三目並べ詳細仕様書

目次

[**1.** **概要** 1](#_Toc167362315)

[**2.** **用語の定義** 1](#_Toc167362316)

[**3.** **ファイル構成** 2](#_Toc167362317)

[3.1. sanmoku.c(main) 2](#_Toc167362318)

[3.2. board.c 3](#_Toc167362319)

[3.3. board.h 3](#_Toc167362320)

[3.4. input\_num.c 3](#_Toc167362321)

[3.5. input\_num.h 3](#_Toc167362322)

[3.6. decision\_count.c 4](#_Toc167362323)

[3.7. decision\_count.h 4](#_Toc167362324)

[3.8. sanmoku.h 4](#_Toc167362325)

[4. 画面構成 5](#_Toc167362326)

[**4.1.** **マス目・〇×の表示** 5](#_Toc167362327)

[**4.2.** **手番の表示** 5](#_Toc167362328)

[**4.3.** **警告文の表示** 5](#_Toc167362329)

[**4.4.** **勝利数とリトライ・終了選択の表示** 6](#_Toc167362330)

[**5.** **機能構成** 7](#_Toc167362331)

[**5.1.** **board.c** 8](#_Toc167362332)

[5.1.1. ShowBoard 8](#_Toc167362333)

[5.1.2. CheckTurn 8](#_Toc167362334)

[5.1.3. ChangeState 8](#_Toc167362335)

[**5.2.** **input\_num.c** 8](#_Toc167362336)

[5.2.1. InitBoard 9](#_Toc167362337)

[5.2.2. PushNum 9](#_Toc167362338)

[5.2.3. CountWin 9](#_Toc167362339)

[5.2.4. PlayEnd 9](#_Toc167362340)

[**5.3.** **decision\_count.c** 10](#_Toc167362341)

[5.3.1. RangeNum 10](#_Toc167362342)

[5.3.2. CheckNum 10](#_Toc167362343)

[5.3.3. CheckWin 11](#_Toc167362344)

[5.3.4. CheckDraw 11](#_Toc167362345)

# **概要**

本書は作成する三目並べのファイル構成、機能構成、画面構成を示したものである。

# **用語の定義**

記号 → 〇×のこと

手番 → 先行・後攻の順のこと

マス目 → 3×3 の格子のこと

キー → キーボード上の記号・文字・数字のこと

リトライ → 三目並べをもう1度行うこと

空 → 空というマスの状態を表す文字列が格納されていること

# **ファイル構成**

## sanmoku.c(main)

ターン数を表す変数の初期値を0で宣言

〇の勝利数を表す変数の初期値を0で宣言

×の勝利数を表す変数の初期値を0で宣言

入力された値を格納する変数の初期値を0で宣言

数字と記号を格納するマスの配列を宣言

**終了を選択されるまでループ**

InitBord関数、ShowBoard関数の順に呼び出す。

**勝利か引き分け判定がつくまでループ**

CheckTurn関数を処理しPushNum関数、RangeNum関数、CheckNum関数の順で呼び出し、処理をする。

RangeNum関数、CheckNum関数がFALSEを返した場合は、4.3の警告文を表示して再度入力を促す。

ChangeState関数でマスの中身を数字と記号を置き換えた後、ShowBoard関数で記号を置き換えたマスを表示する。CheckWin関数、CheckDraw関数を呼び出し勝利・引き分け判定処理をする。

CheckWin関数で勝利した記号、CheckDraw関数でENDが返された場合はループを抜け出す。EXTENDが返された場合は、ターン数を１加算してCheckTurn関数から再度ループする。

**終了が選択されるまでループ**

どちらかが勝利した場合はCountWin関数で勝利数をカウントし、4.4の文章をDisplayWin関数で表示した後、PlayEnd関数を呼び出し、処理をする。

引き分けの場合は何も表示せずにPlayEnd関数を呼び出し、処理をする。

QUITの場合は4.4の記号の合計勝利数を表示して終了し、TRYの場合は終了を選択されるまでループに戻りInitBord関数から再度始まる。

## board.c

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 関数 | ShowBoard | CheckTurn | DisplayWin | ChangeState |
| 説明 | 現在のマスの中身に入っている数字や記号の表示 | 現在のターンを判定して表示する | 「〇の勝利」、「×の勝利」を表示する | 入力した値に当てはまるマスの中身の数字と記号を入れ替える |

## board.h

## input\_num.c

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 関数 | InitBord | PushNum | CountWin | PlayEnd |
| 説明 | 全てのマスの中身に数字を入れる | 値を入力する | 勝利数をカウントする | リトライか終了を入力する |

## input\_num.h

## decision\_count.c

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 関数 | RangeNum | CheckNum | CheckWin | CheckDraw |
| 説明 | 入力された値が範囲内かを判定する | 入力された値の中身に既に記号が入っているかを  判定する | 記号が縦・横・斜めに3つ揃っているか判定する | 引き分け判定をする(表示はしない) |

