15/05/2022

Cassiopée GHIZELLAOUI

Paul ZANOLIN

Documentation Technique

Projet Takuzu



Table des matières

[autosolve.h 3](#_Toc103596539)

[int giveHint(SIZEDGRID usergrid, int \* x, int \* y, int \* val); 3](#_Toc103596540)

[int checkIfUnderIsTheSame(SIZEDGRID usergrid, int x, int y); 3](#_Toc103596541)

[int checkIfRightIsTheSame(SIZEDGRID usergrid, int x, int y); 3](#_Toc103596542)

[int placeHintUnder(SIZEDGRID usergrid, int x, int y, int \* outX, int \* outY, int \* val); 3](#_Toc103596543)

[int placeHintRight(SIZEDGRID usergrid, int x, int y, int \* outX, int \* outY, int \* val); 3](#_Toc103596544)

[int checkIfSpaceBetweenTwoSameUnder(SIZEDGRID usergrid, int x, int y); 3](#_Toc103596545)

[int checkIfSpaceBetweenTwoSameRight(SIZEDGRID usergrid, int x, int y); 3](#_Toc103596546)

[void placeHintInSpace(int x, int y, int baseVal, int \*outX, int \*outY, int \*val); 4](#_Toc103596547)

[int checkIfLineHave2Empty(SIZEDGRID usergrid, int lineNum); 4](#_Toc103596548)

[int checkIfColumnHave2Empty(SIZEDGRID usergrid, int columnNum); 4](#_Toc103596549)

[int placeHintIfSameLine(SIZEDGRID usergrid, int lineNum, int \* x, int \* y, int \* val); 4](#_Toc103596550)

[int placeHintIfSameColumn(SIZEDGRID usergrid, int columnNum, int \* x, int \* y, int \* val); 4](#_Toc103596551)

[int isLineFull(SIZEDGRID usergrid, int lineNum); 4](#_Toc103596552)

[int isColumnFull(SIZEDGRID usergrid, int columnNum); 4](#_Toc103596553)

[int countSymbolInLine(SIZEDGRID usergrid, int lineNum, int \* zero, int \* one, int \* minusOne); 4](#_Toc103596554)

[int countSymbolInColumn(SIZEDGRID usergrid, int columnNum, int \* zero, int \* one, int \* minusOne); 5](#_Toc103596555)

[int placeHintFillLine(SIZEDGRID usergrid, int lineNum, int \* x, int \* y); 5](#_Toc103596556)

[int placeHintFillColumn(SIZEDGRID usergrid, int columnNum, int \* x, int \* y); 5](#_Toc103596557)

[MOVE \* allocMove(); 5](#_Toc103596558)

[void freeMove(MOVE \* moveToFree); 5](#_Toc103596559)

[MOVE \* newMoveWithValues(int x, int y, int hypothesis, int changedOnce, MOVE \* previous); 5](#_Toc103596560)

[void rollbackGridToHypothesis(SIZEDGRID \* grid, MOVE \*\* moveList); 5](#_Toc103596561)

[int getNextCaseToDo(SIZEDGRID usergrid, int rank, int \* x, int \* y, int \* val); 5](#_Toc103596562)

[int recursiveSolve(SIZEDGRID \* usergrid, MOVE \*\* moveList, int printSteps); 6](#_Toc103596563)

[grid.h 6](#_Toc103596564)

[SIZEDGRID genMask(int size); 6](#_Toc103596565)

[SIZEDGRID allocGrid(int size); 6](#_Toc103596566)

[void freeGrid(SIZEDGRID \* grid); 6](#_Toc103596567)

[int\*\* convertToTakuzu4(int tab4[4][4]); 6](#_Toc103596568)

[int\*\* convertToTakuzu8(int tab8[8][8]); 6](#_Toc103596569)

[int\*\* convertToTakuzu16(int tab8[16][16]); 6](#_Toc103596570)

[SIZEDGRID getGrid4(); 7](#_Toc103596571)

[SIZEDGRID getMask4(); // debug 7](#_Toc103596572)

[SIZEDGRID getGrid8(); 7](#_Toc103596573)

[SIZEDGRID getGrid16(); 7](#_Toc103596574)

[void fillWithInt(SIZEDGRID \* gridToFill, int valueToFill); 7](#_Toc103596575)

[void addOneOnTwoUsingMaks(SIZEDGRID gridOne, SIZEDGRID \* gridTwo, SIZEDGRID mask); 7](#_Toc103596576)

[int checkZeroEqualOne(SIZEDGRID usergrid, int x, int y, int val); 7](#_Toc103596577)

[int checkMax2Following(SIZEDGRID usergrid, int x, int y, int val); 7](#_Toc103596578)

[int checkSimilarLinesOrColumns(SIZEDGRID usergrid, int x, int y, int val); 8](#_Toc103596579)

[int isNewValValid(SIZEDGRID usergrid, int x, int y, int val, int showErr); 8](#_Toc103596580)

[int isGridValid(SIZEDGRID usergrid); 8](#_Toc103596581)

[int checkEnded(SIZEDGRID usergrid); 8](#_Toc103596582)

[int countEmpty(SIZEDGRID usergrid); 8](#_Toc103596583)

