要求仕様書

1. 全体

1-1. プログラム作成の背景

2 年後期に開講される「プログラミング言語Ⅱ及び演習」の自由課題の一環で作成する Java プログラムである.

1-2. 課題の目的

作品を作成することで Java 言語とスパイラル開発の理解を深める.

２．作成の前提条件

2-1. 作成における制約条件

Java で開発したプログラムであること.

クラス数 3, 行数 200 行以上のプログラムであること.

2-2. 学則上の制約条件

(1) 必ず本人が作成したプログラムであること

(2) 独自性は必ずしも必要ではないが, ソースコードは作成者本人のオリジナルであること.

(3) 参考にしたサイト, 書籍, プログラム等あれば, 必ず参考資料として明記すること. ない場合は 不正行為とみなされる場合がある.

３．要件 （説明書）

3-1. 概要

本ゲームは, 迷路状の街を効率よく巡ってクーポン券を使用し, いくら得をできるかを競うゲームである. ゲームは 1 日目の 6:00 からスタートし, 4 日目の 6:00 になると終了する. また, ゲーム開始時の所持金は 20000 円であり, この所持金がなくなると制限時間を待たずしてゲームが終了する.

3-2. 詳細

3-2-1. ゲーム開始

S キーを押すとゲームが開始する.

3-2-2. ゲーム中

（1）画面について

左側の画面には町の様子が表示され, （以下ゲーム画面と呼称）右側の画面にはクーポン券や現在時刻などの情報が表示される（以下クーポン画面と呼称).

ゲーム画面上の黄色の三角形はプレーヤーを表しており, その最も鋭利な角がプレーヤーの向いている方向を示してる. また, ゲーム画面上の黒色

の太線は壁を表しており，プレーヤーはこれを超えることはできない．ゲーム画面上の家マークは店を表しており，マーク上には店名が記されている．また，マークの色はその店のイメージカラーを示している．

クーポン画面上部の水色のエリアには，現在時刻，残金，得した金額が表示される．また，画面下部には最大５つのクーポン券が表示される．クーポン券には店名，対象商品名，通常価格，割引価格，有効期限が記載されている．また，クーポン券の色はその店のイメージカラーによって決まっている．

（２）操作方法について

矢印キーを押すことにより，該当する方向へプレーヤーを動かすことができる．また，町は計９枚の迷路によって構成されており，表示されている迷路外へ出ると隣の迷路へ移ることができる．さらに，店とプレーヤーが接触した状態で，クーポン画面上の該当するクーポン券をクリックするとクーポン券を使用することができる．

（３）クーポン券について

クーポン券は一定間隔でランダムに生成され，使用した，もしくは有効期限が切れた場合は自動的に画面から消える．また，クーポン券はその店の営業時間内でのみ使用可能であり，営業時間外では店とプレーヤーが接触した状態でクーポン画面上の該当するクーポン券をクリックしても使用することはできない．各店の営業時間は以下のようである．

RAWSON	24 時間営業
モクドナルド	24 時間営業
ガヌト	24 時間営業
吉田家	24 時間営業
ミニマムバリュー	24 時間営業
TOHU シネマズ	8 時～22 時
ムーンバックス	8 時～23 時
マツモトキヨコ	9 時～23 時
INUQLO	10 時～20 時
YAMATA 電機	10 時～20 時
ミトリ	11 時～20 時

また，クーポン券を使用した場合はその内容に応じて購入や飲食にかかった時間が経過する．

3-2-3. ゲーム終了

得した金額がスコアとして表示される．

S キーを押すと再プレイできる．

ゲームを終える場合はウィンドウの×をクリックする．

スパイラル開発履歴

＜SP1 要件定義＞

●ウィンドウを表示し，迷路画面とクーポン画面を配置する

・起動するとウィンドウが表示される

・大きさは 1000×650

・位置は（0, 0）

・閉じるボタンを押すとプログラムが終了する

・左側に迷路画面を表示，大きさは 650×650，背景色はグレー

・右側にクーポン画面を表示，大きさは 350×650，背景色は黒

●プレーヤーを矢印キーで操作できる

・最初は迷路画面の下側に上を向いて表示される．

・クーポン画面には表示されない

・プレーヤーは黄色い三角形

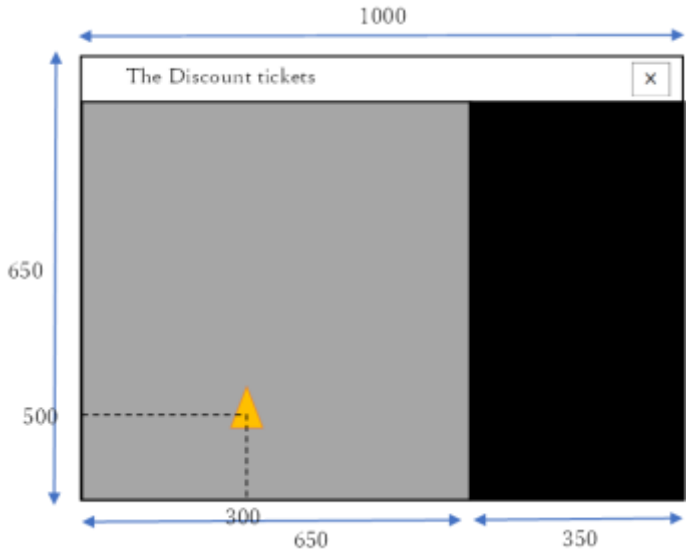
・1 度矢印キーを押すとその向きを向く

・続けて同じ方向の矢印キーを押すとその方向に進む

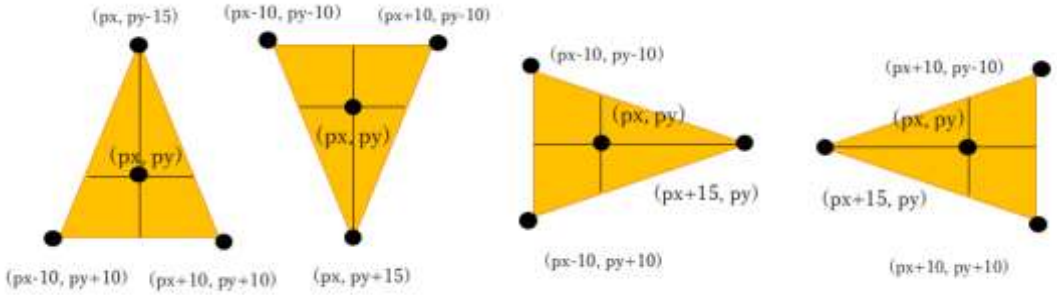
・違う方向の矢印キーを押すとその方向を向く

＜SP1 外部設計＞

●ウィンドウ



●プレーヤー



＜SP1 内部設計＞

●5 つのクラス

●TheDiscountTickets

・JFrame，メインクラス

・MazePanel を乗せる

●GameManager

・Thread

・Player を持つ

・一定時間ごとに Player を移動する

・MazePanel を再描画する

●Player

・ゲーム管理者に移動される

・画面の再描画時に描画される

・mazePanel のキーイベントを受け取るキーリスナー

●MazePanal

・JPanel

・迷路画面

・再描画時に，GameManager を描画する

●TicketaPanel

・Jpanel

・クーポン画面

●スレッド構造

●メインスレッド
・イベント処理のループ
・画面の描画
・閉じるボタンで終了

●GameManager スレッド
・ゲーム自体のループ
・時間の経過とともにプレーヤーを移動
・終了しない（無限ループ）

●キーイベントによるプレーヤーの移動
・上矢印キーに 0，下矢印キーに 1，右矢印キーに 2，左矢印キーに 3 の数字を割り当てる
・現在の方向を記憶する int 型変数 dir に格納されている値と押された矢印キーの値を比較する．同じであればプレーヤーの移動量を表す double 型変数 dx，dy を更新し，異なれば dir を更新する．

＜SP1 の進捗＞

● TheDiscountTickets

```java
/***
 *** The  Discount Tickets
 ***/

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

// TheDanmakuクラス
public class TheDiscountTickets extends JFrame {

    public TheDiscountTickets() {
        super("The Discount tickets");
        System.out.println("The Discount tickets");

        setSize(1000, 650);
        setLocation (0, 0);
        setDefaultCloseOperation (JFrame.EXIT_ON_CLOSE);

        MazePanel mp = new MazePanel();
        mp.setMaximumSize(new Dimension(650, 650)); // 迷路画面の最大値

        TicketsPanel tp = new TicketsPanel();
        tp.setMaximumSize(new Dimension(350, 650)); // クーポン画面の最大値

```

```
        GameManager gm = new GameManager (mp, tp);
        mp.setGM (gm);
        tp.setGM(gm);

        JPanel panel = new JPanel();
        panel.setLayout(new BoxLayout(panel, BoxLayout.X_AXIS));
        panel.add(mp);
        panel.add(tp);
        getContentPane().add(panel,BorderLayout.CENTER);
        gm.start();
    }

    //メイン
    Run | Debug
    public static void main(String[] args) {
        (new TheDiscountTickets()).setVisible(true);
    }
}
```

●GameManager

```
/***
 *** ゲーム管理者
 ***/

import java.awt.*;
import java.util.*;
import java.awt.event.*;

public class GameManager extends Thread {
    private MazePanel mp; // 迷路パネル
    private TicketsPanel tp; // クーポン券パネル
    private Player player;

    // 生成
    public GameManager (MazePanel mp, TicketsPanel tp) {
        this.mp = mp;
        this.tp = tp;
        player = new Player(300, 500);
        mp.addKeyListener(player);
    }

    // 実行
    public void run() {
        long t = 0;
        while(true) {
            if(t % 10 == 0) player.move();
            mp.repaint(); // 迷路を再描画
            tp.repaint(); // クーポンパネルを再描画
            try{
                sleep(2);
            } catch (InterruptedException e){}
            t++;
        }
    }

    // 描画
    public void draw (Graphics g) {
        player.draw(g); // プレイヤーを描画
    }
}
```

●Player

```java
/***
 *** プレーヤークラス
 ***/

import java.awt.*;
import java.awt.event.*;

public class Player implements KeyListener{
    private double px, py; // 位置
    private double dx, dy;
    private MazePanel mp = new MazePanel();
    private int dir;

    // 生成
    public Player (int x, int y){
        px = x;
        py = y;
        dx = 0;
        dy = 0;
        dir = 0;
    }

    // 移動
    public void move() {
        px += dx;
        py += dy;
        dx = 0;
        dy = 0;
    }

    // 描画
    public void draw (Graphics g) {
        int[] x = new int[3];
        int[] y = new int[3];
        if(dir == 0){ // 上
            x[0] = (int)px;
            x[1] = (int)px - 10;
            x[2] = (int)px + 10;
            y[0] = (int)py - 15;
            y[1] = (int)py + 10;
            y[2] = (int)py + 10;
        }else if(dir == 1){ // 下
            x[0] = (int)px;
            x[1] = (int)px - 10;
            x[2] = (int)px + 10;
            y[0] = (int)py + 15;
            y[1] = (int)py - 10;
            y[2] = (int)py - 10;
        }else if(dir == 2){ // 右
            x[0] = (int)px + 15;
            x[1] = (int)px - 10;
            x[2] = (int)px - 10;
            y[0] = (int)py;
            y[1] = (int)py + 10;
            y[2] = (int)py - 10;
        }else{ // 左
            x[0] = (int)px - 15;
            x[1] = (int)px + 10;
            x[2] = (int)px + 10;
            y[0] = (int)py;
            y[1] = (int)py + 10;
            y[2] = (int)py - 10;
        }
        g.setColor (new Color (255,255, 100));
        g.fillPolygon (x, y, 3);
        g.setColor (new Color (255,200,20) );
        g.drawPolygon (x, y, 3);
    }

    // キーリスナー
    public void keyTyped(KeyEvent ev) {}

    public void keyPressed(KeyEvent ev) {
        if(ev.getKeyCode()==KeyEvent.VK_UP){
            if(dir == 0){
                dy -= 50;
            }
            dir = 0;
        }
        if(ev.getKeyCode()==KeyEvent.VK_DOWN){
            if(dir == 1){
                dy += 50;
            }
            dir = 1;
        }
        if(ev.getKeyCode()==KeyEvent.VK_RIGHT){
            if(dir == 2){
                dx += 50;
            }
            dir = 2;
        }
        if(ev.getKeyCode()==KeyEvent.VK_LEFT){
            if(dir == 3){
                dx -= 50;
            }
            dir = 3;
        }
    }

    public void keyReleased(KeyEvent ev) {}
}
```

●MazePanel

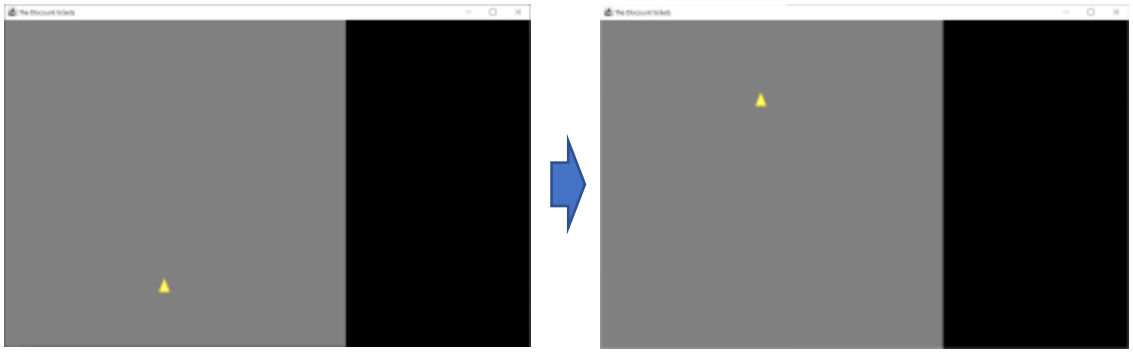
```java
/***
 *** 町の迷路
 ***/

import java.awt.*;
import javax.swing.*;
import java.awt.event.*;

public class MazePanel extends JPanel {
    private GameManager gm;

    // 生成
    public MazePanel(){
        setBackground(Color.gray);
        setFocusable(true);
    }

    // ゲーム管理者のセット
    public void setGM(GameManager gm){
        this.gm = gm;
    }

    // 描画
    public void paintComponent(Graphics g
        super.paintComponent(g);
        gm.draw(g);
    }
}
```

●TicketsPanel

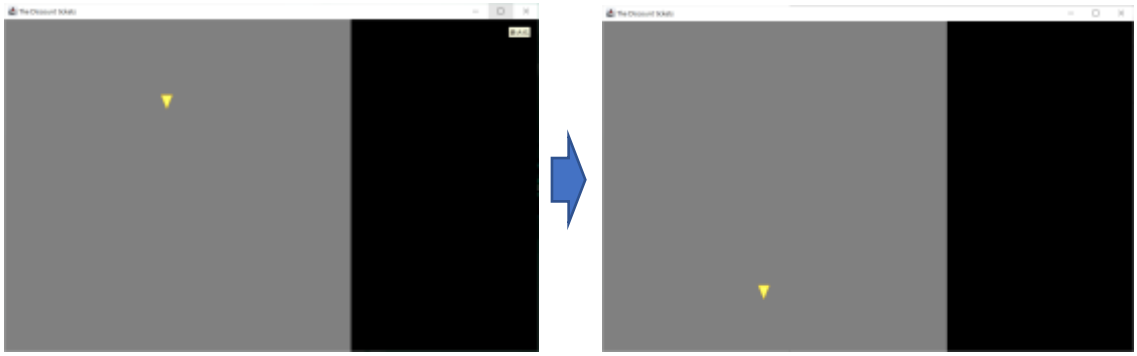
```java
/***
 *** クーポン券パネル
 ***/

import java.awt.*;
import javax.swing.*;

public class TicketsPanel extends JPanel{
    private GameManager gm;
    private MazePanel mp;

    // 生成
    public TicketsPanel(){
        setBackground(Color.black);
    }

    // ゲーム管理者のセット
    public void setGM(GameManager gm){
        this.gm = gm;
    }

    // 迷路パネルのセット
    public void setMazePanel(MazePanel mp){
        this.mp = mp;
    }

    // 描画
    public void paintComponent(Graphics g){
        super.paintComponent(g);
    }
}
```