## decision\_count.h

## sanmoku.h

ELEMENT：マスの配列の要素数

真偽を示す列挙型を定義（TRUE：真を示す、FALSE：偽を示す）

現在の手番を示す列挙型を定義（CIRCLE：〇の手番、CROSS：×の手番）

リトライと終了を示す列挙型を定義（TRY：リトライを示す、QUIT：終了を示す）

記号の勝利を示す列挙型を定義（CIRCLE\_WIN：〇の勝利、CROSS\_WIN：×の勝利）

三目並べの１ゲームの終了と継続を示す列挙型を定義（END：三目並べ終了、EXTEND：三目並べ継続）

# 画面構成

## **マス目・〇×の表示**



マスの数字の左右は半角スペース2個の幅、数字の上下はスペース１個分の高さ、マスの数字は全角で表示する。

マス目を表示し、入力された値のマスに〇×を表示する。

マスのスペースは

## **手番の表示**



4.1.の図の下に左の図にある手番の文章が表示される。



## **警告文の表示**



１～９の値以外が入力された場合に表示される。



記号の重複判定に当てはまった場合に表示される。

リトライ・終了の選択でTとQ以外が入力された場合に表示される。



## **勝利数とリトライ・終了選択の表示**



勝ち判定された場合勝者を4.1の図の下に表示する。

その後勝利数の下にリトライ・終了選択文が表示される。

引き分けの場合は結果を表示せずにリトライ・終了選択文が表示される。







終了が選択された場合に記号ごとの合計勝利数を表示される。

# **機能構成**

フローチャート  
ダイアグラム

自動的に生成された説明

## **board.c**

### ShowBoard

関数の型 ：void

引数 ：マスの配列

戻り値 ：なし

マスの配列を受け取り、4.1のマス目を表示するとともにマスの中身の数字または記号を表示させる。

記号の場合は、全角の記号を表示する。

数字の場合は、全角の数字を表示する。

### CheckTurn

関数の型 ： int

引数 ：ターン数

戻り値 ： CIRCLEまたはCROSS

ループの中でターン数をカウントさせ現在のターン数を受け取り偶数、奇数を判別する。

偶数ならCIRCLE、奇数ならCROSSを返す。

### DisplayWin

関数の型 ：int

引数 ：CIRCLE\_WINまたはCROSS\_WIN

戻り値 ：EXTEND

CIRCLE\_WINまたはCROSS\_WINを受け取り4.4の勝者の文章を表示をする

CIRCLE\_WINを受け取った場合は、「〇の勝利」と表示する。

CROSS\_WINを受け取った場合は、「×の勝利」と表示する。

### ChangeState

関数の型 ：int

引数 ：入力した値、CIRCLEまたはCROSS

戻り値 ：入力された値に該当するマスに現在のターンの記号を入れる

入力された値を格納する変数に当てはまるマスの中身に入っている数字を現在のターンの記号に置き換えて、置き換えたマスの中身を返す。

## **input\_num.c**

### InitBoard

関数の型 ：void

引数 ：マスの配列

戻り値 ：なし

全てのマスの中身に１～９の数字を１つずつ格納する。

### PushNum

関数の型 ：int

引数 ：入力された値を入れる変数・CIRCLEまたはCROSS

戻り値 ：なし

CIRCLEが返された場合は、4.2の「〇の番です」を表示してCROSSが返された場合は、4.2の「×の番です」を表示してプレイヤーに入力を求め、入力された値を配列に格納する。

### CountWin

関数の型 ：int

引数 ：CIRCLE\_WINまたはCROSS\_WIN

戻り値 ：〇の勝利数を表す変数または×の勝利数を表す変数

勝利した記号を受け取り、該当する記号を判別する。

〇が勝利した場合、〇の勝利数を１加算して返す。

×が勝利した場合、×の勝利数を１加算して返す。

### PlayEnd

関数の型 ：int

引数 ：EXTEND

戻り値 ：TRY またはQUIT

EXTENDを受け取り、4.4の文章を表示させプレイヤーに入力を求め、リトライか終了かのどちらかが入力されたか判別する。

リトライの場合は、TRYを返す。

終了の場合は、QUITを返す。

指定されたキー以外が入力された場合は、4,3の警告文を表示し再度入力を促す。

## **decision\_count.c**

### RangeNum

関数の型 ：int

引数 ：入力された値を格納する変数

戻り値 ：TRUEまたはFALSE

入力された値を受け取り、半角の１以上９以下の値か判別する。

範囲内の場合は、TRUEを返す。

範囲外の場合は、FALSEを返す。

### CheckNum

関数の型 ：int

引数 ：入力された値を格納する変数、数字と記号を格納するマスの配列のポインタ

戻り値 ：TRUEまたはFALSE

入力された値を格納する変数を受け取り、値に該当するマスの中身に記号が格納されているか判別する。

格納されていない場合は、TRUEを返す。

格納されている場合は、FALSEを返す。

### CheckWin

関数の型 ：int

引数 ：マスの配列

戻り値 ：CIRCLE\_WINまたはCROSS\_WIN

記号が縦横斜めいずれかに同じ記号が揃っているか判定。

揃っている場合は、揃った記号を返す。揃っていない場合は何も返さない。

### CheckDraw

関数の型 ：int

引数 ：main関数でカウントしたターン数

戻り値 ：ENDまたはEXTEND

ループの中でターン数をカウントさせ現在のターン数を受け取り、全てのマスが記号で埋まっているか判別する。

全てのマスが埋まった場合、ENDを返す。

全てのマスが埋まっていない場合、EXTENDを返す。