[interaction.h 8](#_Toc103596584)

[void printGrid(SIZEDGRID grid, SIZEDGRID mask, int ignoreMask); 8](#_Toc103596585)

[void pickCoords(int\* x, int\* y, int size); 8](#_Toc103596586)

[int getSize(); 9](#_Toc103596587)

[int getValue(int x, int y); 9](#_Toc103596588)

[SIZEDGRID getMask(int size); 9](#_Toc103596589)

[void play(); 9](#_Toc103596590)

[void autoSolveInterface(); 9](#_Toc103596591)

[void genGridInterface(); 9](#_Toc103596592)

[int mainMenu(); 9](#_Toc103596593)

# autosolve.h

## int giveHint(SIZEDGRID usergrid, int \* x, int \* y, int \* val);

Desc

## int checkIfUnderIsTheSame(SIZEDGRID usergrid, int x, int y);

Desc

## int checkIfRightIsTheSame(SIZEDGRID usergrid, int x, int y);

Desc

## int placeHintUnder(SIZEDGRID usergrid, int x, int y, int \* outX, int \* outY, int \* val);

Desc

## int placeHintRight(SIZEDGRID usergrid, int x, int y, int \* outX, int \* outY, int \* val);

Desc

## int checkIfSpaceBetweenTwoSameUnder(SIZEDGRID usergrid, int x, int y);

Desc

## int checkIfSpaceBetweenTwoSameRight(SIZEDGRID usergrid, int x, int y);

Desc

## void placeHintInSpace(int x, int y, int baseVal, int \*outX, int \*outY, int \*val);

Desc

## int checkIfLineHave2Empty(SIZEDGRID usergrid, int lineNum);

Desc

## int checkIfColumnHave2Empty(SIZEDGRID usergrid, int columnNum);

Desc

## int placeHintIfSameLine(SIZEDGRID usergrid, int lineNum, int \* x, int \* y, int \* val);

Desc

## int placeHintIfSameColumn(SIZEDGRID usergrid, int columnNum, int \* x, int \* y, int \* val);

Desc

## int isLineFull(SIZEDGRID usergrid, int lineNum);

Desc

## int isColumnFull(SIZEDGRID usergrid, int columnNum);

Desc

## int countSymbolInLine(SIZEDGRID usergrid, int lineNum, int \* zero, int \* one, int \* minusOne);

Desc

## int countSymbolInColumn(SIZEDGRID usergrid, int columnNum, int \* zero, int \* one, int \* minusOne);

Desc

## int placeHintFillLine(SIZEDGRID usergrid, int lineNum, int \* x, int \* y);

Desc

## int placeHintFillColumn(SIZEDGRID usergrid, int columnNum, int \* x, int \* y);

Desc

## MOVE \* allocMove();

Desc

## void freeMove(MOVE \* moveToFree);

Desc

## MOVE \* newMoveWithValues(int x, int y, int hypothesis, int changedOnce, MOVE \* previous);

Desc

## void rollbackGridToHypothesis(SIZEDGRID \* grid, MOVE \*\* moveList);

Desc

## int getNextCaseToDo(SIZEDGRID usergrid, int rank, int \* x, int \* y, int \* val);

Desc

## int recursiveSolve(SIZEDGRID \* usergrid, MOVE \*\* moveList, int printSteps);

Desc

# grid.h

## SIZEDGRID genMask(int size);

Desc

## SIZEDGRID allocGrid(int size);

Desc

## void freeGrid(SIZEDGRID \* grid);

Desc

## int\*\* convertToTakuzu4(int tab4[4][4]);

Desc

## int\*\* convertToTakuzu8(int tab8[8][8]);

Desc

## int\*\* convertToTakuzu16(int tab8[16][16]);

Desc

## SIZEDGRID getGrid4();

Desc

## SIZEDGRID getMask4(); // debug

Desc

## SIZEDGRID getGrid8();

Desc

## SIZEDGRID getGrid16();

Desc

## void fillWithInt(SIZEDGRID \* gridToFill, int valueToFill);

Desc

## void addOneOnTwoUsingMaks(SIZEDGRID gridOne, SIZEDGRID \* gridTwo, SIZEDGRID mask);

Desc

## int checkZeroEqualOne(SIZEDGRID usergrid, int x, int y, int val);

Desc

## int checkMax2Following(SIZEDGRID usergrid, int x, int y, int val);

Desc

## int checkSimilarLinesOrColumns(SIZEDGRID usergrid, int x, int y, int val);

Desc

## int isNewValValid(SIZEDGRID usergrid, int x, int y, int val, int showErr);

Desc

## int isGridValid(SIZEDGRID usergrid);

Desc

## int checkEnded(SIZEDGRID usergrid);

Desc

## int countEmpty(SIZEDGRID usergrid);

Desc

# interaction.h

## void printGrid(SIZEDGRID grid, SIZEDGRID mask, int ignoreMask);

Desc

## void pickCoords(int\* x, int\* y, int size);

Desc

## int getSize();

Desc

## int getValue(int x, int y);

Desc

## SIZEDGRID getMask(int size);

Desc

## void play();

Desc

## void autoSolveInterface();

Desc

## void genGridInterface();

Desc

## int mainMenu();

Desc