●実行結果

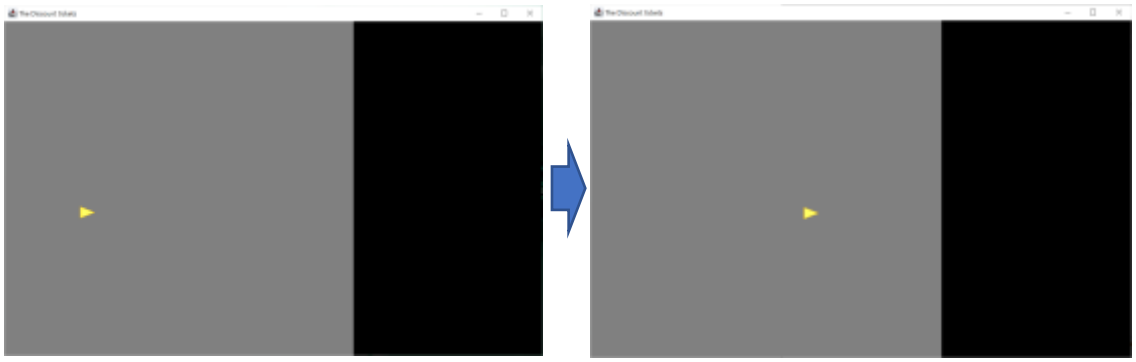
・上向き



・下向き



・右向き



・左向き

// ここまで第１１回レポートより抜粋

＜SP２要件定義＞

●SP1 からの微調整

・ウィンドウの描画エリアを 900×600 とし，全体のサイズは描画エリアに淵やタイトルバーを含めた大きさへ変更

・迷路画面のサイズを 600×600 へ変更

・クーポン画面を 600×300 へ変更

●迷路画面に壁を配置される

・表示される壁の配置は 9 種類（Maze1~9）

・最初は Maze7

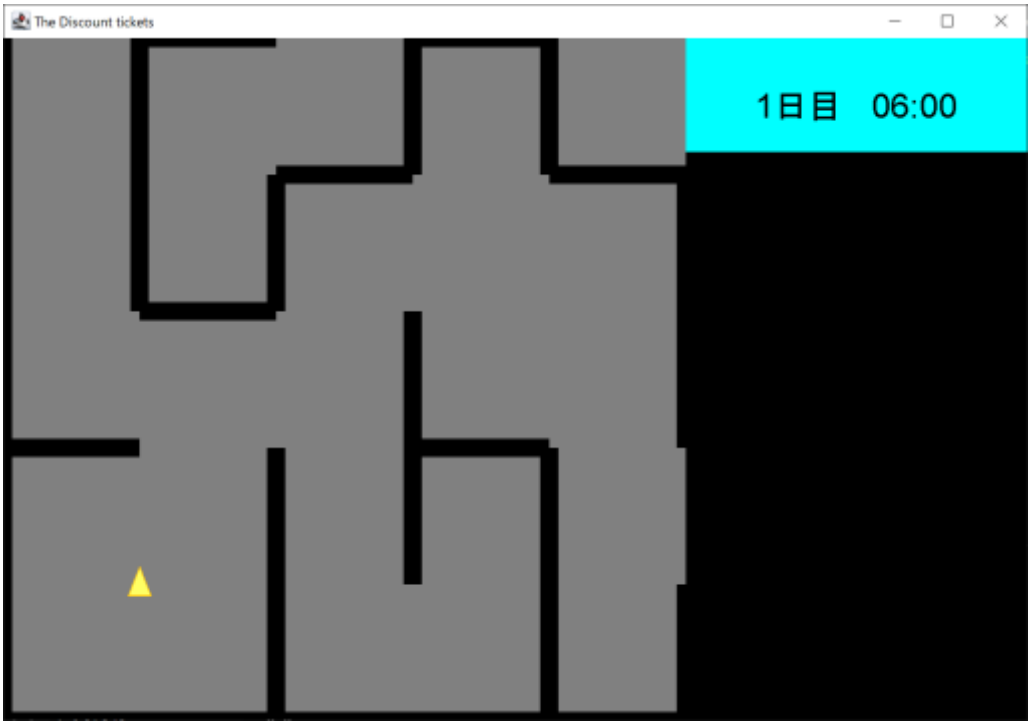
●プレーヤーは壁をすり抜けられない

●プレーヤーが迷路画面外に出ると隣の迷路へ移動する

●クーポン画面の上部に現在時刻が表示される

＜SP２外部設計＞

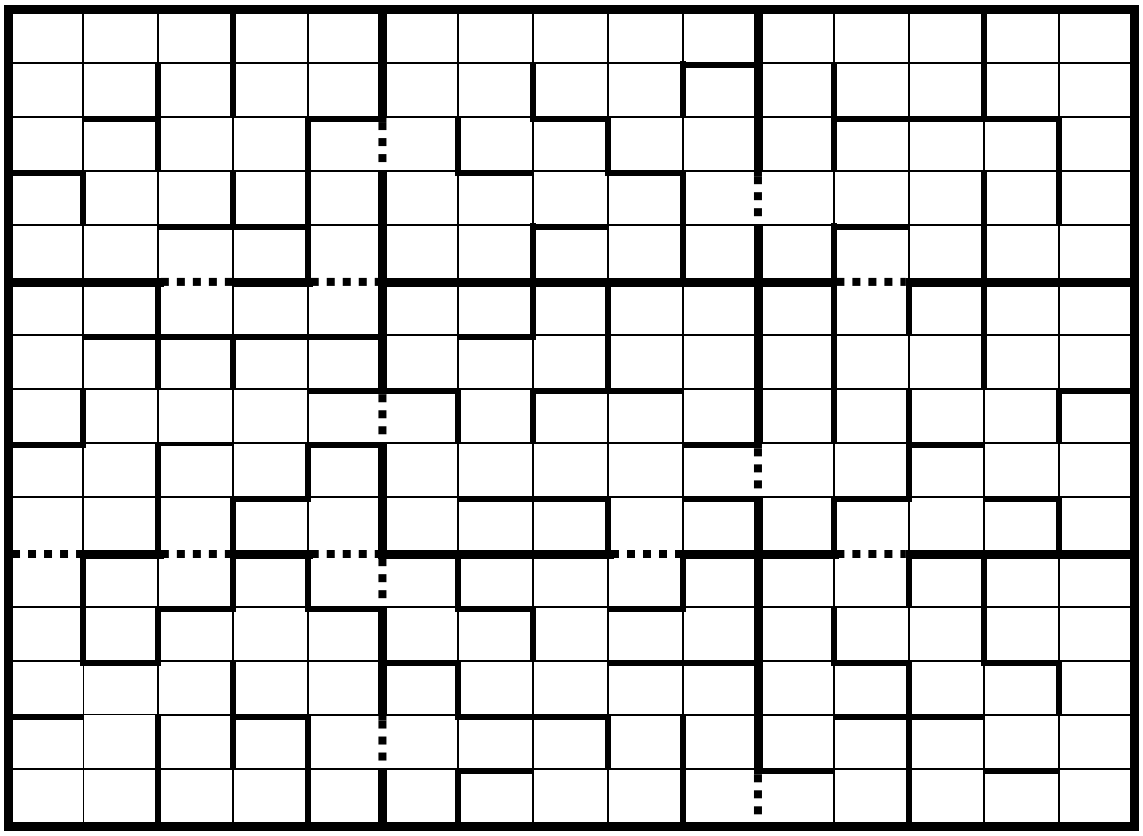
●ゲーム開始時のウィンドウ



●迷路全体図

・各迷路の番号は，電話のダイヤル配列の順に 1~9 が割り振られている．

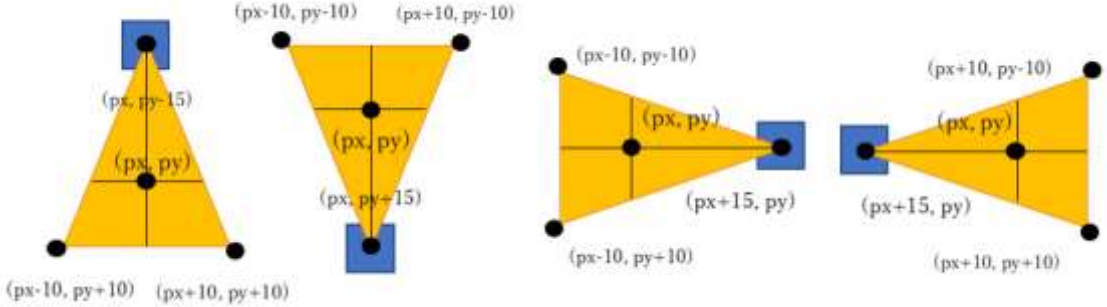
・太線は壁，破線は各迷路の出入り口



●プレーヤーの衝突判定範囲

・青い四角形が判定を行う範囲

・判定範囲は一辺 5 の正方形



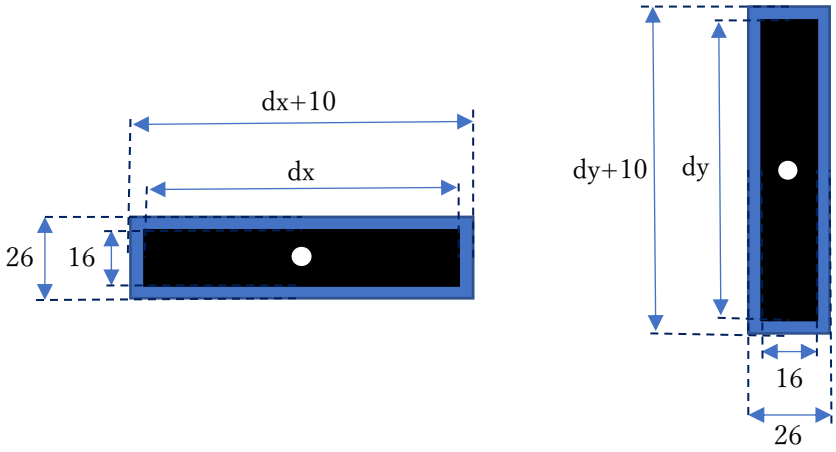
●壁とその衝突判定範囲

・壁の厚さは 16

・白点が壁の位置を表す座標であり，その壁の重心

・青い長方形が判定を行う範囲

・判定範囲は重心を共有する一辺 10 だけ大きい長方形



＜SP 2 内部設計＞

●18 個のクラス

●GameManager

・一定時間ごとに現在時刻を更新する

・一定時間ごとにプレーヤーが迷路外へ出ていないかの判定

・TicketsPanel を再描画する

●Player

・GameChar の派生クラス

・キーイベントが起こると壁との接触判定を行う

・向いている方向に合わせて，衝突判定範囲の中心座標を更新

・現在の迷路を出たとき，新たな座標を得る

●TicketsPanel

・再描画時に，GameManager と TimePanel を描画する

●TimePanel

・JPanel

・現在時刻を表示
・TicketsPanel 上に存在する
・TicketsPanel によって再描画される

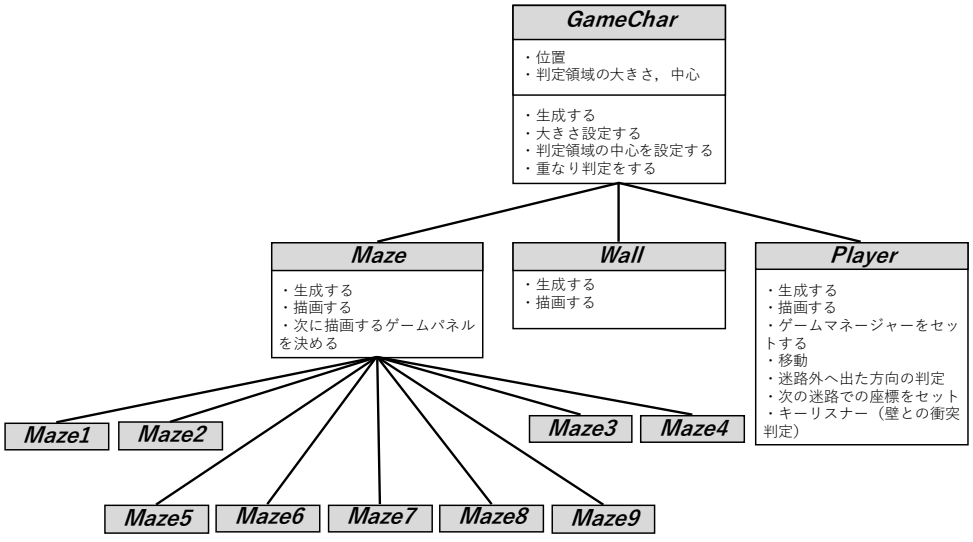
●GameChar
・ゲームオブジェクトの基底クラス

●Wall
・GameChar の派生クラス
・壁の生成と描画を行う

●Maze
・GameChar の派生クラス
・Maze1~9 の基底クラス

●Maze1~9
・Maze の派生クラス
・壁の設置を行う
・隣に位置する Maze を記憶

●クラスの階層
・迷路パネル上に登場する全てのゲームオブジェクトの基底クラス GameChar
・Maze 1 ～ 9 の基底クラス Maze
・Player、Wall も GameChar の派生クラス



●Maze1~9 が MazePanel 上に生成されると同時に, GameManager が持つ ArrayList<Wall> walls も更新

●プレーヤーが迷路外へ出ていないかの判定で行う処理
・Player クラスがもつ，どの方向の外に出たかを判定する out()を実行．Out()は外に出た方向に応じて int 型を返す．（上：0，下：1，右：2，左：3，出ていない：4）
・プレーヤーが外へ出ていれば，出た方向に応じた迷路に更新
・プレーヤーの座標を隣の迷路画面における座標へ変換し，プレーヤーがもつ GameManager を更新

●キーイベントによるプレーヤーと壁の衝突判定
・現在の方向と押された矢印キーの方向が違えば方向のみを更新する．
・現在の方向と押された矢印キーの方向が同じであれば移動量を更新する．その後現在表示されているすべての壁との接触判定を行い，いつでも接触している壁があれば移動量を 0 に再更新する．

●TicketsPanel を再描画するたびに GameManager が持つ現在時刻を文字列へ変換して TimePanel 上に描画する．

＜SP２の進捗＞

● TheDiscountTickets

```
/***
 *** The Discount Tickets
 ***/

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

// TheDiscount クラス
public class TheDiscountTickets extends JFrame {

    public TheDiscountTickets() {
        super("The Discount tickets");
        System.out.println("The Discount tickets");
```

```
        setLayout(null);
        addNotify();
        Insets insets = getInsets();
        int width = 900 + insets.left + insets.right;
        int height = 600 + insets.top + insets.bottom;
        setSize(width, height);
        setLocation (0, 0);
        setDefaultCloseOperation (JFrame.EXIT_ON_CLOSE);

        MazePanel mp = new MazePanel();
        mp.setBounds(0, 0, 600, 600);

        TicketsPanel tp = new TicketsPanel();
        tp.setBounds( 600, 0, 300, 600);

        GameManager gm = new GameManager (mp, tp);
        mp.setGM (gm);
        tp.setGM(gm);

        add(mp);
        add(tp);

        gm.start();
    }

    //メイン
    public static void main(String[] args) {
        (new TheDiscountTickets()).setVisible(true);
    }
}
```

●GameManager

```
/***
 *** ゲーム管理者
 ***/
```

```java
import java.awt.*;
import java.util.*;
import java.awt.event.*;

public class GameManager extends Thread {
    private MazePanel mp; // 迷路パネル
    private TicketsPanel tp; // クーポン券パネル
    private Maze maze;
    private Player player;
    public ArrayList<Wall> walls;
    public int hour;
    public int minute;
    public int day;

    // 生成
    public GameManager (MazePanel mp, TicketsPanel tp) {
        this.mp = mp;
        this.tp = tp;

        walls = new ArrayList<Wall>();
        maze = new Maze_7(this);
        player = new Player(120, 480);
        player.setGM(this);
        mp.addKeyListener(player);
        hour = 6;
        minute = 0;
        day = 1;
    }

    // 実行
    public void run() {
        long t = 0;
        while(true) {
            if(t % 100 == 0 && t > 500) time();
            if(t % 50 == 0) player.move();
            if(t % 200 == 0){
```

```
            int n = player.out();
            if(n != 4){
                maze = maze.next(n);
                player.out_xy(n);
                player.setGM(this);
            }
        }
        mp.repaint(); // 迷路を再描画
        tp.repaint(); // クーポンパネルを再描画
        try{
            sleep(2);
        } catch (InterruptedException e){}
        t++;
    }
}

// 迷路パネルを得る
public MazePanel getMP(){
    return mp;
}

// 現在時刻
public void time(){
    minute += 5;
    if(minute >= 60){
        minute -= 60;
        hour += 1;
        if(hour > 24){
            hour = 0;
            day += 1;
        }
    }
}

// 描画
public void draw (Graphics g) {
```

```
        player.draw(g); // プレイヤーを描画
        maze.draw(g);   // 迷路を描画
    }
}
```

●Player

```
/***
 *** プレーヤークラス
 ***/

import java.awt.*;
import java.util.*;
import java.awt.event.*;

public class Player extends GameChar implements KeyListener{

    private int dx, dy;
    private MazePanel mp;
    private GameManager gm;
    private ArrayList<Wall> walls;
    private int dir;
    private int a = 20;

    // 生成
    public Player (int x, int y){
        px = x;
        py = y;
        dx = 0;
        dy = 0;
        dir = 0;
        setSize(5, 5);
        setCenter(px, py - 15);
        walls = new ArrayList<Wall>();
    }
```

```
    public void setGM(GameManager gm){
        this.gm = gm;
        this.walls = gm.walls;
    }

    // 移動
    public void move() {
        System.out.println(dx +" "+ dy);
        px += dx;
        py += dy;
        dx = 0;
        dy = 0;
    }

    // パネル外判定
    public int out(){
        if(py < 0) return 0;
        if(py > 600) return 1;
        if(px > 600) return 2;
        if(px < 0) return 3;
        else return 4;
    }

    // プレイヤーがウィンドウから出た後の座標
    public void out_xy(int n){
        switch(n){
            case 0 : py += 600; break;
            case 1 : py -= 600; break;
            case 2 : px -= 600; break;
            case 3 : px += 600; break;
            default : break;
        }
    }

    // 描画
    public void draw (Graphics g) {
```

```
        int[] x = new int[3];
        int[] y = new int[3];
        if(dir == 0){ // 上
            x[0] = (int)px;
            x[1] = (int)px - 10;
            x[2] = (int)px + 10;
            y[0] = (int)py - 15;
            y[1] = (int)py + 10;
            y[2] = (int)py + 10;
        }else if(dir == 1){ // 下
            x[0] = (int)px;
            x[1] = (int)px - 10;
            x[2] = (int)px + 10;
            y[0] = (int)py + 15;
            y[1] = (int)py - 10;
            y[2] = (int)py - 10;
        }else if(dir == 2){ // 右
            x[0] = (int)px + 15;
            x[1] = (int)px - 10;
            x[2] = (int)px - 10;
            y[0] = (int)py;
            y[1] = (int)py + 10;
            y[2] = (int)py - 10;
        }else{ // 左
            x[0] = (int)px - 15;
            x[1] = (int)px + 10;
            x[2] = (int)px + 10;
            y[0] = (int)py;
            y[1] = (int)py + 10;
            y[2] = (int)py - 10;
        }
        setCenter(x[0], y[0]);
        g.setColor (new Color (255,255, 100)); // 明るい黄色で
        g.fillPolygon (x, y, 3);               // 塗りつぶして
        g.setColor (new Color (255,200,20) );  // 暗い黄色で
        g.drawPolygon (x, y, 3);               // 枠を描く
```

```
    }

    // キーリスナー
    public void keyTyped(KeyEvent ev) {}

    public void keyPressed(KeyEvent ev) {
        if(ev.getKeyCode()==KeyEvent.VK_UP){
            if(dir == 0){
                dy = -1 * a;
                for(Wall w:walls){
                    if(overlap(w) == true){
                        dy = 0;
                        break;
                    }
                }
            }
            dir = 0;
        }
        if(ev.getKeyCode()==KeyEvent.VK_DOWN){
            if(dir == 1){
                dy = a;
                for(Wall w:walls){
                    if(overlap(w) == true){
                    dy = 0;
                    break;
                    }
                }
            }
            dir = 1;
        }
        if(ev.getKeyCode()==KeyEvent.VK_RIGHT){
            if(dir == 2){
                dx = a;
                for(Wall w:walls){
                    if(overlap(w) == true){
                        dx = 0;
```

```
                    break;
                }
            }
        }
        dir = 2;
    }
    if(ev.getKeyCode()==KeyEvent.VK_LEFT){
        if(dir == 3){
            dx = -1 * a;
            for(Wall w:walls){
                if(overlap(w) == true){
                    dx = 0;
                    break;
                }
            }
        }
        dir = 3;
    }
  }

  public void keyReleased(KeyEvent ev) {}
}
```

●MazePanel

```
/***
 *** 町の迷路
 ***/

import java.awt.*;
import javax.swing.*;
import java.awt.event.*;

public class MazePanel extends JPanel {
    private GameManager gm;

    // 生成
```

```
    public MazePanel(){
        setBackground(Color.gray);
        setFocusable(true);
    }

    // ゲーム管理者のセット
    public void setGM(GameManager gm){
        this.gm = gm;
    }

    public void paintComponent(Graphics g){
        super.paintComponent(g);
        gm.draw(g);
    }
}
```

●TicketsPanel

```
/***
 *** クーポン券パネル
 ***/

import java.awt.*;
import javax.swing.*;

public class TicketsPanel extends JPanel{
    private GameManager gm;
    private MazePanel mp;
    private TimePanel timePanel; // 現在時刻表示欄

    // 生成
    public TicketsPanel(){
        setBackground(Color.black);
    }

    // ゲーム管理者のセット
    public void setGM(GameManager gm){
```

```
        this.gm = gm;
        setLayout(null);
        timePanel = new TimePanel(gm);
        timePanel.setBounds(0, 0, 300, 100);
        add(timePanel);
    }

    // 迷路パネルのセット
    public void setMazePanel(MazePanel mp){
        this.mp = mp;
    }

    // 描画
    public void paintComponent(Graphics g){
        super.paintComponent(g);
        timePanel.paintComponents(g);

    }
}
```

●TimePanel

```
/***
 *** 時間表示パネル
 ***/

import java.awt.*;
import javax.swing.*;

public class TimePanel extends JPanel{
    private GameManager gm;

    // 生成
    public TimePanel(GameManager gm){
        this.gm = gm;
       setBackground(Color.cyan);
    }
```

```
    // 時刻描画描画
    public void draw(Graphics g){
        String s_hour = String.valueOf(gm.hour);
        String s_minute = String.valueOf(gm.minute);
        String s_day = String.valueOf(gm.day);
        String z1 = "", z2 = "";
        if(gm.hour < 10){
            z1 = "0";
        }
        if(gm.minute < 10){
            z2 = "0";
        }
        g.setColor(Color.black);
        g.setFont(new Font("GM Gothic", Font.PLAIN, 30));
        g.drawString(s_day + "日目   " + z1 + s_hour + ":" + z2 +
                     s_minute, 60, 70);
    }

    public void paintComponent(Graphics g){
        super.paintComponent(g);
        draw(g);
    }
}
```

●GameChar

```
/***
 *** ゲームキャラクター
 ***/

import java.awt.*;

public abstract class GameChar {
  protected int px,py; // 位置
  private int cx, cy; // 判定領域の中心
```

```
private int cw,ch; // 判定領域の大きさ

// 生成
public GameChar(){
  this(0,0);
}
public GameChar(int x, int y){
  px=x; py=y;
  cx=x; cy=y;
  cw=ch=0; // 最初の大きさは 0x0
}

// 判定領域の中心の設定
public void setCenter(int dx, int dy){
  cx = dx;
  cy = dy;
}

// 大きさ設定
public void setSize(int w, int h){
  cw=w;
  ch=h;
}

// 位置を得る
public double getX(){ return px; }
public double getY(){ return py; }

// 重なり判定
public boolean overlap(GameChar c){
  double x11=cx-cw/2,y11=cy-ch/2;
  double x12=cx+cw/2,y12=cy+ch/2;
  double x21=c.cx-c.cw/2,y21=c.cy-c.ch/2;
  double x22=c.cx+c.cw/2,y22=c.cy+c.ch/2;
  return
    (((x11<=x21 && x21<x12)||(x21<=x11 && x11<x22)) &&
```

```
        ((y11<=y21 && y21<y12)||(y21<=y11 && y11<y22)));
    }
}
```

●Wall

```
/***
 *** 壁
 ***/
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;

public class Wall extends GameChar{
    private int dx, dy;

    // 生成
    public Wall(int x, int y, int dx, int dy){
        this.px = x;
        this.py = y;
        this.dx = dx;
        this.dy = dy;
        setCenter(x, y);
        setSize(dx + 10, dy + 10);
    }

    // 描画
    public void draw(Graphics g){
        g.setColor(Color.BLACK);
        g.fillRect((int)(px - dx/2), (int)(py - dy/2), (int)dx, (int)dy);
    }
}
```

●Maze

```
/***
 *** 迷路の基底クラス
 ***/
```

```java
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.util.ArrayList;

public abstract class Maze{
    private GameManager gm;

    // 生成
    public Maze(GameManager gm){
        this.gm = gm;
    }

    // 描画
    public abstract void draw(Graphics g);

    // 次のゲームパネル
    public abstract Maze next(int n);

}
```

●Maze_1

```java
/***
 *** 迷路1
 ***/

import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.util.ArrayList;

public class Maze_1 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[15];
```

```
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;

    // 生成
    public Maze_1(GameManager gm){
        super(gm);
        this.gm = gm;
        wall[0] = new Wall(300, 0, 600, a);
        wall[1] = new Wall(0, 300, a, 600);
        wall[2] = new Wall(120, 600, 240, a);
        wall[3] = new Wall(420, 600, 120, a);

        wall[4] = new Wall(600, 120, a, 240);
        wall[5] = new Wall(600, 480, a, 240);

        wall[6] = new Wall(180, 240, 120, a);
        wall[7] = new Wall(420, 240, 120, a);
        wall[8] = new Wall(60, 360, 120, a);
        wall[9] = new Wall(360, 480, 240, a);

        wall[10] = new Wall(360, 120, a, 240);
        wall[11] = new Wall(240, 240, a, 240);
        wall[12] = new Wall(120, 420, a, 120);
        wall[13] = new Wall(360, 420, a, 120);
        wall[14] = new Wall(480, 420, a, 360);

        for(int i= 0; i < 15; i++){
            walls.add(wall[i]);
        }

        gm.walls = this.walls;

        /*
        shop
        */
    }
```

```
    // 描画
    public void draw(Graphics g){
        for(Wall w:walls) w.draw(g);
    }

    // 次のゲームパネル
    public Maze next(int n){
        switch(n){
            case 0 : // なし
            case 1 : return new Maze_4(gm);
            case 2 : return new Maze_2(gm);
            case 3 : // なし
            default : return this;
        }
    }
}
```

●Maze_2

```
/***
 *** 迷路2
 ***/

import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.util.ArrayList;

public class Maze_2 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[17];
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;

    // 生成
    public Maze_2(GameManager gm){
```

```
        super(gm);
        this.gm = gm;
        wall[0] = new Wall(300, 0, 600, a);
        wall[1] = new Wall(0, 120, a, 240);
        wall[2] = new Wall(0, 480, a, 240);
        wall[3] = new Wall(300, 600, 600, a);
        wall[4] = new Wall(600, 180, a, 360);
        wall[5] = new Wall(600, 540, a, 120);

        wall[6] = new Wall(540, 120, 120, a);
        wall[7] = new Wall(300, 240, 120, a);
        wall[8] = new Wall(420, 360, 120, a);
        wall[9] = new Wall(180, 360, 120, a);
        wall[10] = new Wall(300, 480, 120, a);

        wall[11] = new Wall(240, 180, a, 120);
        wall[12] = new Wall(480, 180, a, 120);
        wall[13] = new Wall(120, 300, a, 120);
        wall[14] = new Wall(360, 300, a, 120);
        wall[15] = new Wall(480, 480, a, 240);
        wall[16] = new Wall(240, 540, a, 120);

        for(int i= 0; i < 17; i++){
            walls.add(wall[i]);
        }

        gm.walls = this.walls;

        /*
        shop
        */
    }

    // 描画
    public void draw(Graphics g){
        for(Wall w:walls) w.draw(g);
```

```
    }

    // 次のゲームパネル
    public Maze next(int n){
        switch(n){
            case 0 : // 無し
            case 1 : return new Maze_5(gm);
            case 2 : return new Maze_3(gm);
            case 3 : return new Maze_1(gm);
            default : return this;
        }
    }
}
```

●Maze_3

```
/***
 *** 迷路3
 ***/

import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.util.ArrayList;

public class Maze_3 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[13];
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;

    // 生成
    public Maze_3(GameManager gm){
        super(gm);
        this.gm = gm;
        wall[0] = new Wall(300, 0, 600, a);
        wall[1] = new Wall(0, 180, a, 360);
```

```
        wall[2] = new Wall(0, 540, a, 120);
        wall[3] = new Wall(60, 600, 120, a);
        wall[4] = new Wall(420, 600, 360, a);
        wall[5] = new Wall(600, 300, a, 600);

        wall[6] = new Wall(300, 240, 360, a);
        wall[7] = new Wall(180, 480, 120, a);

        wall[8] = new Wall(360, 120, a, 240);
        wall[9] = new Wall(120, 240, a, 240);
        wall[10] = new Wall(480, 360, a, 240);
        wall[11] = new Wall(360, 480, a, 240);
        wall[12] = new Wall(120, 540, a, 120);

        for(int i= 0; i < 13; i++){
            walls.add(wall[i]);
        }

        gm.walls = this.walls;

        /*
        shop
        */
    }

    // 描画
    public void draw(Graphics g){
        for(Wall w:walls) w.draw(g);
    }

    // 次のゲームパネル
    public Maze next(int n){
        switch(n){
            case 0 : // なし
            case 1 : return new Maze_6(gm);
            case 2 : // なし
```

```
            case 3 : return new Maze_2(gm);
            default : return this;
        }
    }
}
```

●Maze_4

```
/***
 *** 迷路4
 ***/

import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.util.ArrayList;

public class Maze_4 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[19];
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;

    // 生成
    public Maze_4(GameManager gm){
        super(gm);
        this.gm = gm;
        wall[0] = new Wall(120, 0, 240, a);
        wall[1] = new Wall(420, 0, 120, a);
        wall[2] = new Wall(0, 300, a, 600);
        wall[3] = new Wall(180, 600, 120, a);
        wall[4] = new Wall(420, 600, 120, a);
        wall[5] = new Wall(600, 120, a, 240);
        wall[6] = new Wall(600, 480, a, 240);

        wall[7] = new Wall(360, 120, 480, a);
        wall[8] = new Wall(540, 240, 120, a);
```

```
        wall[9] = new Wall(60, 360, 120, a);
        wall[10] = new Wall(300, 360, 120, a);
        wall[11] = new Wall(540, 360, 120, a);
        wall[12] = new Wall(420, 480, 120, a);

        wall[13] = new Wall(240, 60, a, 120);
        wall[14] = new Wall(360, 180, a, 120);
        wall[15] = new Wall(120, 180, a, 120);
        wall[16] = new Wall(480, 420, a, 120);
        wall[17] = new Wall(240, 480, a, 240);
        wall[18] = new Wall(360, 540, a, 120);


        for(int i= 0; i < 19; i++){
            walls.add(wall[i]);
        }

        gm.walls = this.walls;

        /*
        shop
        */
    }

    // 描画
    public void draw(Graphics g){
        for(Wall w:walls) w.draw(g);
    }

    // 次のゲームパネル
    public Maze next(int n){
        switch(n){
            case 0 : return new Maze_1(gm);
            case 1 : return new Maze_7(gm);
            case 2 : return new Maze_5(gm);
            case 3 : // なし
```

```
            default : return this;
        }
    }
}
```

