

# 情報学群実験第 1 最終レポート

1250373 溝口洸熙 \*

2022 年 7 月 28 日

## 概要

## 目次

はじめに	2
仕様 1	3
1.1 処理概要 . . . . .	3
1.2 処理 . . . . .	4
仕様 2	5
2.1 処理概要 . . . . .	5
2.2 処理 . . . . .	6
仕様 3	7
3.1 処理の概要 . . . . .	7
ソースコード	8

---

\* 高知工科大学 情報学群 2 年生

## はじめに

### レポートについて

このレポートは， $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X } 2_{\epsilon}$  を用いて作成している．図やグラフは  $\text{TikZ}$  を用いて描画しており，ソースコードは `listing` を用いて表記している．

### 符号化と変数

あるパネルのステータスを示す符号と，新たに追加したグローバル変数を，以下に示す．

符号とステータス		新たに追加した変数	
符号	ステータス	変数名	役割
0	爆弾以外	<code>int originalTable</code>	生成した盤面の初期状態を記憶する．
1	開かれたパネル		
-1	爆弾		2 手目以降で <code>true</code> になる変数．
-2	旗が立っている	<code>Boolean tr</code>	1 手目で爆弾に当たることを回避するため．

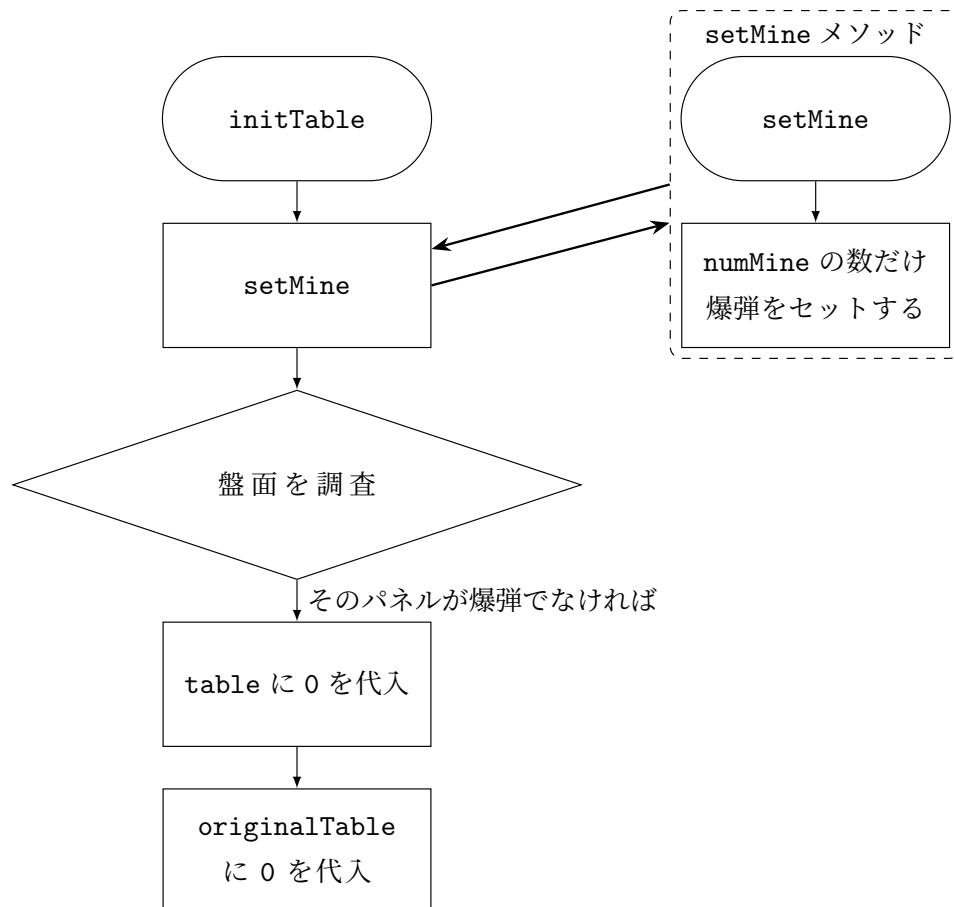
## 仕様 1

### 仕様 1.

ゲーム開始時に、盤面上へランダムに地雷を設置する.

