情報学群実験第 1　最終レポート
学籍番号: 1230341 氏名: 高原成功
2020 年 8 月 3 日（月） 提出

MineSweeper の実装について

概要

今回の情報学群実験第 1 最終レポートでは、MineSweeper を実装する上での考え方や問題点の解決方法を述べている。今回の課題を通して学んだ考え方を今後の勉強にも役立てていきたい。

1.　はじめに

MineSweeper は、正方形のパネルを並べて構成された長方形の盤面が用意され、その中にある地雷のパネルを避けながら地雷以外のパネルを開けていくコンピュータゲームである[1]。私はこのゲームのことを今回の情報学群実験第 1 で初めて知った。MineSweeper の仕様について、文章の説明を読んで大体の事は把握できたが、今までやったことがないゲームだったので、一度このゲームをプレイしてみた後、MineSweeper の実装について考えた。

2.　MineSweeper の仕様

・仕様 1

最初の仕様である、”ゲーム開始時に、盤面にランダムに地雷を設置する”という仕様について考える。盤面は table[x][y]の 2 次元配列で表されているので、乱数を生成し、その乱数の値を x と y に入れてランダムな場所に地雷を設置するようにした[2]。地雷を設置した場所には-1 を格納している。地雷の個数については、地雷を設置した回数を数える変数 count を用意して、while 文で numberOfBombs の回数だけ地雷を設置したらループを抜けて設置を完了するようにした。以上のことを setBombs メソッドに実装し、ランダムに地雷を設置できるようにした。

・仕様 2，仕様 3，仕様 5

仕様 2，仕様 3，仕様 5 は、同じメソッドの中での実装なので、まとめて記す。

仕様 2 は、左クリックでパネルを開く動作になっている。左クリックの動作は、プログラムの雛形で openTile メソッドとその引数を用意してあったので、openTile メソッドの中に動作のプログラムを書き加えて作成した。

左クリックの動作は 2 パターンあり、1 つは、地雷が隠されているパネルをクリックするとゲームオーバーとなり、その旨を表示する。また、全てのパネルを開く。もうひとつは、クリックしたパネルに地雷がなかった場合、隣接する 8 マス内の地雷の個数を表示する、という動作である。

まずは、クリックしたパネルに地雷があった場合を考える。パネルに地雷があった場合は、全てのパネルを開くので、openAllTiles メソッドを呼び出して全てのパネルを開くようにした。また、引数 gui を使って、ゲームオーバーの旨を表示するようにした。openAllTiles メソッドは、for 文で全てのパネルを調査し、-1（地雷）が格納されている場所を全て表示するようにしている。

次に、クリックしたパネルに地雷が無かった場合だが、これは盤面を表した表を使って考える。まず、表 1 において、クリックして開いたパネルを（x1, y1）だとする。この（x1, y1）を変数 x, y を使って（x, y）と置き換えると、隣接する 8 マスは 表 2 のように表すことができる。この考えを使って、隣接する 8 マス内の地雷の調査を実現した。二重の for 文を使って、外側のループ変数を x-1、内側のループ変数を y-1 からスタートさせ、それぞれ 3 回ずつ繰り返すことにより、隣接する 8 マス内の地雷の調査を実現した。8 マス内の-1（地雷）の個数を変数 bombsCount を使ってカウントし、カウントした数は String 型に型変換した後、クリックしたパネルに表示するようにした[2]。クリックして開いたパネルの配列の値は 1 に更新して、地雷以外の全てのパネルの値が 1 になったとき、引数 gui を使ってゲームクリアの旨を表示するようにしている。

以上で仕様 2, 仕様 3, 仕様 5 の実装は完了した。

表 1

x ＼ y	y0	y1	y2
x0			
x1		ここを開いた！	
x2			

表 2

x ＼ y	y0	y1	y2
x0	(x-1, y-1)	(x-1, y)	(x-1, y+1)
x1	(x, y-1)	(x, y)	(x, y+1)
x2	(x+1, y-1)	(x+1, y)	(x+1, y+1)

・仕様 4

仕様 4 は、右クリックでパネルに旗を立て、もう一度右クリックすると旗を取り除くという動作である。仕様 4 も仕様 2 と同じように setFlag メソッドの中に動作のプログラムを書き加えて、右クリックの動作を実装した。

右クリックしたパネルの配列の値を-2 に更新することを旗を立てたフラグとし、クリックしたパネルの値が-2 かそれ以外かで旗が立っているか立っていないかを判定するようにした。左クリックした時、openTile メソッドで旗が立っているかどうかを判定し、もし旗が立っていれば return を使って何もしないようにした。この何もしない動作で、左クリックでパネルを開けないという動作を実現している。

3. 実装する上での問題点と解決方法

仕様 1 〜仕様 5 の全ての実装が完了し、MineSweeper を実行してみた。しかし、実装したプログラムには 3 つの問題点があった。

1 つ目は、仕様 2 での隣接する 8 マス内の地雷の調査の部分にあった。地雷調査を実現している for 文のループ変数は、(x-1, y-1) からスタートしているので、x=0 または y=0 のとき、-1 という存在しない配列のインデックスにアクセスしてしまうのでエラーがおきてしまっていた。この問題は、配列の範囲外にアクセスしないように if 文で判定し、範囲外の場合は continue を使うことで解決することができた。

2 つ目の問題は、地雷調査の部分を修正することで MineSweeper を実行することができたが、実行した MineSweeper の地雷の数がおかしいということだった。これは、仕様 1 で生成した乱数が重複する場合を考慮していなかったため、地雷の数が期待したものにならないということに気がついた。この解決策としては、地雷を設置するときに、配列の値が -1 だった（すでに地雷が設置されていた）ときは、もう一度乱数を決めなおす、という if 文を仕様 1 に付け加えることで解決することができた。

最後の 3 つ目の問題は、仕様 4 の右クリックで旗を立てるときに、配列の値を-2 で上書きしてしまうので、元の値が-1 なのか 0 なのか 1 なのかわからなくなってしまうということだった。この問題は、MineSweeper の盤面を表す配列とは別に、初期盤面を保存しておく配列を作ることで解決できた。旗を立てて配列の値を-2 で上書きしても、初期の盤面を保存しているので、その配列から数値を確認することができる。このようにして、仕様 4 の問題点も解決できた。

4. まとめ

今回の MineSweeper を実装する課題では、プログラムの雛形が用意されていたので比較的プログラムが組みやすく、0 から作るよりも容易に MineSweeper を実装することができた。しかし、組みやすいと言っても MineSweeper の仕様 1〜仕様 5 までを実装し、ちゃんとプログラムが動くようになるまでかなり時間がかかった。これからの講義や実験では、もっと難しく複雑なプログラムを扱うことは明白だろう。今回の課題で得た経験を糧にし、次のレベルへと進んでいきたい。

参考文献

[1] Minesweeper (video game), Wikipedia,
https://en.wikipedia.org/wiki/Minesweeper_(video_game), 2020 年 8 月 1 日閲覧.

[2] 中山清喬/国本大悟 著, 株式会社フレアリンク 監修, スッキリわかる Java 入門 第 3 版, 株式会社インプレス, 2020 年 1 月 21 日 第 1 版第 2 刷発行.