●Maze_5

```
/***
 *** 迷路5
 ***/

import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.util.ArrayList;

public class Maze_5 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[18];
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;

    // 生成
    public Maze_5(GameManager gm){
        super(gm);
        this.gm = gm;
        wall[0] = new Wall(300, 0, 600, a);
        wall[1] = new Wall(0, 120, a, 240);
        wall[2] = new Wall(0, 480, a, 240);
        wall[3] = new Wall(180, 600, 360, a);
        wall[4] = new Wall(540, 600, 120, a);
        wall[5] = new Wall(600, 180, a, 360);
        wall[6] = new Wall(600, 540, a, 120);

        wall[7] = new Wall(180, 120, 120, a);
        wall[8] = new Wall(360, 240, 240, a);
        wall[9] = new Wall(60, 240, 120, a);
```

```
        wall[10] = new Wall(540, 360, 120, a);
        wall[11] = new Wall(240, 480, 240, a);
        wall[12] = new Wall(540, 480, 120, a);

        wall[13] = new Wall(240, 60, a, 120);
        wall[14] = new Wall(360, 120, a, 240);
        wall[15] = new Wall(120, 300, a, 120);
        wall[16] = new Wall(240, 300, a, 120);
        wall[17] = new Wall(360, 540, a, 120);

        for(int i= 0; i < 18; i++){
            walls.add(wall[i]);
        }

        gm.walls = this.walls;

        /*
        shop
        */
    }

    // 描画
    public void draw(Graphics g){
        for(Wall w:walls) w.draw(g);
    }

    // 次のゲームパネル
    public Maze next(int n){
        switch(n){
            case 0 : return new Maze_2(gm);
            case 1 : return new Maze_8(gm);
            case 2 : return new Maze_6(gm);
            case 3 : return new Maze_4(gm);
            default : return this;
        }
    }
```

```
}
```

●Maze_6

```
/***
 *** 迷路6
 ***/

import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.util.ArrayList;

public class Maze_6 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[19];
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;

    // 生成
    public Maze_6(GameManager gm){
        super(gm);
        this.gm = gm;
        wall[0] = new Wall(0, 180, a, 360);
        wall[1] = new Wall(0, 540, a, 120);
        wall[2] = new Wall(60, 600, 120, a);
        wall[3] = new Wall(420, 600, 360, a);
        wall[4] = new Wall(600, 300, a, 600);
        wall[5] = new Wall(420, 0, 360, a);
        wall[6] = new Wall(60, 0, 120, a);

        wall[7] = new Wall(540, 240, 120, a);
        wall[8] = new Wall(300, 360, 120, a);
        wall[9] = new Wall(180, 480, 120, a);
        wall[10] = new Wall(420, 480, 120, a);

        wall[11] = new Wall(120, 180, a, 360);
```

```
        wall[12] = new Wall(120, 540, a, 120);
        wall[13] = new Wall(240, 60, a, 120);
        wall[14] = new Wall(360, 120, a, 240);
        wall[15] = new Wall(480, 300, a, 120);
        wall[16] = new Wall(240, 360, a, 240);
        wall[17] = new Wall(120, 540, a, 120);
        wall[18] = new Wall(480, 540, a, 120);

        for(int i= 0; i < 19; i++){
            walls.add(wall[i]);
        }

        gm.walls = this.walls;

        /*
        shop
        */
    }

    // 描画
    public void draw(Graphics g){
        for(Wall w:walls) w.draw(g);
    }

    // 次のゲームパネル
    public Maze next(int n){
        switch(n){
            case 0 : return new Maze_3(gm);
            case 1 : return new Maze_9(gm);
            case 2 : // なし
            case 3 : return new Maze_5(gm);
            default : return this;
        }
    }
}
```

●Maze_7

```java
/***
 *** 迷路7
 ***/

import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.util.ArrayList;

public class Maze_7 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[18];
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;

    // 生成
    public Maze_7(GameManager gm){
        super(gm);
        this.gm = gm;
        wall[0] = new Wall(0, 300, a, 600);
        wall[1] = new Wall(300, 600, 600, a);
        wall[2] = new Wall(600, 540, a, 120);
        wall[3] = new Wall(600, 240, a, 240);
        wall[4] = new Wall(420, 0, 120, a);
        wall[5] = new Wall(180, 0, 120, a);

        wall[6] = new Wall(300, 120, 120, a);
        wall[7] = new Wall(540, 120, 120, a);
        wall[8] = new Wall(180, 240, 120, a);
        wall[9] = new Wall(60, 360, 120, a);
        wall[10] = new Wall(420, 360, 120, a);

        wall[11] = new Wall(120, 120, a, 240);
        wall[12] = new Wall(360, 60, a, 120);
        wall[13] = new Wall(480, 60, a, 120);
        wall[14] = new Wall(240, 180, a, 120);
```

```
        wall[15] = new Wall(360, 360, a, 240);
        wall[16] = new Wall(240, 480, a, 240);
        wall[17] = new Wall(480, 480, a, 240);

        for(int i= 0; i < 18; i++){
            walls.add(wall[i]);
        }

        gm.walls = this.walls;

        /*
        shop
        */
    }

     // 描画
     public void draw(Graphics g){
        for(Wall w:walls) w.draw(g);
    }

     // 次のゲームパネル
    public Maze next(int n){
        switch(n){
            case 0 : return new Maze_4(gm);
            case 1 : // なし
            case 2 : return new Maze_8(gm);
            case 3 : // なし
            default : return this;
        }
    }
}
```

●Maze_8

```
/***
 *** 迷路8
 ***/
```

```java
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.util.ArrayList;

public class Maze_8 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[19];
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;

    // 生成
    public Maze_8(GameManager gm){
        super(gm);
        this.gm = gm;
        wall[0] = new Wall(0, 240, a, 240);
        wall[1] = new Wall(0, 540, a, 120);
        wall[2] = new Wall(300, 600, 600, a);
        wall[3] = new Wall(600, 240, a, 480);
        wall[4] = new Wall(180, 0, 360, a);
        wall[5] = new Wall(540, 0, 120, a);

        wall[6] = new Wall(180, 120, 120, a);
        wall[7] = new Wall(420, 120, 120, a);
        wall[8] = new Wall(60, 240, 120, a);
        wall[9] = new Wall(480, 240, 240, a);
        wall[10] = new Wall(240, 360, 240, a);
        wall[11] = new Wall(180, 480, 120, a);

        wall[12] = new Wall(120, 60, a, 120);
        wall[13] = new Wall(480, 60, a, 120);
        wall[14] = new Wall(240, 180, a, 120);
        wall[15] = new Wall(120, 300, a, 120);
        wall[16] = new Wall(360, 420, a, 120);
        wall[17] = new Wall(480, 480, a, 240);
```

```
        wall[18] = new Wall(120, 540, a, 120);

        for(int i= 0; i < 19; i++){
            walls.add(wall[i]);
        }

        gm.walls = this.walls;

        /*
        shop
        */
    }

    // 描画
    public void draw(Graphics g){
        for(Wall w:walls) w.draw(g);
    }

    // 次のゲームパネル
    public Maze next(int n){
        switch(n){
            case 0 : return new Maze_5(gm);
            case 1 : // なし
            case 2 : return new Maze_9(gm);
            case 3 : return new Maze_7(gm);
            default : return this;
        }
    }
}
```

●Maze_9

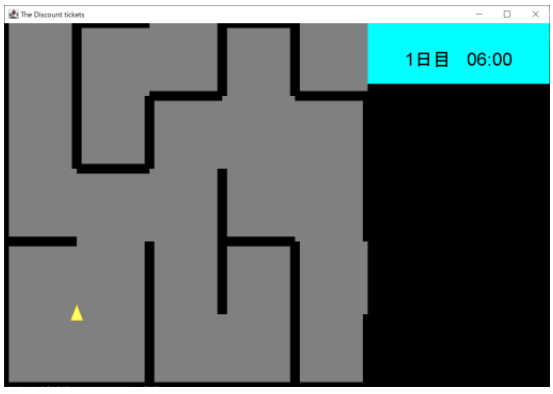
```
/***
 *** 迷路9
 ***/

import java.awt.*;
```

```java
import javax.swing.*;
import java.awt.event.*;
import java.util.ArrayList;

public class Maze_9 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[16];
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;

    // 生成
    public Maze_9(GameManager gm){
        super(gm);
        this.gm = gm;
        wall[0] = new Wall(0, 240, a, 480);
        wall[1] = new Wall(300, 600, 600, a);
        wall[2] = new Wall(600, 300, a, 600);
        wall[3] = new Wall(420, 0, 360, a);
        wall[4] = new Wall(60, 0, 120, a);

        wall[5] = new Wall(180, 240, 120, a);
        wall[6] = new Wall(420, 240, 120, a);
        wall[7] = new Wall(240, 360, 240, a);
        wall[8] = new Wall(60, 480, 120, a);
        wall[9] = new Wall(420, 480, 120, a);

        wall[10] = new Wall(240, 60, a, 120);
        wall[11] = new Wall(360, 120, a, 240);
        wall[12] = new Wall(120, 180, a, 120);
        wall[13] = new Wall(480, 300, a, 120);
        wall[14] = new Wall(240, 420, a, 360);
        wall[15] = new Wall(360, 540, a, 120);


        for(int i= 0; i < 16; i++){
            walls.add(wall[i]);
```

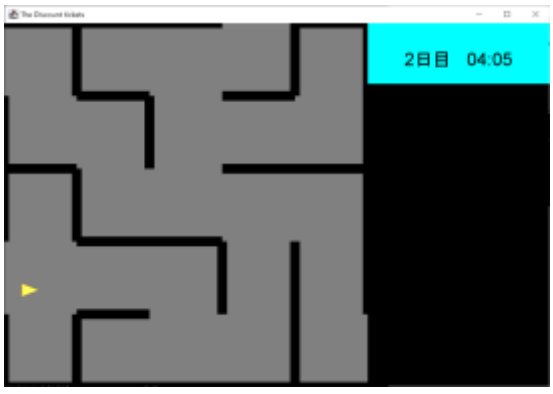
```
        }

        gm.walls = this.walls;

        /*
        shop
        */
    }

    // 描画
    public void draw(Graphics g){
        for(Wall w:walls) w.draw(g);
    }

    // 次のゲームパネル
    public Maze next(int n){
        switch(n){
            case 0 : return new Maze_6(gm);
            case 1 : // なし
            case 2 : // なし
            case 3 : return new Maze_8(gm);
            default : return this;
        }
    }
}
```

●実行結果

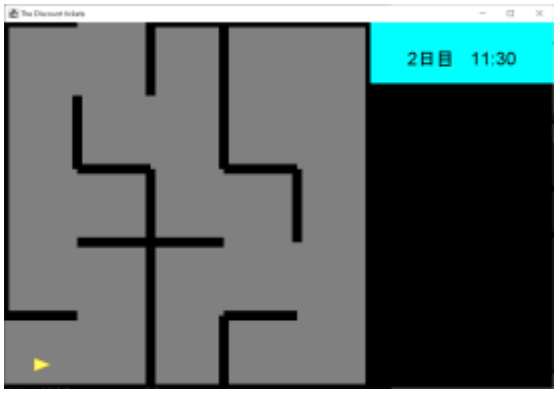
・Maze7(スタート地点)



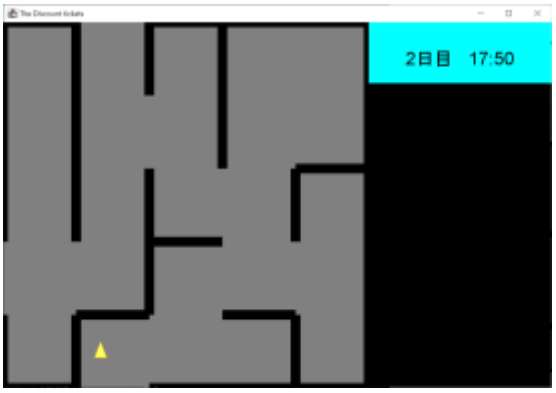
・Maze8(Maze7 から Maze8 へ)



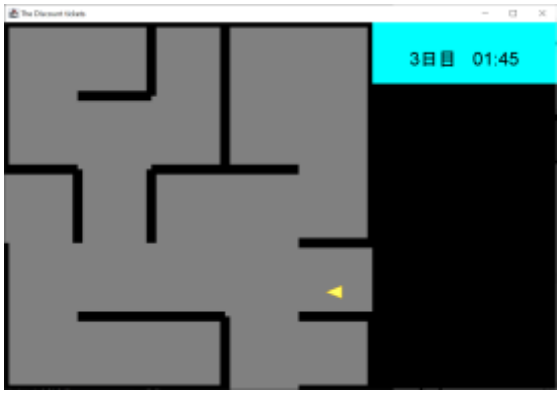
・Maze9(Maze8 から Maze9 へ)



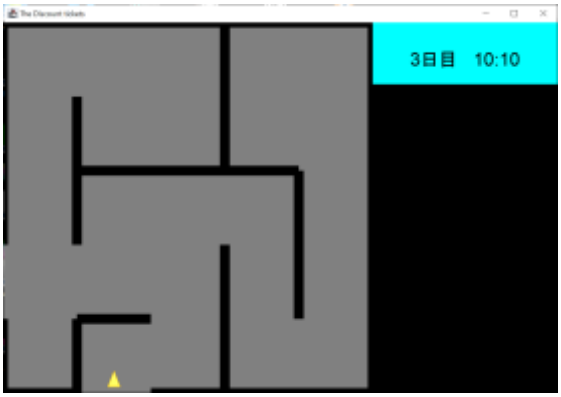
・Maze6(Maze9 から Maze6 へ)



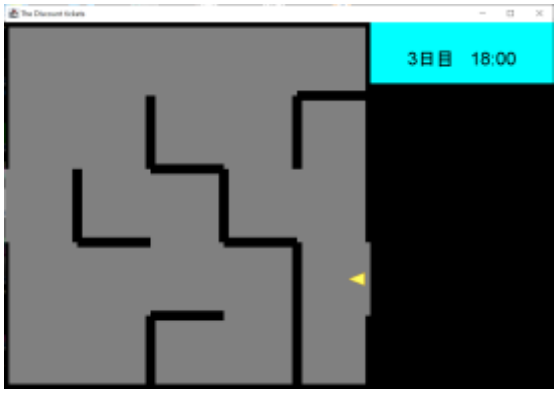
・Maze5(Maze6 から Maze5 へ)



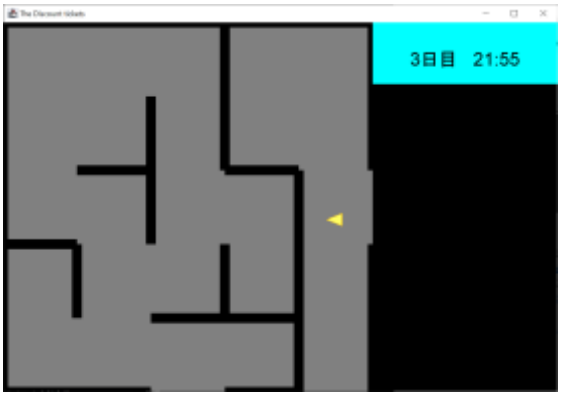
・Maze3(Maze6 から Maze3 へ)



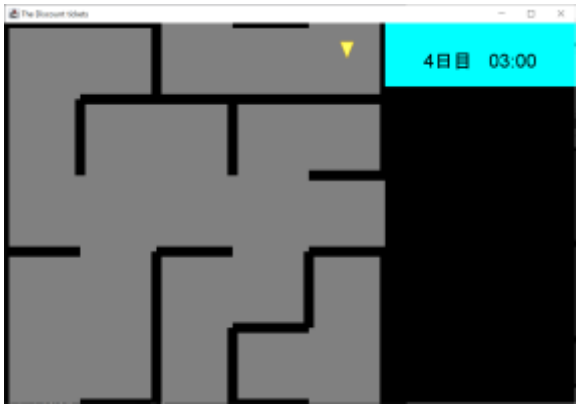
・Maze2(Maze3 から Maze2 へ)



・Maze1(Maze2 から Maze1 へ)



・Maze4(Maze1 から Maze4 へ)



＜SP３要件定義＞

●迷路画面に店が配置される

・表示される店は 11 種類

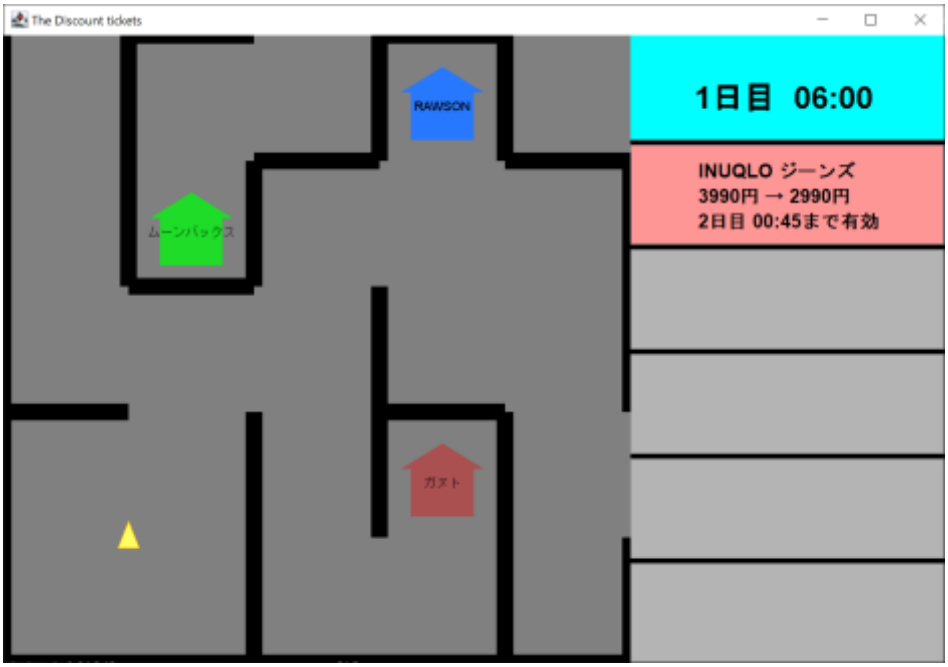
●クーポン画面の時刻表示欄の下部にクーポン券が JButton として配置される.

・クリックしたら消える

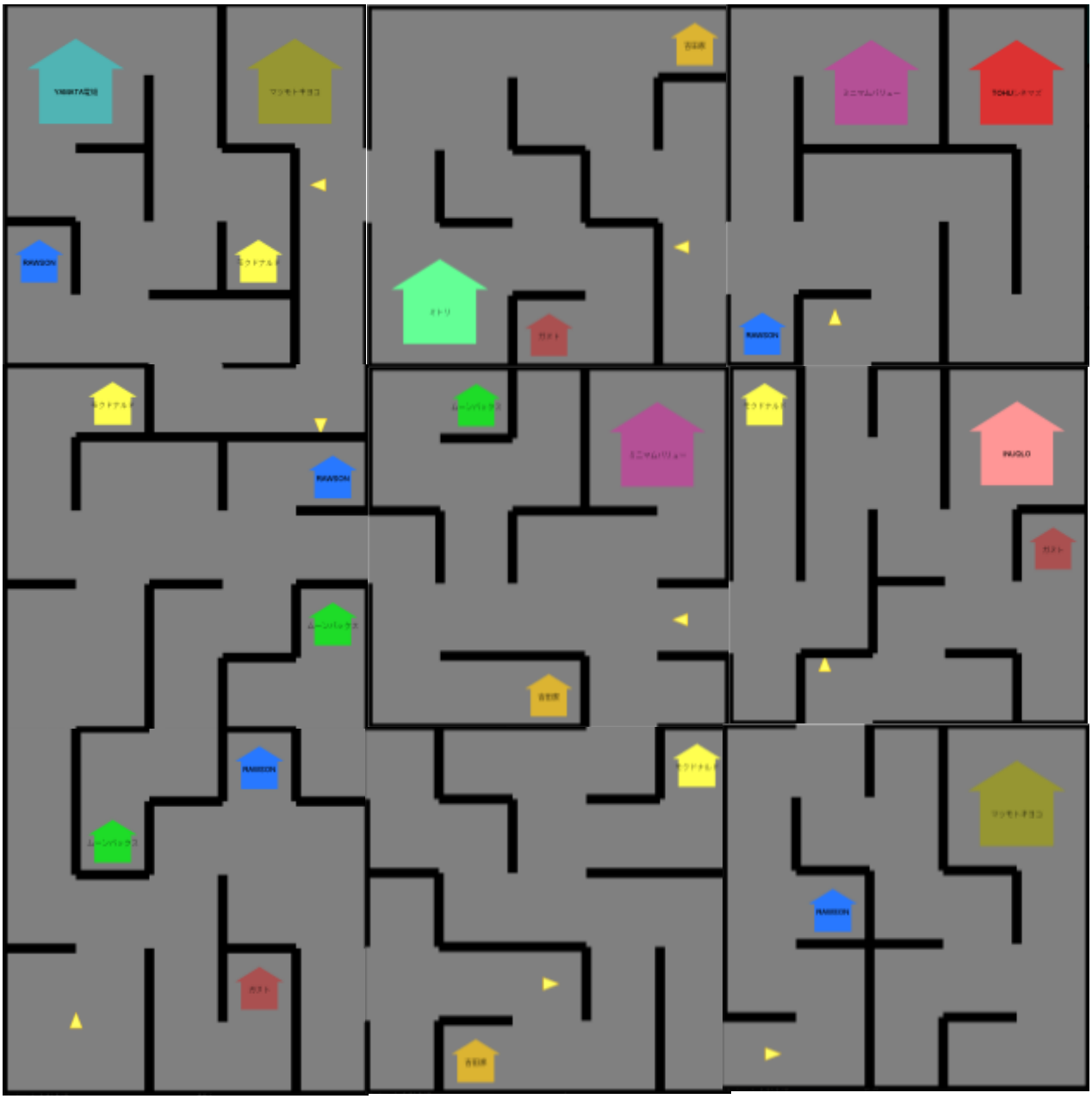
・有効期限が過ぎたら消える

＜SP３外部設計＞

●ゲーム開始時のウィンドウの例



●迷路全体図



＜SP３内部設計＞

●21 個のクラス

●GameManager

・一定時間ごとに現在時刻を更新するとともに，クーポン券の残りの有効期限を減らす

・一定時間ごとにクーポンを生成

●TicketsPanel

・再描画時に，CouponPanel と TimePanel を描画する

●CouponPanel

・生成時に 5 つのデフォルトの JButton を，配列を用いて設置

・アクションコマンドを配列の番号にセット

●Maze1~9

・店が設置される

●Shop

・GameChar クラスを継承

・店を生成する

・店によって色や大きさが異なる

●Coupon

・クーポン券（内部情報）を生成する.

・内部情報として，定価，割引価格，有効期限，int 型の使用可能店舗，String 型の使用可能店舗名，商品名，クーポン券に表示するテキストを所持する.

・生成されるクーポン券はランダムに決定

●デフォルトボタン

・フォント（色：黒，　フォント名：DIALOG，　スタイル：BOLD，　サイズ 18）

・テキストは無し

・背景色　Color(180, 180, 180)

・サイズ　(300, 95)

・Enable　false

●クーポン券ボタン

・フォント(デフォルトボタンのまま)

・テキスト　("<html>" +(店舗名) + "　" + (商品名) + "
" + (定価)+ "円　→　" + (割引価格)+ "円
" +(有効期限を示す文字列))

・背景色　店の色と同じ

・サイズ　(デフォルトボタンのまま)

・Enable　true

●make_Coupon_Button()
・ボタンの配列の中からデフォルトボタン JButton button[i]を探す
・新たなクーポンを生成し，Coupon 型 coupon[i]に代入
・coupon[i]の所持する情報から，button[i]をクーポン券ボタンに変更
・一度クーポン券ボタンを作ったら，それ以降の配列内にデフォルトボタンがあっても処理は行わない.

●アクションイベントによるクーポン券の利用
・getActionCommand();メソッドを用いて押されたボタンの配列の番号を String 型で取得する
・取得した番号を int 型へ変換する
・取得した番号に対応するボタンをデフォルトに戻す.
・取得した番号に対応するボタンより配列の後ろにあるボタンのフォーカスを false にする.

