**三目並べ  
詳細仕様書**

Fチーム  
小島隆聖、柴田鈴花

目次

[目次 2](#_Toc167097589)

[1. 概要 3](#_Toc167097590)

[2. ファイル機能構成 3](#_Toc167097591)

[2.1 ソースファイル 3](#_Toc167097592)

[2.2 ヘッダーファイル 5](#_Toc167097593)

[2.3 ファイル構成図 7](#_Toc167097594)

[3. 関数説明 8](#_Toc167097595)

[3.1 ソースファイル内容 8](#_Toc167097596)

[3.2 ヘッダーファイル内容 14](#_Toc167097597)

[4. 画面構成 15](#_Toc167097598)

[4.1 盤面の表示方法 15](#_Toc167097599)

[4.2 画面遷移一覧 15](#_Toc167097600)

[4.3 画面遷移 16](#_Toc167097601)

# 概要

本仕様書は、C言語を用いた三目並べシステムのファイル機能構成と関数説明、画面構成を示したものである。

# ファイル機能構成

* 1. ソースファイル

1. 全体進行

|  |  |
| --- | --- |
| ソースファイル名 | main.c |
| 関数 | main、PlayerChange |
| 機能 | * メイン * プレイヤーを交代する |

1. 盤面入力機能

|  |  |
| --- | --- |
| ソースファイル名 | board\_input.c |
| 関数 | BoardInitialize、PlayerInput |
| 機能 | * 3×３の9マス盤面の初期化 * プレイヤーの盤面入力 |

1. リトライ機能

|  |  |
| --- | --- |
| ソースファイル名 | retry.c |
| 関数 | Retry |
| 機能 | * リトライor終了のキーボード入力 * リトライの場合 三目並べをリスタート 終了の場合 三目並べ終了 |

1. エラー機能

|  |  |
| --- | --- |
| ソースファイル名 | error.c |
| 関数 | InputErrorBoard、InputErrorRetry |
| 機能 | * 入力エラーがあるか判定 * エラーであった場合、エラー結果を表示 |

1. 盤面出力機能

|  |  |
| --- | --- |
| ソースファイル名 | board\_output.c |
| 関数 | BoardOutput |
| 機能 | * 現在の盤面情報を表示 |

1. 勝敗表示機能

|  |  |
| --- | --- |
| ソースファイル名 | result\_output.c |
| 関数 | ResultJudge、ResultOutput |
| 機能 | * 三目並べの勝敗の判定 * 勝敗の表示 |

* 1. ヘッダーファイル

1. board\_input.h

|  |  |
| --- | --- |
| ヘッダーファイル名 | board\_input.h |
| プロトタイプ宣言  （関数名のみ記載） | BoadInitialize、PlayerInput |

1. retry.h

|  |  |
| --- | --- |
| ヘッダーファイル名 | retry.h |
| プロトタイプ宣言  （関数名のみ記載） | Retry |

1. error.h

|  |  |
| --- | --- |
| ヘッダーファイル名 | error.h |
| プロトタイプ宣言  （関数名のみ記載） | InputErrorBoard、InputErrorRetry |

1. board\_output.h

|  |  |
| --- | --- |
| ヘッダーファイル名 | board\_output.h |
| プロトタイプ宣言  （関数名のみ記載） | BoardOutput |

1. result\_output.h

|  |  |
| --- | --- |
| ヘッダーファイル名 | result\_output.h |
| プロトタイプ宣言  （関数名のみ記載） | ResultJudge、ResultOutput |

1. enum.h

|  |  |
| --- | --- |
| ヘッダーファイル名 | enum.h |
| プロトタイプ宣言  （関数名のみ記載） | (型名)  JUDGE  (列挙子)  ERROR、RETRY、FINISH、 PLAYER１＿WIN、PLAYER2＿WIN、DRAW  (型名)  PLAYER  (列挙子)  PLAYER1、PLAYER2  (マクロ定義)  #define INDEX 3 |

* 1. ファイル構成図

**ソースファイル**

**(関数 )**

**①main.c**

└ main

└ PlayerChange

**②input.c**

├ BoardInitialize

├ PlayerInput

└ BoadInput

**③retry.c**

└ Retry

**④error.c**

├ InputErrorBoard

└ InputErrorRetry

**⑤board\_output.c**

└ BoardOutput

**⑥result\_output.c**

├ ResultJudge

└ ResultOutput

**ヘッダーファイル**

**①input.h（プロトタイプ宣言）**

BoadInitialize()

