              Tile * playersTiles,
             int numTiles)
```

Prints the player's tiles.

Parameters

playersTiles	The player's tiles.
numTiles	The number of tiles the player has.

here if a tile is occupied(not already played) it is displayed

5.21.2.8 show_rules()

```
void show_rules ()
```

Displays the game rules.

Prints the end screen of the game and returns if the player wants to play again. wait for the player to press a button

clean input buffer

5.22 C:/Users/letyc/Desktop/882713_Cantoni/src/tile.c File Reference

This file implements all the functions that initialize the player tiles (with the special tiles) and the boardgame.

```
#include "../include/tile.h"
```

Macros

- #define **SPECIAL_TILE_1_SIDE1** 11
- #define **SPECIAL_TILE_1_SIDE2** 11
- #define **SPECIAL_TILE_2_SIDE1** 12
- #define **SPECIAL_TILE_2_SIDE2** 21

Functions

- **Tile** ∗∗ **initializationBoard** (int numTiles)
  Initializes the game board.
- **Tile** ∗ **initializationPlayerTiles** (int numTiles, **Tile** ∗specialTiles)
  Initializes the player's tiles.
- void **initializationSpecialTiles** (**Tile** ∗specialTiles, int numSpecial)
  Initializes the special tiles.

5.22.1 Detailed Description

This file implements all the functions that initialize the player tiles (with the special tiles) and the boardgame.

Author

Cantoni Letizia 882713

5.22.2 Macro Definition Documentation

5.22.2.1 SPECIAL_TILE_1_SIDE1

```
#define SPECIAL_TILE_1_SIDE1 11
```

5.22.2.2 SPECIAL_TILE_1_SIDE2

```
#define SPECIAL_TILE_1_SIDE2 11
```

5.22.2.3 SPECIAL_TILE_2_SIDE1

```
#define SPECIAL_TILE_2_SIDE1 12
```

5.22.2.4 SPECIAL_TILE_2_SIDE2

```
#define SPECIAL_TILE_2_SIDE2 21
```

5.22.3 Function Documentation

5.22.3.1 initializationBoard()

```
Tile ** initializationBoard (
            int numTiles)
```

Initializes the game board.

Parameters

numTiles	The number of tiles in the game. used to create the 'matrix' of boardgame

Returns

Tile∗∗ Returns the initialized board game.

if a vertical tile is added on the last row

Allocate memory for rows array

Allocate memory for each column in the current row

5.22.3.2 initializationPlayerTiles()

```
 Tile * initializationPlayerTiles (
            int numTiles,
             Tile * specialTiles)
```

Initializes the player's tiles.

Parameters

numTiles	The number of tiles the player has.
specialTiles	The array of special tiles in the game.

Returns

Tile∗ Returns the initialized player's Tiles.

Initialize the random number generator with the current time

Generates a random number between 0 and 9

90% chance of randVar being different from 1

Generates a random index between 0 and 2

Generate a casual number from 1 to 6

5.22.3.3 initializationSpecialTiles()

```
void initializationSpecialTiles (
             Tile * specialTiles,
            int numSpecial)
```

Initializes the special tiles.

Parameters

specialTiles	The special tiles to initialize. **Tile** (p. 7) ∗
numSpecial	The number of special tiles.

Returns

Tile∗ Returns the initialized special tiles.

5.23 C:/Users/letyc/Desktop/882713_Cantoni/src/userInputs.c File Reference

This file implements all the functions that are used to get the user's input to make the moves in the game.

```
#include "../include/userInputs.h"
```

Functions

- int **getTilesNumber** ()

  Asks the player with how many tiles he wants to play with.
- int **getMenuMode** ()

  Displays the game mode menu and returns the user's choice of game mode.(Singleplayer,AI or Game Rules)
- int **getMoveTiles** (int currenTiles)

  Asks the player and then returns the tile to play chosen by the player.
- char **getOrientation** ()

  Asks and returns the action that the player wants to do with the chosen tile. for example flip it, put it vertical, confirm or exit.
- int **getRow** (int numRows)

  Asks and returns the row where the player wants to insert the tile.
- char **getPosition** (int numTiles)

  Asks and returns the position where the player wants to insert the tile. (right or left)

5.23.1 Detailed Description

This file implements all the functions that are used to get the user's input to make the moves in the game.

Author

Cantoni Letizia 882713

5.23.2 Function Documentation

5.23.2.1 getMenuMode()

```
int getMenuMode ()
```

Displays the game mode menu and returns the user's choice of game mode.(Singleplayer,AI or Game Rules)

Returns

int return the user's choice of game mode.

clean the input buffer

Reset the choice

5.23.2.2 getMoveTiles()

```
int getMoveTiles (
            int currenTiles)
```

Asks the player and then returns the tile to play chosen by the player.

Parameters

currenTiles	The current number of tiles.

Returns

int Returns the chosen tile.

clean the input buffer

5.23.2.3 getOrientation()

```
char getOrientation ()
```

Asks and returns the action that the player wants to do with the chosen tile. for example flip it, put it vertical, confirm or exit.

Returns

char Returns the chosen action.

clean the input buffer

5.23.2.4 getPosition()

```
char getPosition (
            int numTiles)
```

Asks and returns the position where the player wants to insert the tile. (right or left)

Parameters

numTiles	The number of tiles.

Returns

char Returns the chosen position.

clean the input buffer

5.23.2.5 getRow()

```
int getRow (
          int numRows)
```

Asks and returns the row where the player wants to insert the tile.

Parameters

numRows	The number of rows.

Returns

int Returns the chosen row.

clean the input buffer

5.23.2.6 getTilesNumber()

```
int getTilesNumber ()
```

Asks the player with how many tiles he wants to play with.

Returns

int return the user's choice of tiles number he wants to play with.

Index