＜SP 3 の進捗＞

●TheDiscountTickets

```
/***
 *** The Discount Tickets
 ***/

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

// TheDanmaku クラス
public class TheDiscountTickets extends JFrame {

    public TheDiscountTickets() {
        super("The Discount tickets");
        System.out.println("The Discount tickets");
        setLayout(null);
        addNotify();
        Insets insets = getInsets();
        int width = 900 + insets.left + insets.right;
```

```
        int height = 600 + insets.top + insets.bottom;
        setSize(width, height);
        setLocation (0, 0);
        setDefaultCloseOperation (JFrame.EXIT_ON_CLOSE);

        MazePanel mp = new MazePanel();
        mp.setBounds(0, 0, 600, 600);

        TicketsPanel tp = new TicketsPanel();
        tp.setBounds( 600, 0, 300, 600);

        GameManager gm = new GameManager (mp, tp);
        mp.setGM (gm);
        tp.setGM(gm);

        add(mp);
        add(tp);

        gm.start();
    }

    //メイン
    public static void main(String[] args) {
        (new TheDiscountTickets()).setVisible(true);
    }
}
```

●GameManager

```
/***
 *** ゲーム管理者
 ***/

import java.awt.*;
import java.util.*;
import java.awt.event.*;
```

```java
public class GameManager extends Thread {
    private MazePanel mp; // 迷路パネル
    private TicketsPanel tp; // クーポン券パネル
    private Maze maze;
    public Player player;
    public ArrayList<Wall> walls;
    public int hour;
    public int minute;
    public int day;

    // 生成
    public GameManager (MazePanel mp, TicketsPanel tp) {
        this.mp = mp;
        this.tp = tp;

        walls = new ArrayList<Wall>();
        maze = new Maze_7(this);
        player = new Player(120, 480);
        player.setGM(this);
        mp.addKeyListener(player);
        hour = 6;
        minute = 0;
        day = 1;
    }

    // 実行
    public void run() {
        long t = 0;
        while(true) {
            if(t % 200 == 0 && t > 400) {
                time();
                tp.couponPanel.d_life(); // クーポンの有効期限の確認と更新
            }
            if(t % 50 == 0) player.move();
            if(t % 200 == 0){
                int n = player.out();
```

```
            if(n != 4){
                maze = maze.next(n);
                player.out_xy(n);
                player.setGM(this);
            }
        }
        if(t % 2500 == 0)tp.couponPanel.make_Coupon_Button();
        mp.repaint(); // 迷路を再描画
        tp.repaint(); // クーポンパネルを再描画
        try{
            sleep(2);
        } catch (InterruptedException e){}
        t++;
    }
}

// 迷路パネルを得る
public MazePanel getMP(){
    return mp;
}

// 現在時刻
public void time(){
    minute += 5;
    if(minute >= 60){
        minute -= 60;
        hour += 1;
        if(hour >= 24){
            hour = 0;
            day += 1;
        }
    }
}

// 描画
public void draw (Graphics g) {
```

```
        player.draw(g); // プレイヤーを描画
        maze.draw(g);   // 迷路を描画
    }
}
```

●MazePanel

```
/***
 *** 町の迷路
 ***/

import java.awt.*;
import javax.swing.*;
import java.awt.event.*;

public class MazePanel extends JPanel {
    private GameManager gm;

    // 生成
    public MazePanel(){
        setBackground(Color.gray);
        setFocusable(true);
    }

    // ゲーム管理者のセット
    public void setGM(GameManager gm){
        this.gm = gm;
    }

    public void paintComponent(Graphics g){
        super.paintComponent(g);
        gm.draw(g);
    }
}
```

●TiketsPanel

```java
/***
 *** クーポン券パネル
 ***/

import java.awt.*;
import javax.swing.*;

public class TicketsPanel extends JPanel{
    private GameManager gm;
    private MazePanel mp;
    private TimePanel timePanel; // 現在時刻表示欄
    public CouponPanel couponPanel; // クーポン表示欄

    // 生成
    public TicketsPanel(){
        setBackground(Color.black);
    }

    // ゲーム管理者のセット
    public void setGM(GameManager gm){
        this.gm = gm;
        setLayout(null);
        timePanel = new TimePanel(gm);
        timePanel.setBounds(0, 0, 300, 100);
        add(timePanel);

        couponPanel = new CouponPanel(gm);
        couponPanel.setBounds(0, 100, 300, 500);
        add(couponPanel);
    }

    // 迷路パネルのセット
    public void setMazePanel(MazePanel mp){
        this.mp = mp;
    }
```

```
    // 描画
    public void paintComponent(Graphics g){
        super.paintComponent(g);
        timePanel.paintComponents(g);
        couponPanel.paintComponents(g);
    }
}
```

●TimePanel

```
import java.awt.*;
import javax.swing.*;

public class TimePanel extends JPanel{
    private GameManager gm;
    private int hour;
    private int minute;
    private int day;

    // 生成
    public TimePanel(GameManager gm){
        this.gm = gm;
       setBackground(Color.cyan);
    }

    // 時刻のセット
    public void setTime(int h, int m){
        minute = m;
        hour = h;
    }

    // 描画
    public void draw(Graphics g){
        String s_hour = String.valueOf(gm.hour);
        String s_minute = String.valueOf(gm.minute);
        String s_day = String.valueOf(gm.day);
```

```
        String z1 = "", z2 = "";

        if(gm.hour < 10){
            z1 = "0";
        }
        if(gm.minute < 10){
            z2 = "0";
        }

        g.setColor(Color.black);
        g.setFont(new Font(Font.DIALOG, Font.BOLD, 30));
        g.drawString(s_day + "日目  " + z1 + s_hour + ":" + z2 + s_minute,
60, 70);

    }

    public void paintComponent(Graphics g){
        super.paintComponent(g);
        draw(g);
    }
}
```

●CouponPanel

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.util.Random;

public class CouponPanel extends JPanel implements ActionListener{
    private GameManager gm;
    private JButton button[] = new JButton[5];
    private Coupon coupon[] = new Coupon[5];

    Random rd = new Random();
    Color b_Color = new Color(180, 180, 180);
```

```
int a = 95;
Integer life[] = new Integer[5] ;

// 生成
public CouponPanel(GameManager gm){
    this.gm = gm;
    setBackground(Color.black);

    for(int i = 0; i < 5; i++){
        button[i] = new JButton();
        button[i].setActionCommand(String.valueOf(i));
        set_default(button[i]);
        life[i] = 0;
        add(button[i]);
    }
}

// クーポンボタンの初期化
private void set_default(JButton bt){
    bt.setForeground(Color.black);
    bt.setFont(new Font(Font.DIALOG, Font.BOLD, 18));
    bt.setText("");
    bt.setBackground(b_Color);
    bt.setPreferredSize(new Dimension(300, a));
    bt.addActionListener(this);
    bt.setEnabled(false);
}


//  クーポンとボタン生成
public void make_Coupon_Button(){
        int n = 0;
        for(int i = 0; i < 5; i++){
            if(button[i].isEnabled() == false){
                if(n == 0){
                    System.out.println("クーポン生成");
```

```
                        coupon[i] = new Coupon(gm);
                        System.out.println(coupon[i].s_name + "  " +
coupon[i].name + "¥r¥n" + String.valueOf(coupon[i].price) + "  →  " +
String.valueOf(coupon[i].d_price));
                        life[i] = coupon[i].life;
                        button[i].setText("<html>" + coupon[i].s_name +
"  " + coupon[i].name + "<br/>" + String.valueOf(coupon[i].price) + "
円  →  " + String.valueOf(coupon[i].d_price) + "円<br>" +
coupon[i].life_S);
                        button[i].setBackground(get_Shop_Color(coupon[i].s
_no));
                        button[i].setEnabled(true);
                        System.out.println(button[i].getFont());
                        n += 1;
                    }
                }
            }
    }

    // クーポンの色を取得
    public Color get_Shop_Color(int s_no){
        Color color;
        switch(s_no){
            case 1 : color = new Color(40, 120, 255); break;
            case 2 : color = new Color(255, 150, 150); break;
            case 3 : color = new Color(255, 255, 80); break;
            case 4 : color = new Color(170, 80, 80); break;
            case 5 : color = new Color(220, 180, 50); break;
            case 6 : color = new Color(30, 220, 40); break;
            case 7 : color = new Color(100, 255, 150); break;
            case 8 : color = new Color(150, 150, 50); break;
            case 9 : color = new Color(220, 50, 50); break;
            case 10 : color = new Color(180, 80, 150); break;
            case 11 : color = new Color(80, 180, 180); break;
            default : color = new Color(0, 0, 0); break;
        }
```

```
        return color;
    }

    // 有効期限の確認
    public void d_life(){
        for(int i = 0; i < 5; i++){
            if(life[i] >= 0){
                life[i] --;
                System.out.println("aaaaaaaaaaa" + life[i]);
                if(life[i] < 0){
                    set_default(button[i]);
                }
            }
        }
    }

    // 描画
    public void paintComponent(Graphics g){
        super.paintComponent(g);
    }

    // アクションリスナー
    public void actionPerformed(ActionEvent e){
        String e_button = e.getActionCommand();
        int n = Integer.valueOf(e_button).intValue();
        set_default(button[n]);
        for(int i = n + 1; i < 5; i++){
            button[i].setFocusable(false);
        }
    }
}
```

●Coupon

```
import java.awt.*;
import javax.swing.*;
```

```java
import java.awt.event.*;
import java.util.Random;

public class Coupon {
    public String s_name = new String(); // 店舗名
    public String name = new String(); // 商品名
    public int price; // 定価
    public int d_price; // 割引価格
    public int life; // 有効期限
    public int s_no; // 使用可能店舗
    public String life_S; // 有効期限を示す文字列

    GameManager gm;
    Random rd = new Random();
    int r;

    public Coupon(GameManager gm){
        this.gm = gm;
        r = rd.nextInt(11);
        switch(r){
            case 0 : make_RAWSON_C(); s_name = "RAWSON"; break;
            case 1 : make_INUQLO_C(); s_name = "INUQLO"; break;
            case 2 : make_Mk_C(); s_name = "モクドナルド"; break;
            case 3 : make_GANUTO_C(); s_name = "ガヌト"; break;
            case 4 : make_YOSHIDAYA_C(); s_name = "吉田家"; break;
            case 5 : make_MOON_C(); s_name = "ムーンバックス"; break;
            case 6 : make_MITORI_C(); s_name = "ミトリ"; break;
            case 7 : make_MATSUMOTO_C(); s_name = "マツモトキヨコ"; break;
            case 8 : make_TOHU_C(); s_name = "TOHU シネマズ"; break;
            case 9 : make_MINVALUE_C(); s_name = "ミニマムバリュー"; break;
            case 10 : make_YAMATA_C(); s_name = "YAMATA 電機"; break;
            default : make_RAWSON_C(); s_name = "RAWSON";
        }
    }
```

```
    private void make_COUPON(int s_no, String name, int price, int
d_price, int life){
        this.s_no = s_no;
        this.name = name;
        this.price = price;
        this.d_price = d_price;
        this.life = life;
        setLife();
    }

    private void make_RAWSON_C(){
        r = rd.nextInt(3);
        switch(r){
            case 0 : make_COUPON(1, "R チキ", 180, 160, 100); break;
            case 1 : make_COUPON(1, "とんかつ弁当", 560, 520, 300); break;
            case 2 : make_COUPON(1, "アイスコーヒーS", 100, 0, 150); break;
        }
    }

    private void make_INUQLO_C(){
        r = rd.nextInt(3);
        switch(r){
            case 0 : make_COUPON(2, "T シャツ", 1380, 980, 300); break;
            case 1 : make_COUPON(2, "ジーンズ", 3990, 2990, 225); break;
            case 2 : make_COUPON(2, "コート", 5990, 4590, 175); break;
        }
    }

    private void make_Mk_C(){
        r = rd.nextInt(3);
        switch(r){
            case 0 : make_COUPON(3, "コーラ S", 100, 0, 300); break;
            case 1 : make_COUPON(3, "ポテト L", 330, 150, 225); break;
            case 2 : make_COUPON(3, "ハンバーガーセット", 790, 4290, 175);
break;
        }
```

```
    }

    private void make_GANUTO_C(){
        r = rd.nextInt(3);
        switch(r){
            case 0 : make_COUPON(4, "ミニサラダ", 290, 260, 300); break;
            case 1 : make_COUPON(4, "パフェ", 690, 620, 225); break;
            case 2 : make_COUPON(4, "ハンバーグセット", 990, 860, 275);
break;
        }
    }

    private void make_YOSHIDAYA_C(){
        r = rd.nextInt(3);
        switch(r){
            case 0 : make_COUPON(5, "牛丼", 430, 980, 300); break;
            case 1 : make_COUPON(5, "温玉", 80, 0, 225); break;
            case 2 : make_COUPON(5, "特盛牛丼", 780, 630, 175); break;
        }
    }

    private void make_MOON_C(){
        r = rd.nextInt(3);
        switch(r){
            case 0 : make_COUPON(6, "ドリップコーヒー Short", 310, 0, 120);
break;
            case 1 : make_COUPON(6, "カフェラテ Tall", 420, 380, 225);
break;
            case 2 : make_COUPON(6, "フラッペ Grande", 580, 460, 175);
break;
        }
    }

    private void make_MITORI_C(){
        r = rd.nextInt(3);
        switch(r){
```

```
            case 0 : make_COUPON(7, "クッション", 680, 560, 300); break;
            case 1 : make_COUPON(7, "デスク", 5980, 4280, 225); break;
            case 2 : make_COUPON(7, "ソファ", 8980, 5980, 175); break;
        }
    }

    private void make_MATSUMOTO_C(){
        r = rd.nextInt(3);
        switch(r){
            case 0 : make_COUPON(8, "エナジードリンク", 210, 0, 150); break;
            case 1 : make_COUPON(8, "シャンプー", 480, 340, 225); break;
            case 2 : make_COUPON(8, "日焼け止めクリーム", 850, 690, 175);
break;
        }
    }

    private void make_TOHU_C(){
        r = rd.nextInt(3);
        switch(r){
            case 0 : make_COUPON(9, "ソフトドリンク", 380, 190, 300); break;
            case 1 : make_COUPON(9, "ポップコーン", 490, 290, 225); break;
            case 2 : make_COUPON(9, "無料鑑賞券", 1900, 0, 100); break;
        }
    }

    private void make_MINVALUE_C(){
        r = rd.nextInt(3);
        switch(r){
            case 0 : make_COUPON(10, "から揚げ", 380, 190, 300); break;
            case 1 : make_COUPON(10, "みかん", 290, 100, 225); break;
            case 2 : make_COUPON(10, "国産牛肉", 2990, 1590, 175); break;
        }
    }

    private void make_YAMATA_C(){
        r = rd.nextInt(3);
```

```
        switch(r){
            case 0 : make_COUPON(11, "乾電池", 780, 490, 180); break;
            case 2 : make_COUPON(11, "ミキサー", 12990, 7990, 225); break;
            case 2 : make_COUPON(11, "電子レンジ", 59700, 34800, 275);
break;
        }
    }

    private void setLife(){
        int m = gm.minute + life * 5 ;
        int life_m = m % 60;
        int h = gm.hour + (m - life_m) / 60;
        int life_h = h % 24;
        int life_d = gm.day + (h - life_h) / 24;

        String life_Sh = String.valueOf(life_h);
        String life_Sm = String.valueOf(life_m);
        String life_Sd = String.valueOf(life_d);
        String z1 = "", z2 = "";

        if(life_h < 10){
            z1 = "0";
        }
        if(life_m < 10){
            z2 = "0";
        }

        life_S = (life_Sd + "日目  " + z1 + life_Sh + ":" + z2 + life_Sm +
"まで有効");
    }
}
```

●GameChar

```
/***
 *** ゲームキャラクター
 ***/
```

```java
import java.awt.*;

public abstract class GameChar {
  protected int px,py; // 位置
  protected int cx, cy; // 判定領域の中心
  protected int cw,ch; // 判定領域の大きさ

  // 生成
  public GameChar(){
    this(0,0);
  }
  public GameChar(int x, int y){
    px=x; py=y;
    cx=x; cy=y;

    cw=ch=0; // 最初の大きさは 0x0
  }

  // 判定領域の中心の設定
  public void setCenter(int dx, int dy){
      cx = dx;
      cy = dy;
  }

  // 大きさ設定
  public void setSize(int w, int h){
    cw=w;
    ch=h;
  }

  // 位置を得る
  public double getX(){ return px; }
  public double getY(){ return py; }

  // 重なり判定
  public boolean overlap(GameChar c){
```

```
    double x11=cx-cw/2,y11=cy-ch/2;
    double x12=cx+cw/2,y12=cy+ch/2;
    double x21=c.cx-c.cw/2,y21=c.cy-c.ch/2;
    double x22=c.cx+c.cw/2,y22=c.cy+c.ch/2;
    return
      (((x11<=x21 && x21<x12)||(x21<=x11 && x11<x22)) &&
       ((y11<=y21 && y21<y12)||(y21<=y11 && y11<y22)));
  }
}
```

●Player

```
/***
 *** プレーヤークラス
 ***/

import java.awt.*;
import java.util.*;
import java.awt.event.*;
//import java.math.*;

public class Player extends GameChar implements KeyListener{

    // 座標(x, y)は GameChar が所持
    private int dx, dy; // 移動量
    private MazePanel mp;
    private GameManager gm;
    private ArrayList<Wall> walls;
    private int dir; // 方向
    private int a = 20;

    // 生成
    public Player (int x, int y){
        px = x;
        py = y;
        dx = 0;
        dy = 0;
```

```
        dir = 0;
        setSize(5, 5); // 衝突判定の大きさ
        setCenter(px, py - 15); // 衝突判定の中心（三角形の先端）
        walls = new ArrayList<Wall>();
    }

    public void setGM(GameManager gm){
        this.gm = gm;
        this.walls = gm.walls;
    }

    // 移動
    public void move() {
        System.out.println(dx +" "+ dy);
        px += dx;
        py += dy;
        dx = 0;
        dy = 0;
    }

    // パネル外へ出た方向の判定
    public int out(){
        if(py < 0) return 0;
        if(py > 600) return 1;
        if(px > 600) return 2;
        if(px < 0) return 3;
        else return 4;
    }

    // プレイヤーがウィンドウから出た後の座標
    public void out_xy(int n){
        switch(n){
            case 0 : py += 600; break;
            case 1 : py -= 600; break;
            case 2 : px -= 600; break;
            case 3 : px += 600; break;
```

```
            default : break;
        }
    }

    // 描画
    public void draw (Graphics g) {
        int[] x = new int[3];
        int[] y = new int[3];
        if(dir == 0){ // 上
            x[0] = (int)px;
            x[1] = (int)px - 10;
            x[2] = (int)px + 10;
            y[0] = (int)py - 15;
            y[1] = (int)py + 10;
            y[2] = (int)py + 10;
        }else if(dir == 1){ // 下
            x[0] = (int)px;
            x[1] = (int)px - 10;
            x[2] = (int)px + 10;
            y[0] = (int)py + 15;
            y[1] = (int)py - 10;
            y[2] = (int)py - 10;
        }else if(dir == 2){ // 右
            x[0] = (int)px + 15;
            x[1] = (int)px - 10;
            x[2] = (int)px - 10;
            y[0] = (int)py;
            y[1] = (int)py + 10;
            y[2] = (int)py - 10;
        }else{ // 左
            x[0] = (int)px - 15;
            x[1] = (int)px + 10;
            x[2] = (int)px + 10;
            y[0] = (int)py;
            y[1] = (int)py + 10;
            y[2] = (int)py - 10;
```

```
        }
        setCenter(x[0], y[0]); // 衝突判定中心の更新
        g.setColor (new Color (255,255, 100)); // 明るい黄色で
        g.fillPolygon (x, y, 3);                // 塗りつぶして
        g.setColor (new Color (255,200,20) );  // 暗い黄色で
        g.drawPolygon (x, y, 3);                // 枠を描く
    }

    // キーリスナー
    public void keyTyped(KeyEvent ev) {}

    public void keyPressed(KeyEvent ev) {
        if(ev.getKeyCode()==KeyEvent.VK_UP){
            if(dir == 0){
                dy = -1 * a;
                for(Wall w:walls){
                    if(overlap(w) == true){
                        dy = 0;
                        break;
                    }
                }
            }
            dir = 0;
        }
        if(ev.getKeyCode()==KeyEvent.VK_DOWN){
            if(dir == 1){
                dy = a;
                for(Wall w:walls){
                    if(overlap(w) == true){
                    dy = 0;
                    break;
                    }
                }
            }
            dir = 1;
        }
```

```
        if(ev.getKeyCode()==KeyEvent.VK_RIGHT){
            if(dir == 2){
                dx = a;
                for(Wall w:walls){
                    if(overlap(w) == true){
                        dx = 0;
                        break;
                    }
                }
            }
            dir = 2;
        }
        if(ev.getKeyCode()==KeyEvent.VK_LEFT){
            if(dir == 3){
                dx = -1 * a;
                for(Wall w:walls){
                    if(overlap(w) == true){
                        dx = 0;
                        break;
                    }
                }
            }
            dir = 3;
        }
    }

    public void keyReleased(KeyEvent ev) {}
}
```

●Shop

```
import java.awt.*;
import javax.swing.*;
import java.util.*;
import java.awt.event.*;

public class Shop extends GameChar{
```

```
    public Color color;
    private int n;
    private int no;
    private String s_shop;

    public Shop(int s_no, int x, int y){
        px=x; py=y;
        cx=x; cy=y;
        this.no = s_no;

        switch(no){
            case 1 : make_RAWSON(); break;
            case 2 : make_INUQLO(); break;
            case 3 : make_Mk(); break;
            case 4 : make_GANUTO(); break;
            case 5 : make_YOSHIDAYA(); break;
            case 6 : make_MOON(); break;
            case 7 : make_MITORI(); break;
            case 8 : make_MATSUMOTO(); break;
            case 9 : make_TOHU(); break;
            case 10 : make_MINVALUE(); break;
            case 11 : make_YAMATA(); break;

        }
    }

    private void make_RAWSON(){ // 1
        n = 1;
        setSize(100, 100);
        color = new Color(40, 120, 255);
        s_shop = new String("RAWSON");
    }

    private void make_INUQLO(){ // 2
        n = 2;
        setSize(200, 200);
```

```
        color = new Color(255, 150, 150);
        s_shop = new String("INUQLO");
    }

    private void make_Mk(){ // 3
        n = 1;
        setSize(100, 100);
        color = new Color(255, 255, 80);
        s_shop = new String("モクドナルド");
    }

    private void make_GANUTO(){ // 4
        n = 1;
        setSize(100, 100);
        color = new Color(170, 80, 80);
        s_shop = new String("ガヌト");
    }

    private void make_YOSHIDAYA(){ // 5
        n = 1;
        setSize(100, 100);
        color = new Color(220, 180, 50);
        s_shop = new String("吉田家");
    }

    private void make_MOON(){ // 6
        n = 1;
        setSize(100, 100);
        color = new Color(30, 220, 40);
        s_shop = new String("ムーンバックス");
    }

    private void make_MITORI(){ // 7
        n = 2;
        setSize(200, 200);
        color = new Color(100, 255, 150);
```

```
        s_shop = new String("ミトリ");
    }

    private void make_MATSUMOTO(){ // 8
        n = 2;
        setSize(200, 200);
        color = new Color(150, 150, 50);
        s_shop = new String("マツモトキヨコ");
    }

    private void make_TOHU(){ // 9
        n = 2;
        setSize(200, 200);
        color = new Color(220, 50, 50);
        s_shop = new String("TOHU シネマズ");
    }

    private void make_MINVALUE(){ // 10
        n = 2;
        setSize(200, 200);
        color = new Color(180, 80, 150);
        s_shop = new String("ミニマムバリュー");
    }

    private void make_YAMATA(){ // 11
        n = 2;
        setSize(200, 200);
        color = new Color(80, 180, 180);
        s_shop = new String("YAMATA 電機");
    }

    // 描画
    public void draw(Graphics g){
        int[] x1 = new int[3];
        int[] y1 = new int[3];
        x1[0] = (int)px;
```

```
        y1[0] = (int)py - n * 30;
        x1[1] = (int)px + n * 40;
        y1[1] = (int)py - n * 5;
        x1[2] = (int)px - n * 40;
        y1[2] = (int)py - n * 5;

        int x2 = px - n * 30;
        int y2 = py - n * 5;