PlayerInput

BoadInput

**②retry.h**

Retry

**③error.h**

InputErrorBoard

InputErrorRetry

**④board\_output.c**

BoadOutput

**⑤result\_output.c**

ResultJudge

ResultOutput

**⑥enum.h（列挙体）（マクロ定義）**

・ JUDGE

├── ERROR

├── RETRY

├── FINISH

├── PLAYER1\_WIN

├── PLAYER2\_WIN

├── DRAW

└── CONTINUE

・ PLAYER

├── PLAYER1

└── PLAYER2

・#define INDEX 3

# 関数説明

## ソースファイル内容

1. main.c（全体進行）

**BoardInitialize関数の呼び出し**→初期盤面マスの作成

**BoardOutput関数の呼び出し**→受け取った盤面を表示

**PlayerInput関数の呼び出し**→勝利/引き分けを判定

**BoardInput関数の呼び出し**→入力値を受け取る

**ResultJudge関数の呼び出し**→読み取った盤面から勝敗判定

**勝利/引き分け判定がない場合**→PlayerChange関数を呼び出し後②に戻る

**勝利/引き分け判定がある場合**→BoadOutput関数を呼び出す

**ResultOutput関数の呼び出し**→勝敗結果を表示

**Retly関数の呼び出し**→リトライor終了を判定

**リトライの場合**→①に戻る

**終了の場合**→終了画面を表示してプログラム終了

③

⑥

⑦

②

⑧

①

④

⑤

⑨

⑪

⑩

ダイアグラム

自動的に生成された説明

* main関数
* （処理）
  + char型配列で縦：3マス×横：3マスの盤面を作成する。
  + BoardInitialize関数を呼び出す。
  + “～三目並べ～”、“― ゲーム開始 ―”を画面表示する。
* カレンダー

  自動的に生成された説明
* BoardOutput関数を呼び出す。
* PlayerInput関数を呼び出す。
* BoardInput関数を呼び出す。
* ResultJudge関数を呼び出す。
* ResultJudge関数の戻り値がCONTINUEであれば、PlayerChange関数を呼び出し後、2．に戻る。
* ※2.～6.の処理を1ターンとする。2.に戻る度にターン数が増加する。
* ResultJudge関数の戻り値がCONTINUE以外であればBoadOutput関数呼び出し後、ResultOutput関数を呼び出す。
* Retly関数を呼び出す。
* 戻り値がFINISHの場合、“― 終了 ―”を画面表示してプログラムを終了する。
* 戻り値がRETRYの場合、1.2に戻る。
* テキスト が含まれている画像

  自動的に生成された説明
* PlayerChange関数
* （処理）
* 入力プレイヤー名(PLAYER1 or PLAYER2)を引数で受け取る。
* プレイヤー名を判定する。
* PLAYER1の場合 PLAYER2を返す。
* PLAYER2の場合 PLAYER1を返す。
* input.c（盤面入力機能）
* BoardInitialize関数
* （処理）
* 引数としてchar配列 (盤面)を受け取る。
* char配列 (盤面)に下図の様に入力位置を示す数字(半角)を代入する。

|  |  |  |
| --- | --- | --- |
| * 1 | * 2 | * 3 |
| * 4 | * 5 | * 6 |
| * 7 | * 8 | * 9 |

* PlayerInput関数
* （処理）
* 入力プレイヤー名と”1~9の数字を入力してください”の文字を画面表示する。
* カレンダー

  自動的に生成された説明
* キーボードからの入力値を受け取る。
* 入力値をInputErrorBoard関数に渡す。
* InputErrorBoard関数からERRORが返ってきた場合、1．に戻る。
* InputErrorBoard関数からCONTINUEが返ってきた場合、入力値を返す。
* BoardInput関数
* （処理）
* 引数として入力プレイヤー名(PLAYER1 or PLAYER2)、入力値、盤面(char配列)を受け取る。
* char配列にプレイヤーの記号を入力する。(入力値と同じ数字位置)
* PLAYER1の場合：o
* PLAYER2の場合：x
* retry.c（リトライ機能）
* Retly関数
* （処理）
* 画面に”リトライ…１、終了…2　入力:”の文字を表示する。
* 時計 が含まれている画像

  自動的に生成された説明
* キーボードからの入力値を受け取る。
* 入力値をInputErrorRetry関数に渡す。
* 戻り値がERRORの場合1．に戻る。
* CONTINUEの場合入力値を判定する。
* 1の場合 RETRYを返す。
* 2の場合 FINISHを返す。
* error.c（エラー機能）
* InputErrorBoard関数
* （処理）
* 引数(入力値、char配列(盤面)を受け取る。
* 引数(入力値)を判定する。  
  入力値が１～９ではない場合、入力値と一致する数字が盤面にない場合  
  →“エラー：適切な数字を入力してください”を画面表示し、ERRORを返す。
* テキスト が含まれている画像