### 1.1 処理概要

Fig 1.1: 盤面上へランダムに地雷を設置する



## 1.2 処理

initTable (src. 1) の処理

盤面を初期化するにあたって、以下の処理を行う。

- 1) 爆弾の設置パネルを setMine メソッドで定める。
- 2) table の全ての行と列に 0 を代入する。ただし、爆弾であるパネルは上書きしない。

```
for (int x = 0; x < this.height; x++) {
    for (int y = 0; y < this.width; y++) {
        if (this.table[x][y] == -1) { // 爆弾がセットされている場所は避ける
            continue;
        }
        this.table[x][y] = 0; // 爆弾の場所以外は0で初期化
    }
}
```

- 3) originalTable の全ての行に 0 を代入する。ただし、爆弾であるパネルは上書きしない。

```
this.originalTable[x][y] = 0;
```

setMine (src. 2) の処理

盤面に爆弾を配置するにあたって、以下の処理を行う。

- 1) 爆弾の個数を数える count 変数を定義する。
- 2) 指定された爆弾の個数が count になるまで、爆弾を配置する。

```
while (count != this.numMine) { //
    numMine の数だけ爆弾をセットできたらループを抜ける
    ...
}
```

- 3) 爆弾の配置はランダムである。乱数で指定されたパネルが既に爆弾であれば再度乱数を生成し、爆弾が新たにセットできる場所では、table,originalTable の乱数値 Index を -1 に設定し、count をインクリメントする。

```
...
int x = new java.util.Random().nextInt(getHeight());
int y = new java.util.Random().nextInt(getWidth());
if (this.table[x][y] == -1) {
    // [x][y] にすでに爆弾がセットされていたら、もう一度乱数を決め直す
    continue;
}
count++;
...
```

仕様 1 終

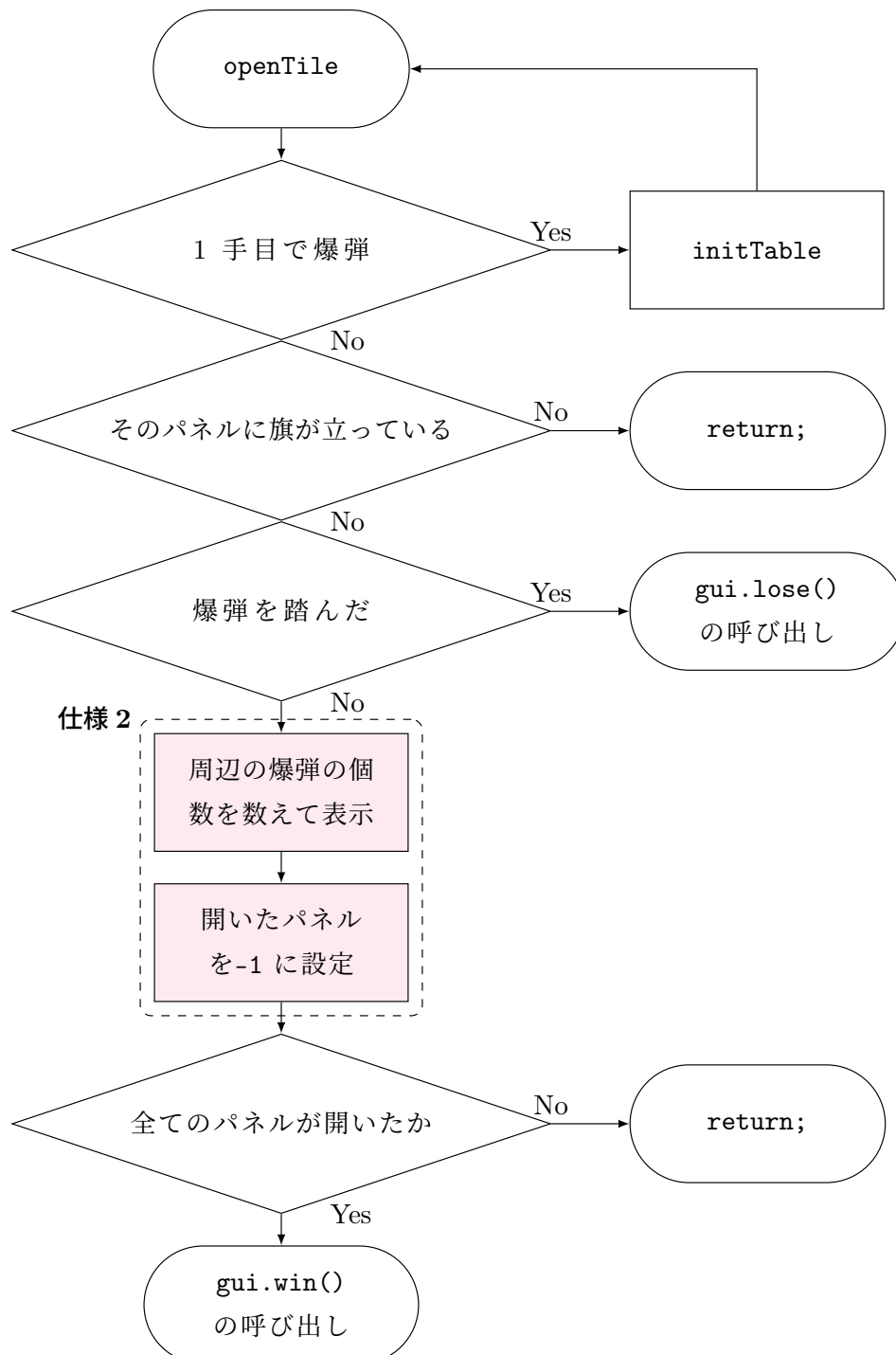
## 仕様 2

### 処理 2.

パネルを左クリックした際、クリックしたパネルを開く.