        FontMetrics fm = g.getFontMetrics();
        Rectangle rectText = fm.getStringBounds(s_shop, g).getBounds();

        int x3 = px - rectText.width/2;
        int y3 = py + n * 20 - rectText.height/2;

        g.setColor (this.color);
        g.fillPolygon (x1 , y1, 3);
        g.fillRect(x2, y2, n * 60, n * 45);

        g.setColor(Color.black);
        g.drawString(s_shop, x3, y3);

    }
}
```

●Wall

```
/***
 *** 迷路の壁
 ***/
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;

public class Wall extends GameChar{
    private int dx, dy;
```

```
    // 生成
    public Wall(int x, int y, int dx, int dy){
        this.px = x;
        this.py = y;
        this.dx = dx;
        this.dy = dy;
        setCenter(x, y); // 衝突判定の中心
        setSize(dx + 10, dy + 10); // 衝突判定の範囲
    }

    // 描画
    public void draw(Graphics g){
        g.setColor(Color.BLACK);
        g.fillRect((int)(px - dx/2), (int)(py - dy/2), (int)dx, (int)dy);
    }

}
```

●Maze

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.util.ArrayList;

public abstract class Maze{
    private GameManager gm;

    // 生成
    public Maze(GameManager gm){
        this.gm = gm;
    }

     // 描画
    public abstract void draw(Graphics g);
```

```
    // 次のゲームパネル
    public abstract Maze next(int n);

}
```

●Maze_1

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.util.ArrayList;

public class Maze_1 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[15];
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;
    private Shop shop[] = new Shop[4];
    private ArrayList<Shop> shops = new ArrayList<Shop>();

    // 生成
    public Maze_1(GameManager gm){
        super(gm);
        this.gm = gm;
        wall[0] = new Wall(300, 0, 600, a);
        wall[1] = new Wall(0, 300, a, 600);
        wall[2] = new Wall(120, 600, 240, a);
        wall[3] = new Wall(420, 600, 120, a);

        wall[4] = new Wall(600, 120, a, 240);
        wall[5] = new Wall(600, 480, a, 240);

        wall[6] = new Wall(180, 240, 120, a);
        wall[7] = new Wall(420, 240, 120, a);
        wall[8] = new Wall(60, 360, 120, a);
        wall[9] = new Wall(360, 480, 240, a);
```

```
        wall[10] = new Wall(360, 120, a, 240);
        wall[11] = new Wall(240, 240, a, 240);
        wall[12] = new Wall(120, 420, a, 120);
        wall[13] = new Wall(360, 420, a, 120);
        wall[14] = new Wall(480, 420, a, 360);

        for(int i= 0; i < 15; i++){
            walls.add(wall[i]);
        }

        gm.walls = this.walls;

        shop[0] = new Shop(8, 480, 120);  // マツモトキヨコ
        shop[1] = new Shop(11, 120, 120); // YAMATA 電機
        shop[2] = new Shop(1, 60, 420); // RAWSON
        shop[3] = new Shop(3, 420, 420); // モクドナルド

        for(int i= 0; i < 4; i++){
            shops.add(shop[i]);
        }
    }

     // 描画
     public void draw(Graphics g){
        for(Wall w:walls) w.draw(g);
        for(Shop s:shops) s. draw(g);
    }

     // 次のゲームパネル
    public Maze next(int n){
        switch(n){
            case 0 : // なし
            case 1 : return new Maze_4(gm);
            case 2 : return new Maze_2(gm);
            case 3 : // なし
```

```
            default : return this;
        }
    }
}
```

●Maze_2

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.util.ArrayList;

public class Maze_2 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[17];
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;
    private Shop shop[] = new Shop[3];
    private ArrayList<Shop> shops = new ArrayList<Shop>();

    // 生成
    public Maze_2(GameManager gm){
        super(gm);
        this.gm = gm;
        wall[0] = new Wall(300, 0, 600, a);
        wall[1] = new Wall(0, 120, a, 240);
        wall[2] = new Wall(0, 480, a, 240);
        wall[3] = new Wall(300, 600, 600, a);
        wall[4] = new Wall(600, 180, a, 360);
        wall[5] = new Wall(600, 540, a, 120);

        wall[6] = new Wall(540, 120, 120, a);
        wall[7] = new Wall(300, 240, 120, a);
        wall[8] = new Wall(420, 360, 120, a);
        wall[9] = new Wall(180, 360, 120, a);
        wall[10] = new Wall(300, 480, 120, a);
```

```
        wall[11] = new Wall(240, 180, a, 120);
        wall[12] = new Wall(480, 180, a, 120);
        wall[13] = new Wall(120, 300, a, 120);
        wall[14] = new Wall(360, 300, a, 120);

        wall[15] = new Wall(480, 480, a, 240);
        wall[16] = new Wall(240, 540, a, 120);

        for(int i= 0; i < 17; i++){
            walls.add(wall[i]);
        }

        gm.walls = this.walls;

        shop[0] = new Shop(7, 120, 480); // ミトリ
        shop[1] = new Shop(5, 540, 60); // 吉田家
        shop[2] = new Shop(4, 300, 540); // ガヌト

        for(int i= 0; i < 3; i++){
            shops.add(shop[i]);
        }

    }

     // 描画
     public void draw(Graphics g){
        for(Wall w:walls) w.draw(g);
        for(Shop s:shops) s. draw(g);
    }

     // 次のゲームパネル
    public Maze next(int n){
        switch(n){
            case 0 : // 無し
            case 1 : return new Maze_5(gm);
            case 2 : return new Maze_3(gm);
```

```
            case 3 : return new Maze_1(gm);
            default : return this;
        }
    }
}
```

●Maze_3

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.util.ArrayList;

public class Maze_3 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[13];
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;
    private Shop shop[] = new Shop[3];
    private ArrayList<Shop> shops = new ArrayList<Shop>();

    // 生成
    public Maze_3(GameManager gm){
        super(gm);
        this.gm = gm;
        wall[0] = new Wall(300, 0, 600, a);
        wall[1] = new Wall(0, 180, a, 360);
        wall[2] = new Wall(0, 540, a, 120);
        wall[3] = new Wall(60, 600, 120, a);
        wall[4] = new Wall(420, 600, 360, a);
        wall[5] = new Wall(600, 300, a, 600);

        wall[6] = new Wall(300, 240, 360, a);
        wall[7] = new Wall(180, 480, 120, a);

        wall[8] = new Wall(360, 120, a, 240);
```

```
        wall[9] = new Wall(120, 240, a, 240);
        wall[10] = new Wall(480, 360, a, 240);
        wall[11] = new Wall(360, 480, a, 240);
        wall[12] = new Wall(120, 540, a, 120);

        for(int i= 0; i < 13; i++){
            walls.add(wall[i]);
        }

        gm.walls = this.walls;

        shop[0] = new Shop(1, 60, 540); // RAWSON
        shop[1] = new Shop(9, 480, 120); // TOHU シネマズ
        shop[2] = new Shop(10, 240, 120); // ミニマムバリュー

        for(int i= 0; i < 3; i++){
            shops.add(shop[i]);
        }

    }

     // 描画
     public void draw(Graphics g){
        for(Wall w:walls) w.draw(g);
        for(Shop s:shops) s. draw(g);
    }

     // 次のゲームパネル
    public Maze next(int n){
        switch(n){
            case 0 : // なし
            case 1 : return new Maze_6(gm);
            case 2 : // なし
            case 3 : return new Maze_2(gm);
            default : return this;
        }
```

```
    }
}
```

●Maze_4

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.util.ArrayList;

public class Maze_4 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[19];
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;
    private Shop shop[] = new Shop[3];
    private ArrayList<Shop> shops = new ArrayList<Shop>();

    // 生成
    public Maze_4(GameManager gm){
        super(gm);
        this.gm = gm;
        wall[0] = new Wall(120, 0, 240, a);
        wall[1] = new Wall(420, 0, 120, a);
        wall[2] = new Wall(0, 300, a, 600);
        wall[3] = new Wall(180, 600, 120, a);
        wall[4] = new Wall(420, 600, 120, a);
        wall[5] = new Wall(600, 120, a, 240);
        wall[6] = new Wall(600, 480, a, 240);

        wall[7] = new Wall(360, 120, 480, a);
        wall[8] = new Wall(540, 240, 120, a);
        wall[9] = new Wall(60, 360, 120, a);
        wall[10] = new Wall(300, 360, 120, a);
        wall[11] = new Wall(540, 360, 120, a);
        wall[12] = new Wall(420, 480, 120, a);
```

```
        wall[13] = new Wall(240, 60, a, 120);
        wall[14] = new Wall(360, 180, a, 120);
        wall[15] = new Wall(120, 180, a, 120);
        wall[16] = new Wall(480, 420, a, 120);
        wall[17] = new Wall(240, 480, a, 240);
        wall[18] = new Wall(360, 540, a, 120);


        for(int i= 0; i < 19; i++){
            walls.add(wall[i]);
        }

        gm.walls = this.walls;

        shop[0] = new Shop(3, 180, 60); // モクドナルド
        shop[1] = new Shop(1, 540, 180); // RAWSON
        shop[2] = new Shop(6, 540, 420); // ムーンバックス

        for(int i= 0; i < 3; i++){
            shops.add(shop[i]);
        }
    }

     // 描画
     public void draw(Graphics g){
        for(Wall w:walls) w.draw(g);
        for(Shop s:shops) s. draw(g);
    }

     // 次のゲームパネル
    public Maze next(int n){
        switch(n){
            case 0 : return new Maze_1(gm);
            case 1 : return new Maze_7(gm);
            case 2 : return new Maze_5(gm);
            case 3 : // なし
```

```
            default : return this;
        }
    }
}
```

●Maze_5

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.util.ArrayList;

public class Maze_5 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[18];
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;
    private Shop shop[] = new Shop[3];
    private ArrayList<Shop> shops = new ArrayList<Shop>();

    // 生成
    public Maze_5(GameManager gm){
        super(gm);
        this.gm = gm;
        wall[0] = new Wall(300, 0, 600, a);
        wall[1] = new Wall(0, 120, a, 240);
        wall[2] = new Wall(0, 480, a, 240);
        wall[3] = new Wall(180, 600, 360, a);
        wall[4] = new Wall(540, 600, 120, a);
        wall[5] = new Wall(600, 180, a, 360);
        wall[6] = new Wall(600, 540, a, 120);

        wall[7] = new Wall(180, 120, 120, a);
        wall[8] = new Wall(360, 240, 240, a);
        wall[9] = new Wall(60, 240, 120, a);
        wall[10] = new Wall(540, 360, 120, a);
        wall[11] = new Wall(240, 480, 240, a);
```

```
        wall[12] = new Wall(540, 480, 120, a);

        wall[13] = new Wall(240, 60, a, 120);
        wall[14] = new Wall(360, 120, a, 240);
        wall[15] = new Wall(120, 300, a, 120);
        wall[16] = new Wall(240, 300, a, 120);
        wall[17] = new Wall(360, 540, a, 120);

        for(int i= 0; i < 18; i++){
            walls.add(wall[i]);
        }

        gm.walls = this.walls;

        shop[0] = new Shop(10, 480, 120); // ミニマムバリュー
        shop[1] = new Shop(6, 180, 60); // ムーンバックス
        shop[2] = new Shop(5, 300, 540); // 吉田家

        for(int i= 0; i < 3; i++){
            shops.add(shop[i]);

        }
    }

     // 描画
     public void draw(Graphics g){
        for(Wall w:walls) w.draw(g);
        for(Shop s:shops) s. draw(g);
    }

     // 次のゲームパネル
    public Maze next(int n){
        switch(n){
            case 0 : return new Maze_2(gm);
            case 1 : return new Maze_8(gm);
            case 2 : return new Maze_6(gm);
```

```
            case 3 : return new Maze_4(gm);
            default : return this;
        }
    }
}
```

●Maze_6

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.util.ArrayList;

public class Maze_6 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[19];
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;
    private Shop shop[] = new Shop[3];
    private ArrayList<Shop> shops = new ArrayList<Shop>();

    // 生成
    public Maze_6(GameManager gm){
        super(gm);
        this.gm = gm;
        wall[0] = new Wall(0, 180, a, 360);
        wall[1] = new Wall(0, 540, a, 120);
        wall[2] = new Wall(60, 600, 120, a);
        wall[3] = new Wall(420, 600, 360, a);
        wall[4] = new Wall(600, 300, a, 600);
        wall[5] = new Wall(420, 0, 360, a);
        wall[6] = new Wall(60, 0, 120, a);

        wall[7] = new Wall(540, 240, 120, a);
        wall[8] = new Wall(300, 360, 120, a);
        wall[9] = new Wall(180, 480, 120, a);
        wall[10] = new Wall(420, 480, 120, a);
```

```
        wall[11] = new Wall(120, 180, a, 360);
        wall[12] = new Wall(120, 540, a, 120);
        wall[13] = new Wall(240, 60, a, 120);
        wall[14] = new Wall(360, 120, a, 240);
        wall[15] = new Wall(480, 300, a, 120);
        wall[16] = new Wall(240, 360, a, 240);
        wall[17] = new Wall(120, 540, a, 120);
        wall[18] = new Wall(480, 540, a, 120);

        for(int i= 0; i < 19; i++){
            walls.add(wall[i]);
        }

        gm.walls = this.walls;

        shop[0] = new Shop(2, 480, 120); // INUQLO
        shop[1] = new Shop(3, 60, 60); // モクドナルド
        shop[2] = new Shop(4, 540, 300); // ガヌト

        for(int i= 0; i < 3; i++){
            shops.add(shop[i]);
        }
    }

     // 描画
     public void draw(Graphics g){
        for(Wall w:walls) w.draw(g);
        for(Shop s:shops) s. draw(g);
    }

     // 次のゲームパネル
    public Maze next(int n){
        switch(n){
            case 0 : return new Maze_3(gm);
            case 1 : return new Maze_9(gm);
```

```
            case 2 : // なし
            case 3 : return new Maze_5(gm);
            default : return this;
        }
    }
}
```

●Maze_7

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.util.ArrayList;

public class Maze_7 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[18];
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;
    private Shop shop[] = new Shop[3];
    private ArrayList<Shop> shops = new ArrayList<Shop>();

    // 生成
    public Maze_7(GameManager gm){
        super(gm);
        this.gm = gm;
        wall[0] = new Wall(0, 300, a, 600);
        wall[1] = new Wall(300, 600, 600, a);
        wall[2] = new Wall(600, 540, a, 120);
        wall[3] = new Wall(600, 240, a, 240);
        wall[4] = new Wall(420, 0, 120, a);
        wall[5] = new Wall(180, 0, 120, a);

        wall[6] = new Wall(300, 120, 120, a);
        wall[7] = new Wall(540, 120, 120, a);
        wall[8] = new Wall(180, 240, 120, a);
        wall[9] = new Wall(60, 360, 120, a);
```

```
        wall[10] = new Wall(420, 360, 120, a);

        wall[11] = new Wall(120, 120, a, 240);
        wall[12] = new Wall(360, 60, a, 120);
        wall[13] = new Wall(480, 60, a, 120);
        wall[14] = new Wall(240, 180, a, 120);
        wall[15] = new Wall(360, 360, a, 240);
        wall[16] = new Wall(240, 480, a, 240);
        wall[17] = new Wall(480, 480, a, 240);

        for(int i= 0; i < 18; i++){
            walls.add(wall[i]);
        }

        gm.walls = this.walls;

        shop[0] = new Shop(1, 420, 60); // RAWSON
        shop[1] = new Shop(6, 180, 180); // ムーンバックス
        shop[2] = new Shop(4, 420, 420); // ガヌト

        for(int i= 0; i < 3; i++){
            shops.add(shop[i]);
        }
    }

     // 描画
     public void draw(Graphics g){
        for(Wall w:walls) w.draw(g);
        for(Shop s:shops) s. draw(g);
    }

     // 次のゲームパネル
    public Maze next(int n){
        switch(n){
            case 0 : return new Maze_4(gm);
            case 1 : // なし
```

```
            case 2 : return new Maze_8(gm);
            case 3 : // なし
            default : return this;
        }
    }
}
```

●Maze_8

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.util.ArrayList;

public class Maze_8 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[19];
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;
    private Shop shop[] = new Shop[2];
    private ArrayList<Shop> shops = new ArrayList<Shop>();

    // 生成
    public Maze_8(GameManager gm){
        super(gm);
        this.gm = gm;
        wall[0] = new Wall(0, 240, a, 240);
        wall[1] = new Wall(0, 540, a, 120);
        wall[2] = new Wall(300, 600, 600, a);
        wall[3] = new Wall(600, 240, a, 480);
        wall[4] = new Wall(180, 0, 360, a);
        wall[5] = new Wall(540, 0, 120, a);

        wall[6] = new Wall(180, 120, 120, a);
        wall[7] = new Wall(420, 120, 120, a);
        wall[8] = new Wall(60, 240, 120, a);
        wall[9] = new Wall(480, 240, 240, a);
```

```
        wall[10] = new Wall(240, 360, 240, a);
        wall[11] = new Wall(180, 480, 120, a);

        wall[12] = new Wall(120, 60, a, 120);
        wall[13] = new Wall(480, 60, a, 120);
        wall[14] = new Wall(240, 180, a, 120);
        wall[15] = new Wall(120, 300, a, 120);
        wall[16] = new Wall(360, 420, a, 120);
        wall[17] = new Wall(480, 480, a, 240);
        wall[18] = new Wall(120, 540, a, 120);

        for(int i= 0; i < 19; i++){
            walls.add(wall[i]);
        }

        gm.walls = this.walls;

        shop[0] = new Shop(3, 540, 60); // モクドナルド
        shop[1] = new Shop(5, 180, 540); // 吉田家

        for(int i= 0; i < 2; i++){
            shops.add(shop[i]);
        }
    }

     // 描画
     public void draw(Graphics g){
        for(Wall w:walls) w.draw(g);
        for(Shop s:shops) s. draw(g);
    }

     // 次のゲームパネル
    public Maze next(int n){
        switch(n){
            case 0 : return new Maze_5(gm);
            case 1 : // なし
```

```
            case 2 : return new Maze_9(gm);
            case 3 : return new Maze_7(gm);
            default : return this;
        }
    }
}
```

●Maze_9

```
import java.awt.*;
import javax.swing.*;
import java.awt.event.*;
import java.util.ArrayList;

public class Maze_9 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[16];
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;
    private Shop shop[] = new Shop[2];
    private ArrayList<Shop> shops = new ArrayList<Shop>();

    // 生成
    public Maze_9(GameManager gm){
        super(gm);
        this.gm = gm;
        wall[0] = new Wall(0, 240, a, 480);
        wall[1] = new Wall(300, 600, 600, a);
        wall[2] = new Wall(600, 300, a, 600);
        wall[3] = new Wall(420, 0, 360, a);
        wall[4] = new Wall(60, 0, 120, a);

        wall[5] = new Wall(180, 240, 120, a);
        wall[6] = new Wall(420, 240, 120, a);
        wall[7] = new Wall(240, 360, 240, a);
        wall[8] = new Wall(60, 480, 120, a);
```

```
        wall[9] = new Wall(420, 480, 120, a);

        wall[10] = new Wall(240, 60, a, 120);
        wall[11] = new Wall(360, 120, a, 240);
        wall[12] = new Wall(120, 180, a, 120);
        wall[13] = new Wall(480, 300, a, 120);
        wall[14] = new Wall(240, 420, a, 360);
        wall[15] = new Wall(360, 540, a, 120);


        for(int i= 0; i < 16; i++){
            walls.add(wall[i]);
        }

        gm.walls = this.walls;

        shop[0] = new Shop(8, 480, 120); // マツモトキヨコ
        shop[1] = new Shop(1, 180, 300); // RAWSON

        for(int i= 0; i < 2; i++){
            shops.add(shop[i]);
        }
    }

     // 描画
     public void draw(Graphics g){
        for(Wall w:walls) w.draw(g);
        for(Shop s:shops) s. draw(g);
    }

     // 次のゲームパネル
    public Maze next(int n){
        switch(n){
            case 0 : return new Maze_6(gm);
            case 1 : // なし
            case 2 : // なし
```

```
            case 3 : return new Maze_8(gm);
            default : return this;
        }
    }
}
```

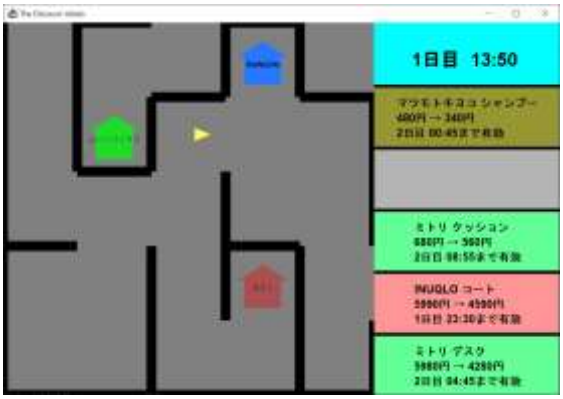
●実行結果

起動後①のようなウィンドウが表示された．しばらくたつとクーポン券が 1 つずつ増えていき，最終的には②のように５つのボタンすべてがクーポン券になった．その後，上から２番目のクーポン券をクリックすると③のようにクーポン券は消え，しばらくすると④のように新たにクーポン券ボタンが生成された．また⑤から⑥のように，有効期限が切れたクーポン券は自然に消えた．