  自動的に生成された説明
* エラーがなければCONTINUEを返す。
* InputErrorRetry関数
* （処理）
* 引数(入力値)を受け取る。
* 引数を判定する。  
  値が１or２でない場合  
  →“エラー：適切な数字を入力してください”を画面表示し、ERRORを返す。
* テキスト が含まれている画像

  自動的に生成された説明
* エラーがなければCONTINUEを返す。
* board\_output.c (盤面出力機能)
* BoardOutput関数
* （処理）
* 引数としてchar配列(盤面)を受け取る。
* 受け取った盤面を読み取り下図の様に画面表示する。
* （盤面構成については4.1 盤面の表示方法にて記載）
* グラフィカル ユーザー インターフェイス, アプリケーション

  自動的に生成された説明
* result.c (勝敗表示機能)
* ResultJudge関数
* （処理）
* 引数（char配列(盤面)・入力プレイヤー名(PLAYER1 or PLAYER2)・現在のターン数）を受け取る。
* 読み取った盤面を走査し、以下の順序で勝敗を判定する。
  + 縦・横・斜めが同一記号で3つ並んでいる場合  
    →PLAYER1\_WIN or PLAYER2\_WINを返す。
  + 現在のターン数が9回目だった場合（char配列が全て記号で埋まっている状態）  
    →DRAWを返す。
  + 勝敗がついていない場合
* →CONTINUEを返す。
* ResultOutput関数
* （処理）
* 引数（勝敗結果）を読み取る。
* 引数がPLAYER１＿WINの場合  
  →“プレイヤー１の勝利！”を画面に表示する。
* 図形, 矢印

  自動的に生成された説明
* 引数がPLAYER2＿WINの場合  
  →“プレイヤー２の勝利！”を画面に表示する。
* 図形, 矢印

  自動的に生成された説明
* 引数がDRAWの場合  
  →“引き分け”を画面に表示する。
* 図形, 矢印

  自動的に生成された説明

## ヘッダーファイル内容

* ①input.h
* void BoardInitialize (char)
* 入力値PlyerInput (void)
* void BoardInput(char)
* ②retry.h
* JUDGE Retry (void)
* ③error.h
* JUDGE InputErrorBoard(入力値、char)
* JUDGE InputErrorRetry(入力値)
* ④board\_output.h
* void BoardOutput(char)
* ⑤result.h
* JUDGE ResultJudge(char,PLAYER,intターン数)
* void ResultOutput(JUDGE)
* ⑥enum.h
* （型名）
* JUDGE 判定や戻り値に使用
* （列挙子）
* ERROR
* RETRY
* FINISH
* PLAYER1\_WIN
* PLAYER2\_WIN
* DRAW
* CONTINUE
* （型名）
* PLAYER 入力プレイヤー判別に使用
* （列挙子）
* PLAYER1
* PLAYER2
* （マクロ定義）
* #define INDEX 3

# 画面構成

### 盤面の表示方法

* 数字と記号は全て半角を使用し、  
  記号は小文字の“o(オー)”と“x(エックス)”を使用する。
* マスの数字と記号の両隣には半角スペースを使用する。
* グラフィカル ユーザー インターフェイス, アプリケーション

  自動的に生成された説明

**␣　␣**

### 画面遷移一覧

* ゲーム開始＆プレイヤー１入力画面
* プレイヤー１入力後＆プレイヤー２入力画面
* プレイヤー２入力後画面
* プレイヤー１勝利画面
* プレイヤー２勝利画面
* 引き分け画面
* リトライ画面
* エラー画面

### 画面遷移

* ゲーム開始＆プレイヤー１入力画面
* カレンダー

  自動的に生成された説明
* プレイヤー１入力後＆プレイヤー２入力前画面
* カレンダー

  中程度の精度で自動的に生成された説明
* プレイヤー２入力後画面
* テキスト

  中程度の精度で自動的に生成された説明
* プレイヤー１勝利画面
* ダイアグラム, 図形, 矢印

  自動的に生成された説明
* プレイヤー２勝利画面
* 図形, 矢印

  自動的に生成された説明
* 引き分け画面
* 図形, 矢印

  自動的に生成された説明
* リトライ画面
* テキスト が含まれている画像

  自動的に生成された説明
* エラー画面

テキスト が含まれている画像

自動的に生成された説明