三目並べ詳細仕様書

Iチームメンバー

杉友泰宣　藤井健

目次

[1. 概要 4](#_Toc167955586)

[2. 画面遷移図 4](#_Toc167955587)

[3. システム機能フロー図 5](#_Toc167955588)

[3.1 フロー説明 5](#_Toc167955589)

[3.1.1 [盤面を初期化] 5](#_Toc167955590)

[3.1.2 [3×3のマスと先後を画面出力] 5](#_Toc167955591)

[3.1.3 [縦横のマス数を入力] 6](#_Toc167955592)

[3.1.4 [入力した値は規定値内か] 6](#_Toc167955593)

[3.1.5 [既に入力されていないか] 6](#_Toc167955594)

[3.1.6 [三目並べの画面のマスに「〇」「×」を出力] 6](#_Toc167955595)

[3.1.7 [一直線がすべて「〇」、または「×」ですか] 7](#_Toc167955596)

[3.1.8 [出力したのは9回目ですか] 7](#_Toc167955597)

[3.1.9 [結果を出力] 7](#_Toc167955598)

[3.1.10 [もう一度遊びますか] 7](#_Toc167955599)

[4. ファイル構成図 9](#_Toc167955600)

[5. 機能構成 9](#_Toc167955601)

[5.1 tic\_tac\_toe.c 9](#_Toc167955602)

[三目並べを行う前の処理 10](#_Toc167955603)

[三目並べを行う際の処理 10](#_Toc167955604)

[5.2 game\_ready.c 11](#_Toc167955605)

[InitSquare()関数 11](#_Toc167955606)

[PreScreen()関数 11](#_Toc167955607)

[5.3 get\_input.c 11](#_Toc167955608)

[SquareNumber()関数 11](#_Toc167955609)

[NumRange()関数 12](#_Toc167955610)

[MarkChecker()関数 12](#_Toc167955611)

[5.4 get\_output.c 12](#_Toc167955612)

[PrintMark()関数 12](#_Toc167955613)

[Bingo()関数 13](#_Toc167955614)

[SquareFull()関数 13](#_Toc167955615)

[5.5 get\_result.c 14](#_Toc167955616)

[GameResult()関数 14](#_Toc167955617)

[PlayAgain()関数 14](#_Toc167955618)

[5.6 define.h 14](#_Toc167955619)

[真か偽を表す列挙型 14](#_Toc167955620)

[三目並べの続行を表す列挙型 14](#_Toc167955621)

[勝利を表す列挙型 15](#_Toc167955622)

[マスが埋まっているかを表すためのマクロ定義 15](#_Toc167955623)

# 概要

本詳細仕様書は、二人のプレイヤーが交互に縦横それぞれ1～3の数字を入力し、マスに「o」または「x」を出力する三目並べの詳細を示すものである。

# 画面遷移図

ダイアグラム

自動的に生成された説明

# 画面構成

### ・数字と記号は全て半角で表示する。

### ・記号は’o’（オー）と‘x’（エックス）で表示する。

### ・マスは縦区切りの’|’だけでマスの表示をする。

ダイアグラム, 図形

自動的に生成された説明

### ・マスの上に半角スペース４つの後ろに’o’を表示し、’o’の後ろに半角スペースを２つ入れ、先攻と表示する。またその次の行に’o’と先攻と同じように’x’と後攻を表示する。

ダイアグラム

自動的に生成された説明

### ・盤面は１マス半角スペース二つで表示する。

### ・数字または記号がある場合は前に半角スペース、後ろに半角数字または記号を表示する。

ダイアグラム, 概略図

自動的に生成された説明

### ・実行すると下記の画像の通りに表示される。

ダイアグラム が含まれている画像

自動的に生成された説明

### ・縦の値を入力した後に横の値を入力してくださいを表示する。

概略図 が含まれている画像

自動的に生成された説明

### ・NumRange関数でFALSEを受け取った場合下記の画像のように表示される。

テキスト

自動的に生成された説明

### ・MarkCheck関数でFALSEを受け取った場合下記の画像のように表示される。

テキスト

自動的に生成された説明

### ・どちらかが勝利した際は下記の画像のように表示される。

図形 が含まれている画像

自動的に生成された説明

### ・引き分けだった場合下記の画像のように表示される。

図形 が含まれている画像

自動的に生成された説明

# システム機能フロー図

グラフ, ダイアグラム

自動的に生成された説明

#### フロー説明

##### [盤面を初期化]

3×3のマスをすべて空白にし、[3.1.2](#_[先攻後攻をランダムで決定])に移行する。

##### [3×3のマスと先後を画面出力]

「o」「x」を出力するための3×3のマスと、先攻後攻と「o」「x」が出力し、[3.1.3](#_[縦横のマス数を入力])へ移行する。

##### [縦横のマス数を入力]

「o」「x」を出力した回数をもとに、（〇or×）の番です。と出力し

縦の値を半角で1～3の整数値で入力してください：

横の値を半角で1～3の整数値で入力してください：を出力し、プレイヤーが数値を入力する。

##### [入力した値は規定値内か]

[3.1.3](#_[縦横のマス数を入力])で入力した数値が1～3の内の半角整数値であるかを判別し、1～3以内の半角整数値でなければ半角で1～3の以内の整数値を入力してください。を出力し、[3.1.3](#_[縦横のマス数を入力])に移行し、1～3以内であれば[3.1.5](#_[既に入力されていないか])に移行する。