①

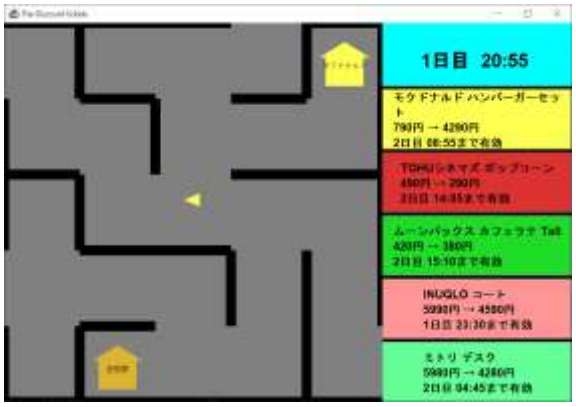
②

③

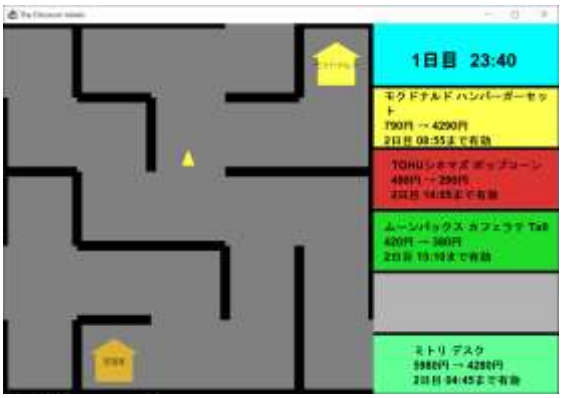


④

⑤



⑥



また，各 Maze は以下のようになった．

・Maze_ 1

・Maze_2

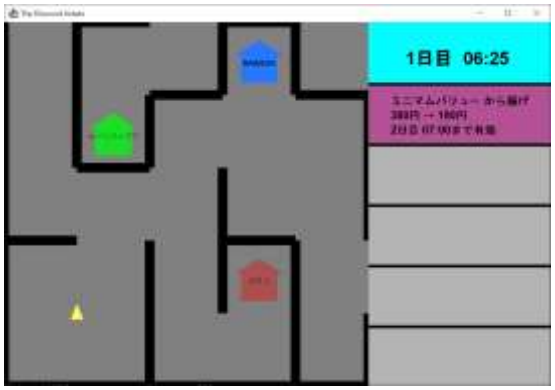
・Maze_ 3

・Maze_4

・Maze_5

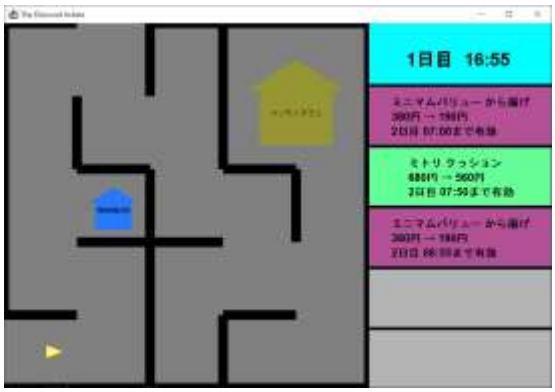
・Maze_6

・Maze_7



・Maze_8

・Maze_9



＜最終（SP4）要件定義＞

●現在時刻の下に残金と得した金額を表示

●プレーヤーが店と接触した状態でその店のクーポンをクリックするとクーポン券を使える

・クーポン券に応じて残金が減り，得した金額が増える

・クーポン券に応じて時間が経過する

●表示画面の切り替え

・スタート画面で S キーを押すとゲームスタート

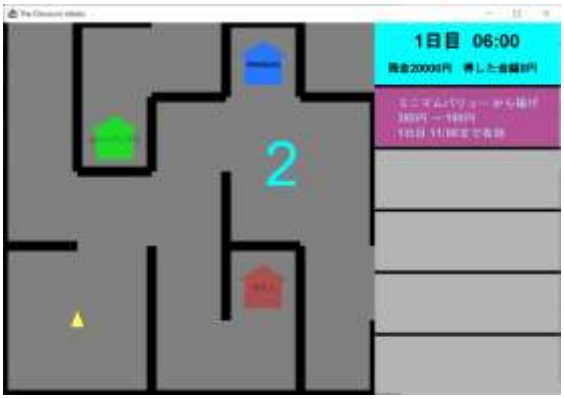
・ゲーム画面で制限時間が経過，もしくは残金がゼロになるとスタート画面に戻る

・ゲームプレイ後のスタート画面では，ゲーム名の代わりにスコア（得した金額）が表示される

＜最終（SP4）外部設計＞

●スタート画面

●ゲーム画面



●スタート画面（ゲームプレイ後）

＜最終（SP4）内部設計＞

●27 個のクラス

●GameManager

・ステージが StartStage ならば，mazePanel と ticketsPanel を見えなくする．

・ステージが StartStage か GameStage かによってループ処理を変える．

●BackPanel

・背景のパネル

・mazePanel と ticketsPanel が貼り付けられている

●Msg

・メッセージの基底クラス

●MsgBrink

・点滅するメッセージのクラス

・Msg の派生クラス

●Stage

・ステージの基底クラス

●StageStart

・スタート画面

・Stage の派生クラス

・ゲーム名の「The Discount Tickets」と指示文「Press S key to Start !!」が表示される

・ゲームプレイ後であれば，スコアと指示文「Press S key to Restart !!」が表示される

・S キーを押すとゲーム画面に切り替わる

●StageGame

・ゲーム画面

・Stage の派生クラス

・スタート画面から移ってくると，「3」，「2」，「1」，「Start!!」とカウントダウンが表示される．

・カウントダウン終了後，ゲームが始まる

・制限時間が経過するか残金がなくなると，スタート画面に切り替わる

●Coupon

・クーポンを使える店の営業時間を記憶

●CouponPanel

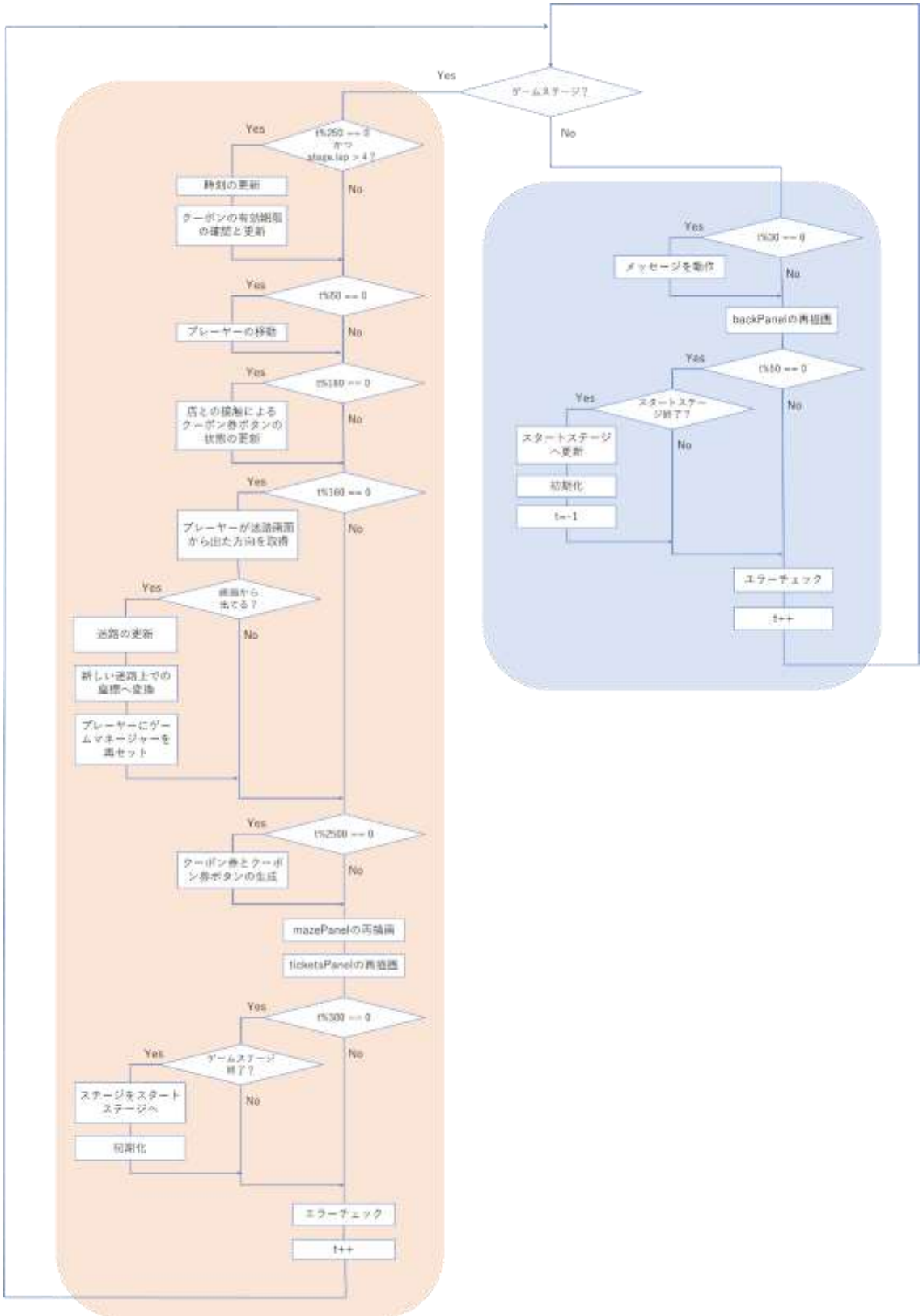
・接触している店が営業時間内であれば，その店のクーポンボタンを setEnable(true)に

・クーポンボタンがクリックされたら，残金，得した金額，現在時刻を更新し，デフォルトボタンに戻す

●TimePanel

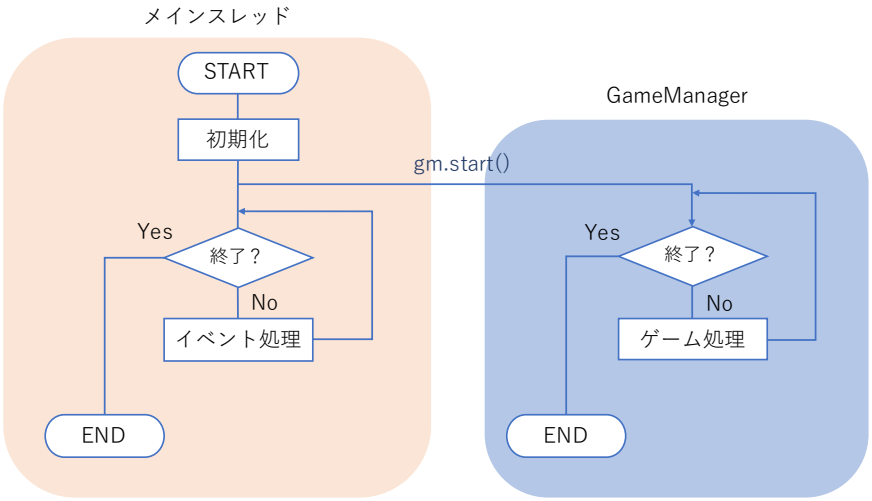
・現在時刻の下に残金と得した金額を表示

●ゲームマネージャー内のフローチャート



●全体のフローチャート

青で示された GameManager の処理が，上記で示したゲームマネージャー内のフローチャートに該当する．そのため，Game マネージャースレッドで終了することはなく，無限にループを繰り返す．



＜最終（SP4）のコードと実行結果＞

●TheDiscountTickets

```
/***
 *** The Discount Tickets
 ***/

import java.awt.*;
import javax.swing.*;

// TheDanmaku クラス
public class TheDiscountTickets extends JFrame {

    public TheDiscountTickets() {
        super("The Discount tickets");
        System.out.println("The Discount tickets");
        addNotify();
        Insets insets = getInsets();
```

```
        int width = 900 + insets.left + insets.right;
        int height = 600 + insets.top + insets.bottom;
        setSize(width, height);
        setLocation (0, 0);
        setDefaultCloseOperation (JFrame.EXIT_ON_CLOSE);

        BackPanel bp = new BackPanel();
        bp.setBounds (0, 0, 600, 900);
        bp.setLayout(null);

        MazePanel mp = new MazePanel();
        mp.setBounds(0, 0, 600, 600);

        TicketsPanel tp = new TicketsPanel();
        tp.setBounds(600, 0, 300, 600);

        bp.add(mp);
        bp.add(tp);
        add(bp);

        GameManager gm = new GameManager (mp, tp, bp);
        mp.setGM (gm);
        tp.setGM(gm);
        bp.setGM(gm);

        gm.start();
    }

    //メイン
    public static void main(String[] args) {
        (new TheDiscountTickets()).setVisible(true);
    }
}
```

●GameManager

```java
/***
 *** ゲーム管理者
 ***/

import java.awt.*;
import java.util.*;

public class GameManager extends Thread {
    private MazePanel mp; // 迷路パネル
    private TicketsPanel tp; // クーポン券パネル
    private BackPanel bp; // バックパネル
    private Maze maze; // 迷路
    public Player player; // プレーヤー
    public ArrayList<Wall> walls; // 壁リスト
    public ArrayList<Shop> shops; // 店リスト
    public ArrayList<Msg> msgs; // メッセージリスト
    public Stage stage; // ステージ
    public int hour; // 時
    public int minute; // 分
    public int day; // 日
    public int money; // 所持金
    public int score; // 得した金額･スコア
    public int pc; // プレイ回数

    // 生成
    public GameManager (MazePanel mp, TicketsPanel tp, BackPanel bp) {
        this.mp = mp;
        this.tp = tp;
        this.bp = bp;
        pc = 0;
        walls = new ArrayList<Wall>();
        shops = new ArrayList<Shop>();
        msgs=new ArrayList<Msg>(); // メッセージのリスト
        stage=new StageStart(this); // 最初のステージ
        initGame();
```

```
    }

    // 初期化
    public void initGame(){
        if(stage.stageN == 2){
            pc++;
            maze = new Maze_7(this);
            msgs.clear();
            mp.setVisible(true);
            tp.setVisible(true);
            player = new Player(120, 480);
            player.setGM(this);
            bp.addKeyListener(player);
            tp.couponPanel.set_default_All();
            hour = 6;
            minute = 0;
            day = 1;
            money = 20000;
            score = 0;
        }
        else{
            msgs.clear();
            mp.setVisible(false);
            tp.setVisible(false);
            bp.setVisible(true);
            bp.addKeyListener(stage);
        }
    }

    // 実行
    public void run() {
        long t = 0;

        while(true) {
            if(stage.stageN == 2){ // ゲーム中
                if(t % 250 == 0 && stage.lap > 4) {
```

```
                    time(); // 現在時刻の更新
                    tp.couponPanel.d_life(1); // クーポンの有効期限の確認と更
新
                }
                if(t % 50 == 0) player.move(); // プレーヤーの移動
                if(t % 160 == 0) tp.couponPanel.activeCoupon(); // 店との接
触によるクーポン券ボタンの状態の更新
                if(t % 200 == 0){
                    int n = player.out(); // プレーヤーが迷路画面から出ていない
か
                    if(n != 4){ // 出ていたら
                        maze = maze.next(n); // 迷路を更新
                        player.out_xy(n); // 次の迷路での座標へ変換
                        player.setGM(this); // ゲームマネージャーの再セット
                    }
                }
                if(t % 2500 == 0)tp.couponPanel.make_Coupon_Button(); // ク
ーポン券とクーポンボタンの生成
                mp.repaint(); // 迷路を再描画
                tp.repaint(); // クーポンパネルを再描画
                if(t % 300 == 0){ // ゲーム終了判定
                    if(!stage.proceed(t)){
                        stage=stage.next();
                        initGame();
                    }
                }
                try{
                    sleep(2);
                } catch (InterruptedException e){}
                t++;
            }
            else{ // スタート(スコア)画面
                if(t % 30 == 0) playAllMessages(); // 30 回に 1 回メッセージを
動作
                bp.repaint();
                if(t % 50 == 0){ // ゲームスタート判定
```

```
                if(!stage.proceed(t)){
                    stage=stage.next();
                    initGame();
                    t = -1;
                }
            }
            try{
                sleep(2);
            } catch (InterruptedException e){}
            t++;
        }
    }
}

// すべてのメッセージを動作
private void playAllMessages(){
    Iterator<Msg> it=msgs.iterator();
    while(it.hasNext()){
        Msg m=it.next();
        if(!m.play()){
            it.remove();
        }
    }
}

// 迷路パネルを得る
public BackPanel getBP(){
    return bp;
}

// 現在時刻
public void time(){
    minute += 5;
    if(minute >= 60){
        minute -= 60;
        hour += 1;
```

```
            if(hour >= 24){
                hour = 0;
                day += 1;
            }
        }
    }

    // メッセージの追加
    public void addMessage(Msg m){
        msgs.add(m);
    }

    // 描画
    public void draw (Graphics g) {
        player.draw(g); // プレイヤーを描画
        maze.draw(g);   // 迷路を描画
        drawMsgs(g);
    }

    public void drawMsgs(Graphics g){
        for(Msg m:msgs) m.draw(g,900,600); // すべてのメッセージを描画
    }
}
```

●MazePanel

```
/***
 *** 町の迷路
 ***/

import java.awt.*;
import javax.swing.*;

public class MazePanel extends JPanel {
    private GameManager gm;

    // 生成
```

```
    public MazePanel(){
        setBackground(Color.gray);
        setFocusable(true);
    }

    // ゲーム管理者のセット
    public void setGM(GameManager gm){
        this.gm = gm;
    }

    public void paintComponent(Graphics g){
        super.paintComponent(g);
        gm.draw(g);
    }
}
```

●TicketsPanel

```
/***
 *** クーポン券パネル
 ***/

import java.awt.*;
import javax.swing.*;

public class TicketsPanel extends JPanel{
    private GameManager gm;
    private TimePanel timePanel; // 現在時刻表示欄
    public CouponPanel couponPanel; // クーポン表示欄

    // 生成
    public TicketsPanel(){
        setBackground(Color.black);
    }

    // ゲーム管理者のセット
    public void setGM(GameManager gm){
```

```
        this.gm = gm;
        setLayout(null);
        timePanel = new TimePanel(gm);
        timePanel.setBounds(0, 0, 300, 100);
        add(timePanel);

        couponPanel = new CouponPanel(gm);
        couponPanel.setBounds(0, 100, 300, 500);
        add(couponPanel);
    }

    // 描画
    public void paintComponent(Graphics g){
        super.paintComponent(g);
        timePanel.paintComponents(g);
        couponPanel.paintComponents(g);
    }
}
```

●BackPanel

```
/***
 *** バックパネル
 ***/

import java.awt.*;
import javax.swing.*;

public class BackPanel extends JPanel {
    private GameManager gm;

    // 生成
    public BackPanel(){
        setBackground(Color.gray);
        setFocusable(true);
    }
```

```
    // ゲーム管理者のセット
    public void setGM(GameManager gm){
        this.gm = gm;
    }

    public void paintComponent(Graphics g){
        super.paintComponent(g);
        gm.drawMsgs(g);
    }
}
```

●TimePanel

```
/***
 *** 現在時刻を表示するパネル
 ***/

import java.awt.*;
import javax.swing.*;

public class TimePanel extends JPanel{
    private GameManager gm;

    // 生成
    public TimePanel(GameManager gm){
        this.gm = gm;
       setBackground(Color.cyan);
    }

    // 描画
    public void draw(Graphics g){
        String s_money= String.valueOf(gm.money);
        String z1 = "", z2 = "";

        if(gm.hour < 10){
            z1 = "0";
        }
```

```
        if(gm.minute < 10){
            z2 = "0";
        }
        if(gm.money < 0){
            s_money = "----";
        }

        g.setColor(Color.black);
        g.setFont(new Font(Font.DIALOG, Font.BOLD, 30));
        g.drawString(gm.day + "日目  " + z1 + gm.hour + ":" + z2 +
gm.minute, 62, 40);

        g.setFont(new Font(Font.DIALOG, Font.BOLD, 18));
        g.drawString("残金" + s_money + "円   得した金額" + gm.score + "円",
20, 80);
    }

    public void paintComponent(Graphics g){
        super.paintComponent(g);
        draw(g);
    }
}
```

●CouponPanel

```
/***
 *** クーポン券パネル
 ***/

import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.util.Random;
import java.util.ArrayList;

public class CouponPanel extends JPanel implements ActionListener{
```

```
    private GameManager gm;
    private JButton button[] = new JButton[5];
    private Coupon coupon[] = new Coupon[5];
    private ArrayList<Shop> shops;

    Random rd = new Random();
    Color b_Color = new Color(180, 180, 180); // ボタンのデフォルト色
    int a = 95; // ボタンの高さ(y)
    Integer c[] = new Integer[5]; // クーポンの店
    Integer life[] = new Integer[5] ;

    // 生成
    public CouponPanel(GameManager gm){
        this.gm = gm;
        this.shops = gm.shops;
        setBackground(Color.black);

        for(int i = 0; i < 5; i++){
            button[i] = new JButton();
            button[i].setActionCommand(String.valueOf(i));
            button[i].addActionListener(this);
            set_default(button[i]);
            life[i] = 0;
            add(button[i]);
            c[i] = 0;
        }
    }

    // クーポンボタンの初期化
    private void set_default(JButton bt){
        bt.setForeground(Color.black);
        bt.setFont(new Font(Font.DIALOG, Font.BOLD, 18));
        bt.setText("");
        bt.setBackground(b_Color);
        bt.setPreferredSize(new Dimension(300, a));
        bt.setEnabled(false);
```

```
    }

    // 全てのクーポンボタンの初期化
    public void set_default_All(){
        for(int i = 0; i < 5; i++){
            set_default(button[i]);
        }
    }

    //  クーポンとボタン生成
    public void make_Coupon_Button(){
        int n = 0;
        for(int i = 0; i < 5; i++){
            if(button[i].getText() == ""){
                if(n == 0){
                    System.out.println("クーポン生成");
                    coupon[i] = new Coupon(gm);
                    c[i] = coupon[i].s_no;
                    System.out.println(coupon[i].s_name + "  " +
coupon[i].name + "¥r¥n" + String.valueOf(coupon[i].price) + "  →  " +
String.valueOf(coupon[i].d_price));
                    life[i] = coupon[i].life;
                    button[i].setText("<html>" + coupon[i].s_name + "  " +
coupon[i].name + "<br/>" + String.valueOf(coupon[i].price) + "円   →  " +
String.valueOf(coupon[i].d_price) + "円<br>" + coupon[i].life_S);
                    button[i].setBackground(get_Shop_Color(coupon[i].s_no)
);
                    n ++;
                }
            }
        }
    }

    // クーポンの色を取得
    public Color get_Shop_Color(int s_no){
        Color color;
```

```
        switch(s_no){
            case 1 : color = new Color(40, 120, 255); break;
            case 2 : color = new Color(255, 150, 150); break;
            case 3 : color = new Color(255, 255, 80); break;
            case 4 : color = new Color(170, 80, 80); break;
            case 5 : color = new Color(220, 180, 50); break;
            case 6 : color = new Color(30, 220, 40); break;
            case 7 : color = new Color(100, 255, 150); break;
            case 8 : color = new Color(150, 150, 50); break;
            case 9 : color = new Color(220, 50, 50); break;
            case 10 : color = new Color(180, 80, 150); break;
            case 11 : color = new Color(80, 180, 180); break;
            default : color = new Color(0, 0, 0); break;
        }
        return color;
    }

    // 有効期限の確認
    public void d_life(int n){
        for(int i = 0; i < 5; i++){
            if(life[i] > 0){
                life[i] -= n;
            }
            if(life[i] <= 0){
                set_default(button[i]);
                c[i] = 0;
            }
        }
    }

    // 店との接触によるクーポン券の更新
    public void activeCoupon(){
        int no = gm.player.crush_PS();
        this.shops = gm.shops;
        int sArray_no = shops.size();
        System.out.println(no);
```

```
        System.out.println(sArray_no);
        if(no < sArray_no){
            int n = shops.get(no).no;
            System.out.println(n);
            for(int i = 0; i < 5; i++ ){
                if(c[i] == n){
                    if(coupon[i].min_time <= gm.hour && gm.hour <
coupon[i].max_time){
                        button[i].setEnabled(true);
                    }
                }
            }
        }else{
            for(int i = 0; i < 5; i++){
                button[i].setEnabled(false);
            }
        }
    }

    // 描画
    public void paintComponent(Graphics g){
        super.paintComponent(g);
    }

    // アクションリスナー
    public void actionPerformed(ActionEvent e){
        String e_button = e.getActionCommand();
        int n = Integer.valueOf(e_button).intValue();

        c[n] = 0;
        System.out.println(button[n]);
        if(gm.money > coupon[n].d_price){
            gm.score += coupon[n].price - coupon[n].d_price;
            gm.minute += coupon[n].c_time;
            d_life(coupon[n].c_time / 5);
        }
```

```
        gm.money -= coupon[n].d_price;

        set_default(button[n]);
        for(int i = n + 1; i < 5; i++){
            button[i].setFocusable(false);
        }
    }
}
```