### 2.1 処理概要

Fig 2.2: タイルを開くときの処理



## 2.2 処理

openTile (src. 3) の処理の一部

- 1) パネルに爆弾がない場合、その周辺の爆弾個数を returnMine (src. 4) メソッドで取得し、そのパネルに表示する.

```
String mc = String.valueOf(mineCount);  
gui.setTextToTile(x, y, mc); // 爆弾の個数を表示
```

- 2) 全てのパネルが開いたか否か確認する. もし、全てのパネルが開いていたら勝利となるので gui.win を呼び出し、開いていないパネルが存在すれば、return; する.

```
int mineCount = this.returnMine(x, y, gui); // 周辺の爆弾の個数を調査  
this.table[x][y] = 1; // 開かれたパネルの値を1に設定  
...  
// 爆弾以外のパネルが全て開いているか確認  
for (int i = 0; i < getHeight(); i++) {  
    for (int j = 0; j < getWidth(); j++) {  
        if (this.table[i][j] == 0) { return; }  
    }  
}  
gui.win();
```

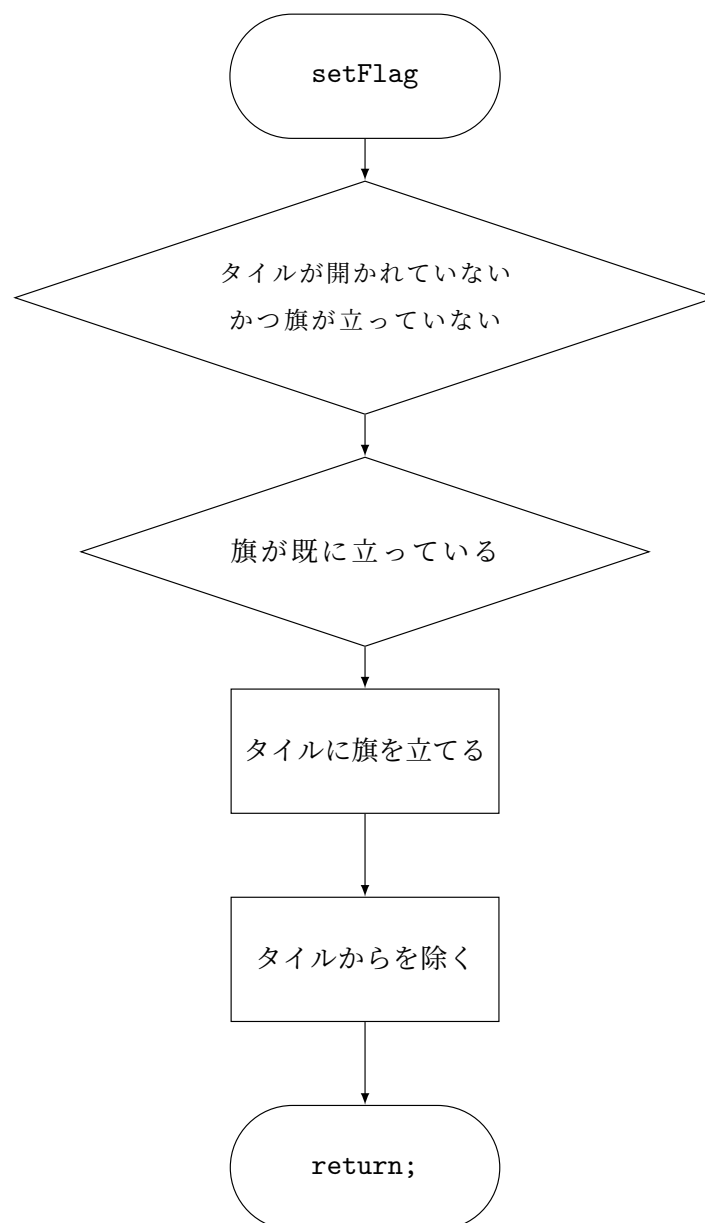
## 仕様 3

### 仕様 3.

開いていないパネルを右クリックした際、そのパネルに旗を立てる。また、旗が立てられているパネルの場合には畑を取り除き、旗が取り除かれるまで左クリックでパネルは開けない。

### 3.1 処理の概要

Fig 3.3: 旗を立てるときの処理



## ソースコード

src. 1: `initTable`

src. 2: `setMine`

src. 3: `openTile`

src. 4: `returnMine`