##### [既に入力されていないか]

入力した数値の位置が既に埋まっていないかを判別し、入力されているなら入力済みのマスです。他のマスを選んでください。を出力し[3.1.3](#_[縦横のマス数を入力])に移行し、入力されてなければ[3.1.6](#_[「〇」「×」を交互に出力])に移行する。

##### [三目並べの画面のマスに「o」「x」を出力]

三目並べの画面のマスに「o」「x」を出力した回数をもとに、偶数回数は「o」、奇数回数は「x」を出力し、[3.1.7](#_[一直線がすべて「〇」、または「×」ですか])に移行する。

ダイアグラム

中程度の精度で自動的に生成された説明

##### [一直線がすべて「o」、または「x」ですか]

縦、横、斜めのいずれかに三つの「o」もしくは「x」が並んでいるかを判別し、並んでいなければ[3.1.8](#_[出力したのは9回目ですか])に移行し、並んでいれば[3.1.9](#_[勝ったプレイヤー名を出力])に移行する。

##### [出力したのは9回目ですか]

「o」と「x」が3×3のマス全てが埋まってあるかを判別し埋まってなければ[3.1.3](#_[縦横のマス数を入力])に移行し、埋まっていれば[3.1.9](#_[結果を出力])に移行する。

##### [結果を出力]

「o」が勝利した場合、「〇」が勝ちました！

「x」が勝利した場合、「×」が勝ちました！を出力し、[3.1.10](#_[もう一度遊びますか])に移行する。

引き分けだった場合、引き分けでした。を出力し、[3.1.10](#_[もう一度遊びますか])に移行する。

##### [もう一度遊びますか]

もう一度遊ぶ…0／やめる…0以外を出力する。

0を入力で「もう一度遊ぶ」、0以外を入力で「やめる」を選択する。

0を選択すると、[3.1.1](#_[プレイヤー名を入力])に移行する。

0以外を選択すると、ゲームを終わる。

# ファイル構成図

テーブル

自動的に生成された説明

# 機能構成

## tic\_tac\_toe.c

###### main()関数

出力した回数をカウントする変数を宣言。

縦横の値を入力する要素数２の一次元配列を宣言。

「o」「x」を出力する各要素数3の二次元配列を宣言。

入力した数値が範囲内かどうかの判定結果を格納する変数を宣言。

入力した数値のマスが既に入力されているかどうかの判定結果を格納する変数を宣言。

三目並べの続行の判定結果を格納する変数を宣言。

勝ち、または引き分けの結果を格納する変数を宣言。

三目並べの終了が出力されるまで回すループ

出力した回数をカウントする変数の初期値を０とする。

三目並べの続行の判定結果を格納する変数の初期値をCONTINUEとする。

勝ち、または引き分けの結果を格納する変数の初期値をUNKNOWNとする。

InitSquare()関数、PreScreen()関数の順で関数を呼び出し、処理を行う。

全てのマスに「o」「x」が出力されるもしくは縦横斜めで３つ「o」「x」が出力されるまで回すループ

入力した数値が範囲内かどうかの判定結果を格納する変数の初期値をFALSEとする。

入力した数値のマスが既に入力されているかどうかの判定結果を格納する変数の初期値をFALSEとする。

・半角整数値1～３の入力を行うループ

入力の処理としてSquareNumber ()関数、NumRange()関数、MarkChecker()関数の順で関数を呼び出す。

NumRange()関数、MarkChecker()どちらもFALSEの場合、再度半角整数値1～３の入力を行うループの処理を行う。TRUEの時に半角整数値1～３の入力を行うループから抜け出す。

・PrintMark()関数で「o」「x」を含んだ三目並べ画面を出力する。

・Bingo()関数、SquareFull()関数の戻り値がUNKNOWNの時、再度全てのマスに「o」「x」が出力されるもしくは縦横斜めで３つ「o」「x」が出力されるまで回すループを行う。O\_WIN、またはX\_WIN、DROWのとき全てのマスに「o」「x」が出力されるもしくは縦横斜めで３つ「o」「x」が出力されるまで回すループが終了する。

・全てのマスに「o」「x」が出力されるもしくは縦横斜めで３つ「o」「x」が出力されるまで回すループのループ回数をカウントし偶数回「o」奇数回「x」を出力するように切り替える。