●Coupon

```
/***
 *** クーポン券
 ***/
import java.util.Random;

public class Coupon {
    public String s_name = new String(); // 店舗名
    public String name = new String(); // 商品名
    public int price; // 定価
    public int d_price; // 割引価格
    public int life; // 有効期限
    public int s_no; // 使用可能店舗
    public String life_S; // 有効期限を示す文字列
    public int c_time; // 使用したときの経過時間
    public int min_time; // 使用可能開始時刻
    public int max_time; // 使用可能終了時刻

    GameManager gm;
    Random rd = new Random();
    int r;

    public Coupon(GameManager gm){
        this.gm = gm;
        if(gm.hour >= 10 && gm.hour < 18){
            r = rd.nextInt(11);
            switch(r){
```

```
                case 0 : make_RAWSON_C(); s_name = "RAWSON"; break;
                case 1 : make_INUQLO_C(); s_name = "INUQLO"; break;
                case 2 : make_Mk_C(); s_name = "モクドナルド"; break;
                case 3 : make_GANUTO_C(); s_name = "ガヌト"; break;
                case 4 : make_YOSHIDAYA_C(); s_name = "吉田家"; break;
                case 5 : make_MOON_C(); s_name = "ムーンバックス"; break;
                case 6 : make_MITORI_C(); s_name = "ミトリ"; break;
                case 7 : make_MATSUMOTO_C(); s_name = "マツモトキヨコ";
break;
                case 8 : make_TOHU_C(); s_name = "TOHU シネマズ"; break;
                case 9 : make_MINVALUE_C(); s_name = "ミニマムバリュー";
break;
                case 10 : make_YAMATA_C(); s_name = "YAMATA 電機"; break;
            }
        }
        else if(gm.hour >= 8 && gm.hour < 21){
            r = rd.nextInt(8);
            switch(r){
                case 0 : make_RAWSON_C(); s_name = "RAWSON"; break;
                case 1 : make_Mk_C(); s_name = "モクドナルド"; break;
                case 2 : make_GANUTO_C(); s_name = "ガヌト"; break;
                case 3 : make_YOSHIDAYA_C(); s_name = "吉田家"; break;
                case 4 : make_MOON_C(); s_name = "ムーンバックス"; break;
                case 5 : make_MATSUMOTO_C(); s_name = "マツモトキヨコ";
break;
                case 6 : make_TOHU_C(); s_name = "TOHU シネマズ"; break;
                case 7 : make_MINVALUE_C(); s_name = "ミニマムバリュー";
break;
            }
        }
        else{
            r = rd.nextInt(5);
            switch(r){
                case 0 : make_RAWSON_C(); s_name = "RAWSON"; break;
                case 1 : make_Mk_C(); s_name = "モクドナルド"; break;
                case 2 : make_GANUTO_C(); s_name = "ガヌト"; break;
```

```
                case 3 : make_YOSHIDAYA_C(); s_name = "吉田家"; break;
                case 4 : make_MINVALUE_C(); s_name = "ミニマムバリュー";
break;
            }
        }
    }

    public Coupon(GameManager gm, int n){
        make_COUPON(12, "", 0, 0, 10000, 0);
        s_name = "";
    }


    private void make_COUPON(int s_no, String name, int price, int
d_price, int life, int c_time){
        this.s_no = s_no;
        this.name = name;
        this.price = price;
        this.d_price = d_price;
        this.life = life;
        this.c_time = c_time;
        setLife();
    }

    private void make_RAWSON_C(){
        r = rd.nextInt(3);
        min_time = 0;
        max_time = 24;
        switch(r){
            case 0 : make_COUPON(1, "R チキ", 180, 160, 60, 5); break;
            case 1 : make_COUPON(1, "とんかつ弁当", 560, 520, 100, 5);
break;
            case 2 : make_COUPON(1, "アイスコーヒーS", 100, 0, 20, 5); break;
        }
    }

    private void make_INUQLO_C(){
```

```
        r = rd.nextInt(3);
        min_time = 10;
        max_time = 20;
        switch(r){
            case 0 : make_COUPON(2, "Tシャツ", 1380, 980, 60, 20); break;
            case 1 : make_COUPON(2, "ジーンズ", 3990, 2990, 40, 20); break;
            case 2 : make_COUPON(2, "コート", 5990, 4590, 25, 20); break;
        }
    }

    private void make_Mk_C(){
        r = rd.nextInt(3);
        min_time = 0;
        max_time = 24;
        switch(r){
            case 0 : make_COUPON(3, "コーラ S", 100, 0, 40, 10); break;
            case 1 : make_COUPON(3, "ポテト L", 330, 150, 60, 10); break;
            case 2 : make_COUPON(3, "ハンバーガーセット", 790, 490, 75, 10);
break;
        }
    }

    private void make_GANUTO_C(){
        r = rd.nextInt(3);
        min_time = 0;
        max_time = 24;
        switch(r){
            case 0 : make_COUPON(4, "ミニサラダ", 290, 260, 60, 10); break;
            case 1 : make_COUPON(4, "パフェ", 690, 620, 70, 15); break;
            case 2 : make_COUPON(4, "ハンバーグセット", 990, 860, 50, 30);
break;
        }
    }

    private void make_YOSHIDAYA_C(){
        r = rd.nextInt(3);
```

```
        min_time = 0;
        max_time = 24;
        switch(r){
            case 0 : make_COUPON(5, "牛丼", 430, 380, 60, 10); break;
            case 1 : make_COUPON(5, "温玉", 80, 0, 25, 5); break;
            case 2 : make_COUPON(5, "特盛牛丼", 780, 630, 50, 15); break;
        }
    }

    private void make_MOON_C(){
        r = rd.nextInt(3);
        min_time = 8;
        max_time = 23;
        switch(r){
            case 0 : make_COUPON(6, "ドリップコーヒー Short", 310, 0, 15, 5);
break;
            case 1 : make_COUPON(6, "カフェラテ Tall", 420, 380, 75, 5);
break;
            case 2 : make_COUPON(6, "フラッペ Grande", 580, 460, 50, 5);
break;
        }
    }

    private void make_MITORI_C(){
        r = rd.nextInt(3);
        min_time = 11;
        max_time = 20;
        switch(r){
            case 0 : make_COUPON(7, "クッション", 680, 560, 75, 20); break;
            case 1 : make_COUPON(7, "デスク", 5980, 4280, 50, 30); break;
            case 2 : make_COUPON(7, "ソファ", 8980, 5980, 30, 45); break;
        }
    }

    private void make_MATSUMOTO_C(){
        r = rd.nextInt(3);
```

```
        min_time = 9;
        max_time = 23;
        switch(r){
            case 0 : make_COUPON(8, "エナジードリンク", 210, 0, 15, 5);
break;
            case 1 : make_COUPON(8, "シャンプー", 480, 340, 60, 5); break;
            case 2 : make_COUPON(8, "日焼け止めクリーム", 850, 690, 50, 5);
break;
        }
    }

    private void make_TOHU_C(){
        r = rd.nextInt(3);
        min_time = 8;
        max_time = 22;
        switch(r){
            case 0 : make_COUPON(9, "ソフトドリンク", 380, 190, 60, 5); break;
            case 1 : make_COUPON(9, "ポップコーン", 490, 290, 50, 5); break;
            case 2 : make_COUPON(9, "無料鑑賞券", 1900, 0, 12, 120); break;
        }
    }

    private void make_MINVALUE_C(){
        r = rd.nextInt(3);
        min_time = 0;
        max_time = 24;
        switch(r){
            case 0 : make_COUPON(10, "から揚げ", 380, 190, 60, 15); break;
            case 1 : make_COUPON(10, "みかん", 290, 100, 50, 15); break;
            case 2 : make_COUPON(10, "国産牛肉", 2990, 1590, 30, 15);
break;
        }
    }

    private void make_YAMATA_C(){
        r = rd.nextInt(3);
```

```
        min_time = 10;
        max_time = 20;
        switch(r){
            case 0 : make_COUPON(11, "乾電池", 780, 490, 80, 15); break;
            case 1 : make_COUPON(11, "ヘッドホン", 12990, 7990, 50, 45);
break;
            case 2 : make_COUPON(11, "電子レンジ", 21700, 13800, 30, 60);
break;
        }
    }

    private void setLife(){
        int m = gm.minute + life * 5 ;
        int life_m = m % 60;
        int h = gm.hour + (m - life_m) / 60;
        int life_h = h % 24;
        int life_d = gm.day + (h - life_h) / 24;
        String z1 = "", z2 = "";
/*
        if(life_h >= max_time || life_h < min_time){
            life_h = max_time;
            life_m = 0;
        }*/
        if(life_h < 10){
            z1 = "0";
        }
        if(life_m < 10){
            z2 = "0";
        }

        life_S = (life_d + "日目  " + z1 + life_h + ":" + z2 + life_m + "
まで有効");
    }
}
```

●Stage

```
/***
 *** ステージの基底クラス
 ***/

import java.awt.event.*;

public abstract class Stage implements KeyListener {
    public int lap; // 経過時間
    protected int max; // 最大時間
    protected GameManager gm; // ゲーム管理者
    public boolean start;
    public int stageN;

    // 生成
    public Stage(GameManager gm){
      lap=0;
      max=0;
      this.gm=gm;
    }

    // 進行
    public abstract boolean proceed(long t);

    // 次のステージ
    public abstract Stage next();

    // キーリスナー
    public abstract void keyPressed(KeyEvent ev);
    public abstract void keyReleased(KeyEvent ev);
    public abstract void keyTyped(KeyEvent ev);
}
```

●StageStart

```
/***
 *** ゲームスタート画面・スコア画面
 ***/
import java.awt.*;
import java.awt.event.*;

public class StageStart extends Stage {

  // 生成
  public StageStart(GameManager gm){
    super(gm);
    start=false;
    stageN = 1;
  }

  // 進行
  public boolean proceed(long t){
    if(lap==0){
      if(gm.pc == 0){ // 初めてのプレイ
        gm.addMessage(new MsgBrink("T h e  D i s c o u n t  T i c k e t
s",200,Color.orange,32));
        gm.addMessage(new MsgBrink("Press S key to
Start !!",350,Color.white,24));
      }else{ // 2回目以上のプレイ(プレイ終了後)
        gm.addMessage(new MsgBrink("S c o r e  :  " + gm.score + "円
",200,Color.red,32));
        gm.addMessage(new MsgBrink("Press S key to
Restart !!",350,Color.white,24));
      }
    }
    lap++;
    return !start;
  }

  // 次のステージ
```

```
  public Stage next(){
    return new StageGame(gm);
  }

  // キーリスナー
  public void keyPressed(KeyEvent ev){
    if(ev.getKeyCode()==KeyEvent.VK_S){
      start=true;
    }
  }
  public void keyReleased(KeyEvent ev){}
  public void keyTyped(KeyEvent ev){}
}
```

●StageGame

```
/***
 *** ゲームステージ
 ***/
import java.awt.*;
import java.awt.event.*;

public class StageGame extends Stage {

  // 生成
  public StageGame(GameManager gm){
    super(gm);
    stageN = 2;
  }

  // 進行
  public boolean proceed(long t){
    if(lap==0){
        gm.msgs.clear();
      gm.addMessage(new MsgBrink("3",200,Color.cyan,100));
    }
```

```
    if(lap==1){
        gm.msgs.clear();
      gm.addMessage(new MsgBrink("2",200,Color.cyan,100));
    }
    if(lap==2){
        gm.msgs.clear();
      gm.addMessage(new MsgBrink("1",200,Color.cyan,100));
    }
    if(lap==3){
        gm.msgs.clear();
      gm.addMessage(new MsgBrink("Start!!",200,Color.cyan,100));
    }
    if(lap == 4){
        gm.msgs.clear();
        gm.addMessage(new MsgBrink("",200,Color.cyan,100));
    }
    lap++;
    return checkTime(gm.day, gm.hour, gm.minute) && gm.money > 0;
  }

  // 次のステージ
  public Stage next(){
    return new StageStart(gm);
  }

  // 制限時間の確認
  private boolean checkTime(int day, int hour, int minute){
    return !(day >=4 && hour >= 6 && minute >= 0);
  }

  // キーリスナー
  public void keyPressed(KeyEvent ev){}
  public void keyReleased(KeyEvent ev){}
  public void keyTyped(KeyEvent ev){}

}
```

●Msg

```
/***
 *** メッセージの基底クラス
 ***/

import java.awt.*;

public abstract class Msg {
    protected int age; // 年齢
    protected int life; // 寿命
    protected boolean alive; // 生存フラグ

    // 生成
    public Msg() {
        life = 0; // 寿命無し
        alive = true; // 生きている
    }

    // 描画
    public abstract void draw (Graphics g, int w, int h);

    // 動作
    public boolean play () {
        age++; // 年齢を増やす
        if (life > 0) // 寿命が設定してあるなら、
          alive = (age < life); // 寿命が来たら死ぬ
        return alive; // 生きているかどうか
    }

    // 死ぬ
    public void die () {
        alive = false;
    }

    // 左右中央に描画
    // Font はセットしてある
```

```
    protected void drawCenter (Graphics g, String s, int w, int y) {
        FontMetrics fm = g.getFontMetrics ();
        int sw = fm.stringWidth(s);
        int sh = fm.getHeight();
        int x = (w - sw) / 2;
        y += sh / 2;
        g.drawString (s, x, y);
    }
}
```

●MsgBrink

```
/*** 点滅メッセージ
 *** Y 座標、色、大きさを指定し、点滅するメッセージ
 *** 死なない限り点滅し続ける
 ***/

import java.awt.*;

public class MsgBrink extends Msg {
    private String msg; // メッセージ文字列
    private Color col; // 色
    private Font font; // フォント
    private int py; // y 座標
    private int on, off; // 点灯・消灯時間

    // 生成
    public MsgBrink (String m, int y, Color c, int s){
        this (m,y,c,s,10,5, 0);
    }
    public MsgBrink (String m, int y, Color c, int s, int on, int off,
int l){
        msg = m;
        py = y;
        col = c;
        font = new Font ("", Font.PLAIN, s);
        this.on = on;
```

```
        this.off = off;
        life = l;
        age = 0;
    }

    // 描画
    public void draw (Graphics g, int w, int h) {
        if(age % (on + off) < on) { // 点灯しているタイミングなら、
            g.setColor (col); // 色と
            g.setFont (font); // フォントをセットして
            drawCenter (g,msg, w, py); // 左右中央に描画
        }
    }
}
```

●GameChar

```
/***
 *** ゲームキャラクター
 ***/

 public abstract class GameChar {
    protected int px,py; // 位置
    protected int cx, cy; // 判定領域の中心
    protected int cw,ch; // 判定領域の大きさ

    // 生成
    public GameChar(){
      this(0,0);
    }

    public GameChar(int x, int y){
      px=x; py=y;
      cx=x; cy=y;
      cw=ch=0; // 最初の大きさは 0x0
    }
```

```
    // 判定領域の中心の設定
    public void setCenter(int dx, int dy){
      cx = dx;
      cy = dy;
    }

    // 大きさ設定
    public void setSize(int w, int h){
      cw=w;
      ch=h;
    }

    // 位置を得る
    public double getX(){ return px; }
    public double getY(){ return py; }

    // 重なり判定
    public boolean overlap(GameChar c){
      double x11=cx-cw/2,y11=cy-ch/2;
      double x12=cx+cw/2,y12=cy+ch/2;
      double x21=c.cx-c.cw/2,y21=c.cy-c.ch/2;
      double x22=c.cx+c.cw/2,y22=c.cy+c.ch/2;
      return
      (((x11<=x21 && x21<x12)||(x21<=x11 && x11<x22)) &&
       ((y11<=y21 && y21<y12)||(y21<=y11 && y11<y22)));
    }
}
```

●Player

```
/***
 *** プレーヤークラス
 ***/

import java.awt.*;
import java.util.*;
import java.awt.event.*;
```

```
public class Player extends GameChar implements KeyListener{

    // 座標(x, y)は GameChar が所持
    private int dx, dy; // 移動量
    private GameManager gm;
    private ArrayList<Wall> walls;
    private ArrayList<Shop> shops;
    private int dir; // 方向
    private int a = 20;

    // 生成
    public Player (int x, int y){
        px = x;
        py = y;
        dx = 0;
        dy = 0;
        dir = 0;
        setSize(5, 5); // 衝突判定の大きさ
        setCenter(px, py - 15); // 衝突判定の中心(三角形の先端)
        walls = new ArrayList<Wall>();
        shops = new ArrayList<Shop>();
    }

    // ゲームマネージャーのセット
    public void setGM(GameManager gm){
        this.gm = gm;
        this.walls = gm.walls;
        this.shops = gm.shops;
    }

    // 移動
    public void move() {
        System.out.println(dx +" "+ dy);
        px += dx;
        py += dy;
```

```
        dx = 0;
        dy = 0;
    }

    // パネル外へ出た方向の判定
    public int out(){
        if(py < 0) return 0;
        if(py > 600) return 1;
        if(px > 600) return 2;
        if(px < 0) return 3;
        else return 4;
    }

    // プレイヤーがウィンドウから出た後の座標
    public void out_xy(int n){
        switch(n){
            case 0 : py += 600; break;
            case 1 : py -= 600; break;
            case 2 : px -= 600; break;
            case 3 : px += 600; break;
            default : break;
        }
    }

    // 店との衝突判定
    public int crush_PS(){
        int no = 0;
        for(Shop s:shops){
            System.out.println(overlap(s));
            if(overlap(s) == true){
                return no;
            }
            no++;
        }
        return no;
    }
```

```
// 描画
public void draw (Graphics g) {
    int[] x = new int[3];
    int[] y = new int[3];
    if(dir == 0){ // 上
        x[0] = (int)px;
        x[1] = (int)px - 10;
        x[2] = (int)px + 10;
        y[0] = (int)py - 15;
        y[1] = (int)py + 10;
        y[2] = (int)py + 10;
    }else if(dir == 1){ // 下
        x[0] = (int)px;
        x[1] = (int)px - 10;
        x[2] = (int)px + 10;
        y[0] = (int)py + 15;
        y[1] = (int)py - 10;
        y[2] = (int)py - 10;
    }else if(dir == 2){ // 右
        x[0] = (int)px + 15;
        x[1] = (int)px - 10;
        x[2] = (int)px - 10;
        y[0] = (int)py;
        y[1] = (int)py + 10;
        y[2] = (int)py - 10;
    }else{ // 左
        x[0] = (int)px - 15;
        x[1] = (int)px + 10;
        x[2] = (int)px + 10;
        y[0] = (int)py;
        y[1] = (int)py + 10;
        y[2] = (int)py - 10;
    }
    setCenter(x[0], y[0]); // 衝突判定中心の更新
    g.setColor (new Color (255,255, 100)); // 明るい黄色で
```

```
        g.fillPolygon (x, y, 3);                // 塗りつぶして
        g.setColor (new Color (255,200,20) );  // 暗い黄色で
        g.drawPolygon (x, y, 3);                // 枠を描く
    }

    // キーリスナー
    public void keyTyped(KeyEvent ev) {}

    public void keyPressed(KeyEvent ev) {
        if(gm.stage.lap > 3){
            if(ev.getKeyCode()==KeyEvent.VK_UP){
                if(dir == 0){
                    dy = -1 * a;
                    for(Wall w:walls){
                        if(overlap(w) == true){
                            dy = 0;
                            break;
                        }
                    }
                }
                dir = 0;
            }
            if(ev.getKeyCode()==KeyEvent.VK_DOWN){
                if(dir == 1){
                    dy = a;
                    for(Wall w:walls){
                        if(overlap(w) == true){
                        dy = 0;
                        break;
                        }
                    }
                }
                dir = 1;
            }
            if(ev.getKeyCode()==KeyEvent.VK_RIGHT){
                if(dir == 2){
```

```
                    dx = a;
                    for(Wall w:walls){
                        if(overlap(w) == true){
                            dx = 0;
                            break;
                        }
                    }
                }
                dir = 2;
            }
            if(ev.getKeyCode()==KeyEvent.VK_LEFT){
                if(dir == 3){
                    dx = -1 * a;
                    for(Wall w:walls){
                        if(overlap(w) == true){
                            dx = 0;
                            break;
                        }
                    }
                }
                dir = 3;
            }
        }
    }

    public void keyReleased(KeyEvent ev) {}
}
```

●Shop

```
/***
 *** 店
 ***/
import java.awt.*;

public class Shop extends GameChar{
    public Color color;
```

```
    private int n; // 店の大きさ
    public int no; // 店番号
    private String s_shop; // 店名

    public Shop(int s_no, int x, int y){
        px=x; py=y;
        cx=x; cy=y;
        this.no = s_no;

        switch(no){
            case 1 : make_RAWSON(); break;
            case 2 : make_INUQLO(); break;
            case 3 : make_Mk(); break;
            case 4 : make_GANUTO(); break;
            case 5 : make_YOSHIDAYA(); break;
            case 6 : make_MOON(); break;
            case 7 : make_MITORI(); break;
            case 8 : make_MATSUMOTO(); break;
            case 9 : make_TOHU(); break;
            case 10 : make_MINVALUE(); break;
            case 11 : make_YAMATA(); break;
        }
    }

    private void make_RAWSON(){ // 1
        n = 1;
        setSize(100, 100);
        color = new Color(40, 120, 255);
        s_shop = new String("RAWSON");
    }

    private void make_INUQLO(){ // 2
        n = 2;
        setSize(200, 200);
        color = new Color(255, 150, 150);
        s_shop = new String("INUQLO");
```

```
    }

    private void make_Mk(){ // 3
        n = 1;
        setSize(100, 100);
        color = new Color(255, 255, 80);
        s_shop = new String("モクドナルド");
    }

    private void make_GANUTO(){ // 4
        n = 1;
        setSize(100, 100);
        color = new Color(170, 80, 80);
        s_shop = new String("ガヌト");
    }

    private void make_YOSHIDAYA(){ // 5
        n = 1;
        setSize(100, 100);
        color = new Color(220, 180, 50);
        s_shop = new String("吉田家");
    }

    private void make_MOON(){ // 6
        n = 1;
        setSize(100, 100);
        color = new Color(30, 220, 40);
        s_shop = new String("ムーンバックス");
    }

    private void make_MITORI(){ // 7
        n = 2;
        setSize(200, 200);
        color = new Color(100, 255, 150);
        s_shop = new String("ミトリ");
    }
```

```
    private void make_MATSUMOTO(){ // 8
        n = 2;
        setSize(200, 200);
        color = new Color(150, 150, 50);
        s_shop = new String("マツモトキヨコ");
    }

    private void make_TOHU(){ // 9
        n = 2;
        setSize(200, 200);
        color = new Color(220, 50, 50);
        s_shop = new String("TOHU シネマズ");
    }

    private void make_MINVALUE(){ // 10
        n = 2;
        setSize(200, 200);
        color = new Color(180, 80, 150);
        s_shop = new String("ミニマムバリュー");
    }

    private void make_YAMATA(){ // 11
        n = 2;
        setSize(200, 200);
        color = new Color(80, 180, 180);
        s_shop = new String("YAMATA 電機");
    }

    // 描画
    public void draw(Graphics g){
        int[] x1 = new int[3];
        int[] y1 = new int[3];
        x1[0] = (int)px;
        y1[0] = (int)py - n * 30;
        x1[1] = (int)px + n * 40;
```

```
        y1[1] = (int)py - n * 5;
        x1[2] = (int)px - n * 40;
        y1[2] = (int)py - n * 5;

        int x2 = px - n * 30;
        int y2 = py - n * 5;

        FontMetrics fm = g.getFontMetrics();
        Rectangle rectText = fm.getStringBounds(s_shop, g).getBounds();

        int x3 = px - rectText.width/2;
        int y3 = py + n * 20 - rectText.height/2;

        g.setColor (this.color);
        g.fillPolygon (x1 , y1, 3);
        g.fillRect(x2, y2, n * 60, n * 45);

        g.setColor(Color.black);
        g.drawString(s_shop, x3, y3);
    }
}
```

●Wall

```
/***
 *** 迷路の壁
 ***/

import java.awt.*;

public class Wall extends GameChar{
    private int dx, dy;

    // 生成
    public Wall(int x, int y, int dx, int dy){
        this.px = x;
        this.py = y;
```

```
        this.dx = dx;
        this.dy = dy;
        setCenter(x, y); // 衝突判定の中心
        setSize(dx + 10, dy + 10); // 衝突判定の範囲
    }

    // 描画
    public void draw(Graphics g){
        g.setColor(Color.BLACK);
        g.fillRect((int)(px - dx/2), (int)(py - dy/2), (int)dx, (int)dy);
    }
}
```

●Maze

```
/***
 *** Maze1～9 の基底クラス
 ***/
import java.awt.*;

public abstract class Maze{
    private GameManager gm;

    // 生成
    public Maze(GameManager gm){
        this.gm = gm;
    }

     // 描画
    public abstract void draw(Graphics g);

     // 次のゲームパネル
    public abstract Maze next(int n);
}
```

●Maze_1

```
/***
 *** Maze1
 ***/
import java.awt.*;
import java.util.ArrayList;

public class Maze_1 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[15];
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;
    private Shop shop[] = new Shop[4];
    private ArrayList<Shop> shops = new ArrayList<Shop>();

    // 生成
    public Maze_1(GameManager gm){
        super(gm);
        this.gm = gm;
        wall[0] = new Wall(300, 0, 600, a);
        wall[1] = new Wall(0, 300, a, 600);
        wall[2] = new Wall(120, 600, 240, a);
        wall[3] = new Wall(420, 600, 120, a);
        wall[4] = new Wall(600, 120, a, 240);
        wall[5] = new Wall(600, 480, a, 240);

        wall[6] = new Wall(180, 240, 120, a);
        wall[7] = new Wall(420, 240, 120, a);
        wall[8] = new Wall(60, 360, 120, a);
        wall[9] = new Wall(360, 480, 240, a);

        wall[10] = new Wall(360, 120, a, 240);
        wall[11] = new Wall(240, 240, a, 240);
        wall[12] = new Wall(120, 420, a, 120);
        wall[13] = new Wall(360, 420, a, 120);
        wall[14] = new Wall(480, 420, a, 360);
```

```
        for(int i= 0; i < 15; i++){
            walls.add(wall[i]);
        }

        gm.walls = this.walls;

        shop[0] = new Shop(8, 480, 120);  // マツモトキヨコ
        shop[1] = new Shop(11, 120, 120); // YAMATA 電機
        shop[2] = new Shop(1, 60, 420); // RAWSON
        shop[3] = new Shop(3, 420, 420); // モクドナルド

        for(int i= 0; i < 4; i++){
            shops.add(shop[i]);
        }

        gm.shops = this.shops;
    }

     // 描画
     public void draw(Graphics g){
        for(Wall w:walls) w.draw(g);
        for(Shop s:shops) s. draw(g);
    }

     // 次のゲームパネル
    public Maze next(int n){
        switch(n){
            case 0 : // なし
            case 1 : return new Maze_4(gm);
            case 2 : return new Maze_2(gm);
            case 3 : // なし
            default : return this;
        }
    }
}
```