・GameResult()関数を用いて勝敗を表示し、PlayAgain()関数を用いてENDの時は三目並べの終了が出力されるまで回すループを終了し、CONTINUEの時は、再度三目並べの終了が出力されるまで回すループから始める。

## game\_ready.c

###### InitSquare()関数

引数：char型 盤面の情報を持つ2次元配列のポインタ[列数を表すマクロ]

戻り値の型：void

説明：配列の要素をすべて空白で初期化する。

###### PreScreen()関数

引数：char型 盤面の情報を持つ二次元配列の要素[行数を表すマクロ][列数を表すマクロ]

戻り値の型：void

説明：盤面の情報を持つ二次元配列の要素[行数][列数]と3×3のマス、先攻後攻と「o」「x」が出力する。 表示画面…[3.1.2](#_[3×3のマスと先後を画面出力])0

## get\_input.c

###### SquareNumber()関数

引数：char型 マスの座標の情報を持つ一次元配列のポインタ

int型 ターン数の情報を持つ変数

戻り値の型：void

説明：ターン数から手番を判別し出力する。  
1から3の整数値を配列の要素番号「0」に縦の数値を入力し、「1」に横の数値を入力する。表示画面…[3.1.3](#_[縦横のマス数を入力])

###### NumRange()関数

引数：char型 マスの座標の情報を持つ一次元配列[2]

戻り値の型：[真か偽を表す列挙型](#_真か偽を表す列挙型)のTRUEまたはFALSEを代入した変数

説明：配列の数値が半角整数値1～3の範囲内かどうかを判別し、範囲内であればTRUE、範囲外であれば[3.1.4](#_[入力した値は規定値内か])の警告文を出力しFALSEを返す。

###### MarkChecker()関数

引数：char型 マスの座標の情報を持つ一次元配列[2]

char型 盤面の情報を持つ二次元配列[行数を表すマクロ][列数を表すマクロ]

戻り値の型：[真か偽を表す列挙型](#_真か偽を表す列挙型)のTRUEまたはFALSEを代入した変数

説明：マスの座標における配列の要素が空白であるときTRUEを返し、また空白でないときは[3.1.5](#_[既に入力されていないか])の警告文を出力しFALSEを返す。get\_output.c

###### PrintMark()関数

引数：char型 マスの座標の情報を持つ一次元配列[2]

char型 盤面の情報を持つ二次元配列のポインタ[列数を表すマクロ]

int型 ターン数の情報を持つ変数

戻り値の型：void

説明：配列の要素番号に「o」「x」を入力し、三目並べ画面を出力する。

表示画面…[3.16](#_[三目並べの画面のマスに「o」「x」を出力])

###### Bingo()関数

引数：char型 盤面の情報を持つ二次元配列[行数を表すマクロ][列数を表すマクロ]  
int型 ターン数の情報を持つ変数

戻り値の型：[勝利を表す列挙型](#_勝利を表す列挙型)のO\_WINまたはX\_WINまたはUNKNOWNを格納した変数

説明：「o」が縦、横、斜めのどれかに三つ並んだか判定し並んでいるときはO\_WIN、「x」のときはX\_WIN、並んでいないときはUNKOWNを格納した変数を返す。

###### SquareFull()関数

引数：int型、ターン数の情報を持つ変数

戻り値の型：[勝利を表す列挙型](#_勝利を表す列挙型)のDROWまたはUNKNOWNを格納した変数

説明：ターン数がCAPACITY\_FULLの時DROW、そうでないときはUNKNOWNを格納した変数を返す。

## get\_result.c

###### GameResult()関数

引数:int型、勝敗の情報をもつ変数

戻り値の型：void

説明：勝敗の情報を持つ変数が[勝利を表す列挙型](#_勝利を表す列挙型)のO\_WINまたはX\_WINまたはDROWのときそれぞれ結果の出力をする。 表示画面… [3.1.9](#_[勝ったプレイヤー名を出力])

###### PlayAgain()関数

引数：なし

戻り値：[三目並べ続行を表す列挙型](#_三目並べの続行を表す列挙型)のENDまたはCONTINUEを格納した変数

説明：「もう一度遊ぶ…０/やめる…０以外」で0以外を入力するとEND、０を入力するとCONTINUEを代入した変数を返す。 表示画面…[3.1.10](#_[もう一度遊びますか])

## define.h

###### 真か偽を表す列挙型

|  |  |
| --- | --- |
| TRUE：真 | FALSE：偽 |

###### 三目並べの続行を表す列挙型

|  |  |
| --- | --- |
| CONTINUE：三目並べをもう一度遊ぶ | END：三目並べを終了する |

###### 勝利を表す列挙型

|  |  |
| --- | --- |
| UNKNOWN：勝敗の結果が出ていない。 | DROW：引き分け |
| O\_WIN：「o」の勝利 | X\_WIN：「x」の勝利 |

###### マスが埋まっているかを表すためのマクロ定義

|  |
| --- |
| CAPACITY\_FULL：マスがすべて埋まっている。 9 |