●Maze_2

```
/***
 *** Maze2
 ***/
import java.awt.*;
import java.util.ArrayList;

public class Maze_2 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[17];
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;
    private Shop shop[] = new Shop[3];
    private ArrayList<Shop> shops = new ArrayList<Shop>();

    // 生成
    public Maze_2(GameManager gm){
        super(gm);
        this.gm = gm;
        wall[0] = new Wall(300, 0, 600, a);
        wall[1] = new Wall(0, 120, a, 240);
        wall[2] = new Wall(0, 480, a, 240);
        wall[3] = new Wall(300, 600, 600, a);
        wall[4] = new Wall(600, 180, a, 360);
        wall[5] = new Wall(600, 540, a, 120);

        wall[6] = new Wall(540, 120, 120, a);
        wall[7] = new Wall(300, 240, 120, a);
        wall[8] = new Wall(420, 360, 120, a);
        wall[9] = new Wall(180, 360, 120, a);
        wall[10] = new Wall(300, 480, 120, a);

        wall[11] = new Wall(240, 180, a, 120);
        wall[12] = new Wall(480, 180, a, 120);
        wall[13] = new Wall(120, 300, a, 120);
        wall[14] = new Wall(360, 300, a, 120);
```

```
        wall[15] = new Wall(480, 480, a, 240);
        wall[16] = new Wall(240, 540, a, 120);

        for(int i= 0; i < 17; i++){
            walls.add(wall[i]);
        }

        gm.walls = this.walls;

        shop[0] = new Shop(7, 120, 480); // ミトリ
        shop[1] = new Shop(5, 540, 60); // 吉田家
        shop[2] = new Shop(4, 300, 540); // ガヌト

        for(int i= 0; i < 3; i++){
            shops.add(shop[i]);
        }

        gm.shops = this.shops;
    }

     // 描画
     public void draw(Graphics g){
        for(Wall w:walls) w.draw(g);
        for(Shop s:shops) s. draw(g);
    }

     // 次のゲームパネル
    public Maze next(int n){
        switch(n){
            case 0 : // 無し
            case 1 : return new Maze_5(gm);
            case 2 : return new Maze_3(gm);
            case 3 : return new Maze_1(gm);
            default : return this;
        }
```

```
    }
}
```

●Maze_3

```
/***
 *** Maze3
 ***/
import java.awt.*;
import java.util.ArrayList;

public class Maze_3 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[13];
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;
    private Shop shop[] = new Shop[3];
    private ArrayList<Shop> shops = new ArrayList<Shop>();

    // 生成
    public Maze_3(GameManager gm){
        super(gm);
        this.gm = gm;
        wall[0] = new Wall(300, 0, 600, a);
        wall[1] = new Wall(0, 180, a, 360);
        wall[2] = new Wall(0, 540, a, 120);
        wall[3] = new Wall(60, 600, 120, a);
        wall[4] = new Wall(420, 600, 360, a);
        wall[5] = new Wall(600, 300, a, 600);

        wall[6] = new Wall(300, 240, 360, a);
        wall[7] = new Wall(180, 480, 120, a);

        wall[8] = new Wall(360, 120, a, 240);
        wall[9] = new Wall(120, 240, a, 240);
        wall[10] = new Wall(480, 360, a, 240);
        wall[11] = new Wall(360, 480, a, 240);
```

```
        wall[12] = new Wall(120, 540, a, 120);

        for(int i= 0; i < 13; i++){
            walls.add(wall[i]);
        }

        gm.walls = this.walls;

        shop[0] = new Shop(1, 60, 540); // RAWSON
        shop[1] = new Shop(9, 480, 120); // TOHU シネマズ
        shop[2] = new Shop(10, 240, 120); // ミニマムバリュー

        for(int i= 0; i < 3; i++){
            shops.add(shop[i]);
        }

        gm.shops = this.shops;
    }

     // 描画
     public void draw(Graphics g){
        for(Wall w:walls) w.draw(g);
        for(Shop s:shops) s. draw(g);
    }

     // 次のゲームパネル
    public Maze next(int n){
        switch(n){
            case 0 : // なし
            case 1 : return new Maze_6(gm);
            case 2 : // なし
            case 3 : return new Maze_2(gm);
            default : return this;
        }
    }
}
```

●Maze_4

```
/***
 *** Maze4
 ***/
import java.awt.*;
import java.util.ArrayList;

public class Maze_4 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[19];
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;
    private Shop shop[] = new Shop[3];
    private ArrayList<Shop> shops = new ArrayList<Shop>();

    // 生成
    public Maze_4(GameManager gm){
        super(gm);
        this.gm = gm;
        wall[0] = new Wall(120, 0, 240, a);
        wall[1] = new Wall(420, 0, 120, a);
        wall[2] = new Wall(0, 300, a, 600);
        wall[3] = new Wall(180, 600, 120, a);
        wall[4] = new Wall(420, 600, 120, a);
        wall[5] = new Wall(600, 120, a, 240);
        wall[6] = new Wall(600, 480, a, 240);

        wall[7] = new Wall(360, 120, 480, a);
        wall[8] = new Wall(540, 240, 120, a);
        wall[9] = new Wall(60, 360, 120, a);
        wall[10] = new Wall(300, 360, 120, a);
        wall[11] = new Wall(540, 360, 120, a);
        wall[12] = new Wall(420, 480, 120, a);

        wall[13] = new Wall(240, 60, a, 120);
        wall[14] = new Wall(360, 180, a, 120);
```

```
        wall[15] = new Wall(120, 180, a, 120);
        wall[16] = new Wall(480, 420, a, 120);
        wall[17] = new Wall(240, 480, a, 240);
        wall[18] = new Wall(360, 540, a, 120);


        for(int i= 0; i < 19; i++){
            walls.add(wall[i]);
        }

        gm.walls = this.walls;

        shop[0] = new Shop(3, 180, 60); // モクドナルド
        shop[1] = new Shop(1, 540, 180); // RAWSON
        shop[2] = new Shop(6, 540, 420); // ムーンバックス

        for(int i= 0; i < 3; i++){
            shops.add(shop[i]);
        }

        gm.shops = this.shops;
    }

     // 描画
     public void draw(Graphics g){
        for(Wall w:walls) w.draw(g);
        for(Shop s:shops) s. draw(g);
    }

     // 次のゲームパネル
    public Maze next(int n){
        switch(n){
            case 0 : return new Maze_1(gm);
            case 1 : return new Maze_7(gm);
            case 2 : return new Maze_5(gm);
            case 3 : // なし
```

```
            default : return this;
        }
    }
}
```

●Maze_5

```
/***
 *** Maze5
 ***/
import java.awt.*;
import java.util.ArrayList;

public class Maze_5 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[18];
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;
    private Shop shop[] = new Shop[3];
    private ArrayList<Shop> shops = new ArrayList<Shop>();

    // 生成
    public Maze_5(GameManager gm){
        super(gm);
        this.gm = gm;
        wall[0] = new Wall(300, 0, 600, a);
        wall[1] = new Wall(0, 120, a, 240);
        wall[2] = new Wall(0, 480, a, 240);
        wall[3] = new Wall(180, 600, 360, a);
        wall[4] = new Wall(540, 600, 120, a);
        wall[5] = new Wall(600, 180, a, 360);
        wall[6] = new Wall(600, 540, a, 120);

        wall[7] = new Wall(180, 120, 120, a);
        wall[8] = new Wall(360, 240, 240, a);
        wall[9] = new Wall(60, 240, 120, a);
```

```
        wall[10] = new Wall(540, 360, 120, a);
        wall[11] = new Wall(240, 480, 240, a);
        wall[12] = new Wall(540, 480, 120, a);

        wall[13] = new Wall(240, 60, a, 120);
        wall[14] = new Wall(360, 120, a, 240);
        wall[15] = new Wall(120, 300, a, 120);
        wall[16] = new Wall(240, 300, a, 120);
        wall[17] = new Wall(360, 540, a, 120);

        for(int i= 0; i < 18; i++){
            walls.add(wall[i]);
        }

        gm.walls = this.walls;

        shop[0] = new Shop(10, 480, 120); // ミニマムバリュー
        shop[1] = new Shop(6, 180, 60); // ムーンバックス
        shop[2] = new Shop(5, 300, 540); // 吉田家

        for(int i= 0; i < 3; i++){
            shops.add(shop[i]);

        }

        gm.shops = this.shops;
    }

     // 描画
     public void draw(Graphics g){
        for(Wall w:walls) w.draw(g);
        for(Shop s:shops) s. draw(g);
    }

     // 次のゲームパネル
    public Maze next(int n){
```

```
        switch(n){
            case 0 : return new Maze_2(gm);
            case 1 : return new Maze_8(gm);
            case 2 : return new Maze_6(gm);
            case 3 : return new Maze_4(gm);
            default : return this;
        }
    }
}
```

●Maze_6

```
/***
 *** Maze6
 ***/
import java.awt.*;
import java.util.ArrayList;

public class Maze_6 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[19];
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;
    private Shop shop[] = new Shop[3];
    private ArrayList<Shop> shops = new ArrayList<Shop>();

    // 生成
    public Maze_6(GameManager gm){
        super(gm);
        this.gm = gm;
        wall[0] = new Wall(0, 180, a, 360);
        wall[1] = new Wall(0, 540, a, 120);
        wall[2] = new Wall(60, 600, 120, a);
        wall[3] = new Wall(420, 600, 360, a);
        wall[4] = new Wall(600, 300, a, 600);
        wall[5] = new Wall(420, 0, 360, a);
```

```
        wall[6] = new Wall(60, 0, 120, a);

        wall[7] = new Wall(540, 240, 120, a);
        wall[8] = new Wall(300, 360, 120, a);
        wall[9] = new Wall(180, 480, 120, a);
        wall[10] = new Wall(420, 480, 120, a);

        wall[11] = new Wall(120, 180, a, 360);
        wall[12] = new Wall(120, 540, a, 120);
        wall[13] = new Wall(240, 60, a, 120);
        wall[14] = new Wall(360, 120, a, 240);
        wall[15] = new Wall(480, 300, a, 120);
        wall[16] = new Wall(240, 360, a, 240);
        wall[17] = new Wall(120, 540, a, 120);
        wall[18] = new Wall(480, 540, a, 120);

        for(int i= 0; i < 19; i++){
            walls.add(wall[i]);
        }

        gm.walls = this.walls;

        shop[0] = new Shop(2, 480, 120); // INUQLO
        shop[1] = new Shop(3, 60, 60); // モクドナルド
        shop[2] = new Shop(4, 540, 300); // ガヌト

        for(int i= 0; i < 3; i++){
            shops.add(shop[i]);
        }

        gm.shops = this.shops;
    }

     // 描画
     public void draw(Graphics g){
        for(Wall w:walls) w.draw(g);
```

```
        for(Shop s:shops) s. draw(g);
    }

     // 次のゲームパネル
    public Maze next(int n){
        switch(n){
            case 0 : return new Maze_3(gm);
            case 1 : return new Maze_9(gm);
            case 2 : // なし
            case 3 : return new Maze_5(gm);
            default : return this;
        }
    }
}
```

●Maze_7

```
/***
 *** Maze7
 ***/
import java.awt.*;
import java.util.ArrayList;

public class Maze_7 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[18];
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;
    private Shop shop[] = new Shop[3];
    private ArrayList<Shop> shops = new ArrayList<Shop>();

    // 生成
    public Maze_7(GameManager gm){
        super(gm);
        this.gm = gm;
        wall[0] = new Wall(0, 300, a, 600);
```

```
        wall[1] = new Wall(300, 600, 600, a);
        wall[2] = new Wall(600, 540, a, 120);
        wall[3] = new Wall(600, 240, a, 240);
        wall[4] = new Wall(420, 0, 120, a);
        wall[5] = new Wall(180, 0, 120, a);

        wall[6] = new Wall(300, 120, 120, a);
        wall[7] = new Wall(540, 120, 120, a);
        wall[8] = new Wall(180, 240, 120, a);
        wall[9] = new Wall(60, 360, 120, a);
        wall[10] = new Wall(420, 360, 120, a);

        wall[11] = new Wall(120, 120, a, 240);
        wall[12] = new Wall(360, 60, a, 120);
        wall[13] = new Wall(480, 60, a, 120);
        wall[14] = new Wall(240, 180, a, 120);
        wall[15] = new Wall(360, 360, a, 240);
        wall[16] = new Wall(240, 480, a, 240);
        wall[17] = new Wall(480, 480, a, 240);

        for(int i= 0; i < 18; i++){
            walls.add(wall[i]);
        }

        gm.walls = this.walls;

        shop[0] = new Shop(1, 420, 60); // RAWSON
        shop[1] = new Shop(6, 180, 180); // ムーンバックス
        shop[2] = new Shop(4, 420, 420); // ガヌト

        for(int i= 0; i < 3; i++){
            shops.add(shop[i]);
        }

        gm.shops = this.shops;
    }
```

```
    // 描画
    public void draw(Graphics g){
       for(Wall w:walls) w.draw(g);
       for(Shop s:shops) s. draw(g);
    }

    // 次のゲームパネル
    public Maze next(int n){
       switch(n){
           case 0 : return new Maze_4(gm);
           case 1 : // なし
           case 2 : return new Maze_8(gm);
           case 3 : // なし
           default : return this;
       }
    }
}
```

●Maze_8

```
/***
 *** Maze8
 ***/
import java.awt.*;
import java.util.ArrayList;

public class Maze_8 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[19];
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;
    private Shop shop[] = new Shop[2];
    private ArrayList<Shop> shops = new ArrayList<Shop>();

    // 生成
```

```
public Maze_8(GameManager gm){
    super(gm);
    this.gm = gm;
    wall[0] = new Wall(0, 240, a, 240);
    wall[1] = new Wall(0, 540, a, 120);
    wall[2] = new Wall(300, 600, 600, a);
    wall[3] = new Wall(600, 240, a, 480);
    wall[4] = new Wall(180, 0, 360, a);
    wall[5] = new Wall(540, 0, 120, a);

    wall[6] = new Wall(180, 120, 120, a);
    wall[7] = new Wall(420, 120, 120, a);
    wall[8] = new Wall(60, 240, 120, a);
    wall[9] = new Wall(480, 240, 240, a);
    wall[10] = new Wall(240, 360, 240, a);
    wall[11] = new Wall(180, 480, 120, a);

    wall[12] = new Wall(120, 60, a, 120);
    wall[13] = new Wall(480, 60, a, 120);
    wall[14] = new Wall(240, 180, a, 120);
    wall[15] = new Wall(120, 300, a, 120);
    wall[16] = new Wall(360, 420, a, 120);
    wall[17] = new Wall(480, 480, a, 240);
    wall[18] = new Wall(120, 540, a, 120);

    for(int i= 0; i < 19; i++){
        walls.add(wall[i]);
    }

    gm.walls = this.walls;

    shop[0] = new Shop(3, 540, 60); // モクドナルド
    shop[1] = new Shop(5, 180, 540); // 吉田家

    for(int i= 0; i < 2; i++){
        shops.add(shop[i]);
```

```
        }

        gm.shops = this.shops;
    }

     // 描画
     public void draw(Graphics g){
        for(Wall w:walls) w.draw(g);
        for(Shop s:shops) s. draw(g);
    }

     // 次のゲームパネル
    public Maze next(int n){
        switch(n){
            case 0 : return new Maze_5(gm);
            case 1 : // なし
            case 2 : return new Maze_9(gm);
            case 3 : return new Maze_7(gm);
            default : return this;
        }
    }
}
```

●Maze_9

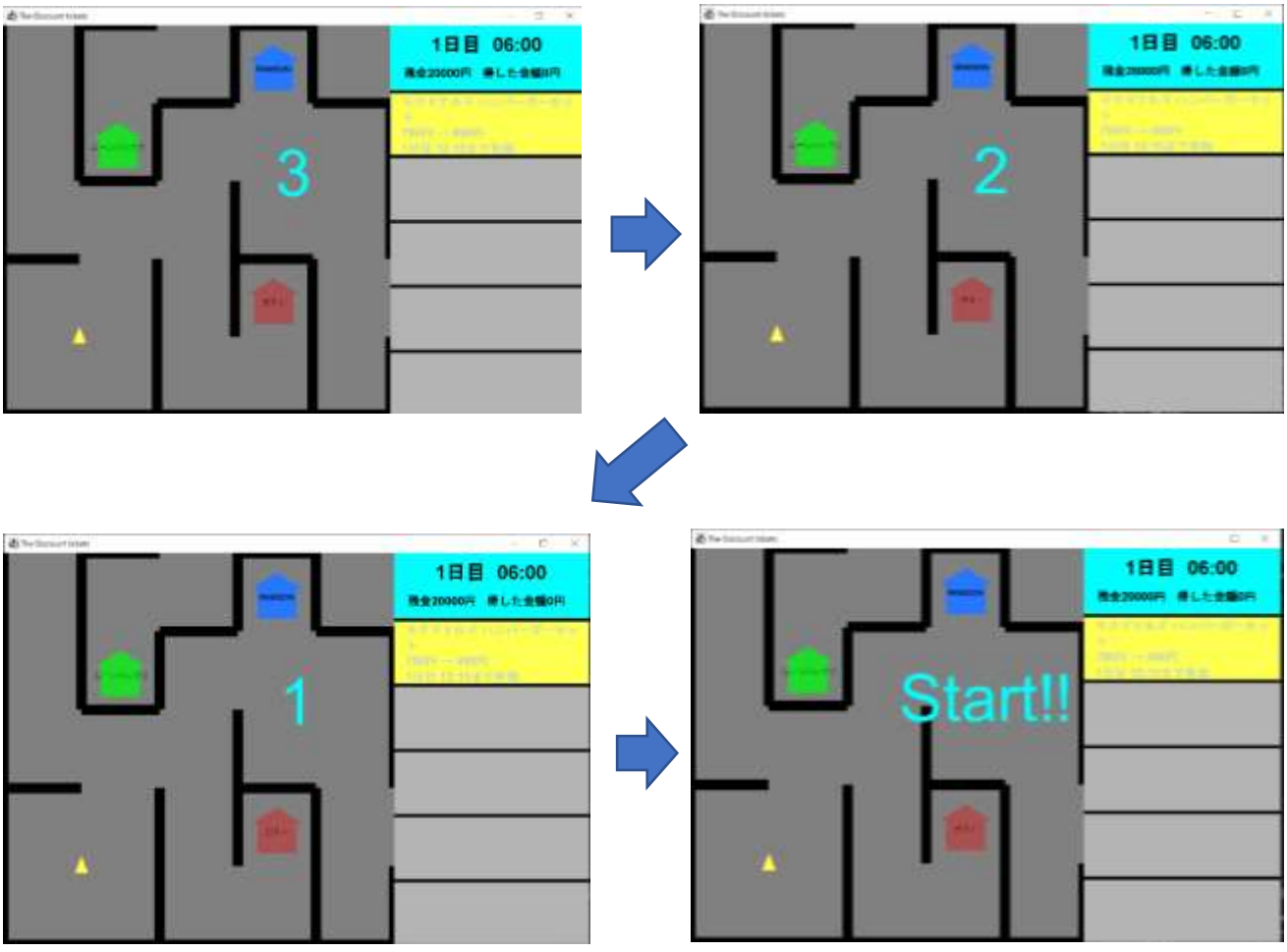
```
/***
 *** Maze9
 ***/
import java.awt.*;
import java.util.ArrayList;

public class Maze_9 extends Maze{
    private GameManager gm;
    private Wall wall[] = new Wall[16];
    private ArrayList<Wall> walls = new ArrayList<Wall>();
    private int a = 16;
```

```
    private Shop shop[] = new Shop[2];
    private ArrayList<Shop> shops = new ArrayList<Shop>();

    // 生成
    public Maze_9(GameManager gm){
        super(gm);
        this.gm = gm;
        wall[0] = new Wall(0, 240, a, 480);
        wall[1] = new Wall(300, 600, 600, a);
        wall[2] = new Wall(600, 300, a, 600);
        wall[3] = new Wall(420, 0, 360, a);
        wall[4] = new Wall(60, 0, 120, a);

        wall[5] = new Wall(180, 240, 120, a);
        wall[6] = new Wall(420, 240, 120, a);
        wall[7] = new Wall(240, 360, 240, a);
        wall[8] = new Wall(60, 480, 120, a);
        wall[9] = new Wall(420, 480, 120, a);

        wall[10] = new Wall(240, 60, a, 120);
        wall[11] = new Wall(360, 120, a, 240);
        wall[12] = new Wall(120, 180, a, 120);
        wall[13] = new Wall(480, 300, a, 120);
        wall[14] = new Wall(240, 420, a, 360);
        wall[15] = new Wall(360, 540, a, 120);


        for(int i= 0; i < 16; i++){
            walls.add(wall[i]);
        }

        gm.walls = this.walls;

        shop[0] = new Shop(8, 480, 120); // マツモトキヨコ
        shop[1] = new Shop(1, 180, 300); // RAWSON
```

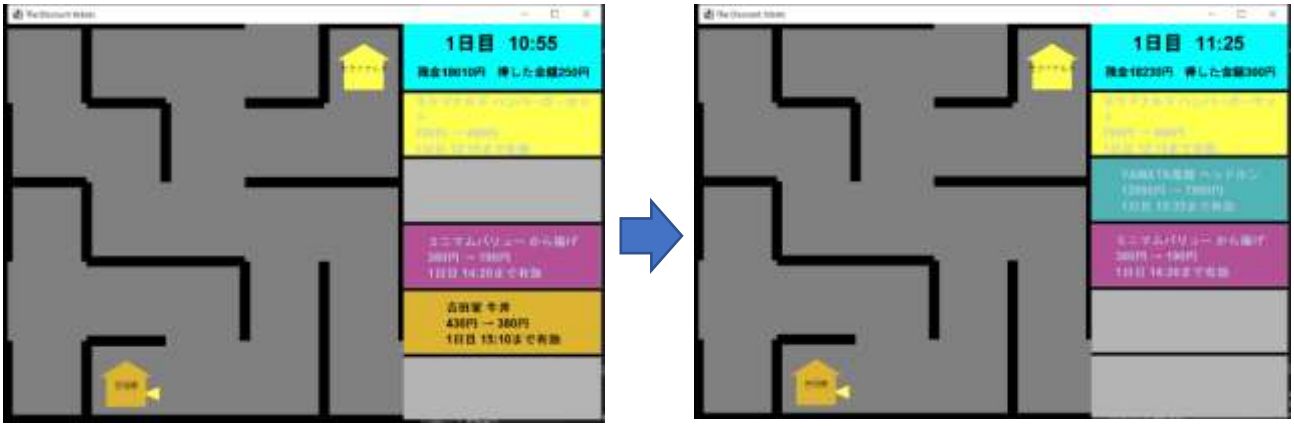
```
        for(int i= 0; i < 2; i++){
            shops.add(shop[i]);
        }

        gm.shops = this.shops;
    }

     // 描画
     public void draw(Graphics g){
        for(Wall w:walls) w.draw(g);
        for(Shop s:shops) s. draw(g);
    }

     // 次のゲームパネル
    public Maze next(int n){
        switch(n){
            case 0 : return new Maze_6(gm);
            case 1 : // なし
            case 2 : // なし
            case 3 : return new Maze_8(gm);
            default : return this;
        }
    }
}
```

●実行結果

ゲームウィンドウを開くとスタート画面が表示され, S キーを押すとカウントダウンが始まってゲームが開始した.

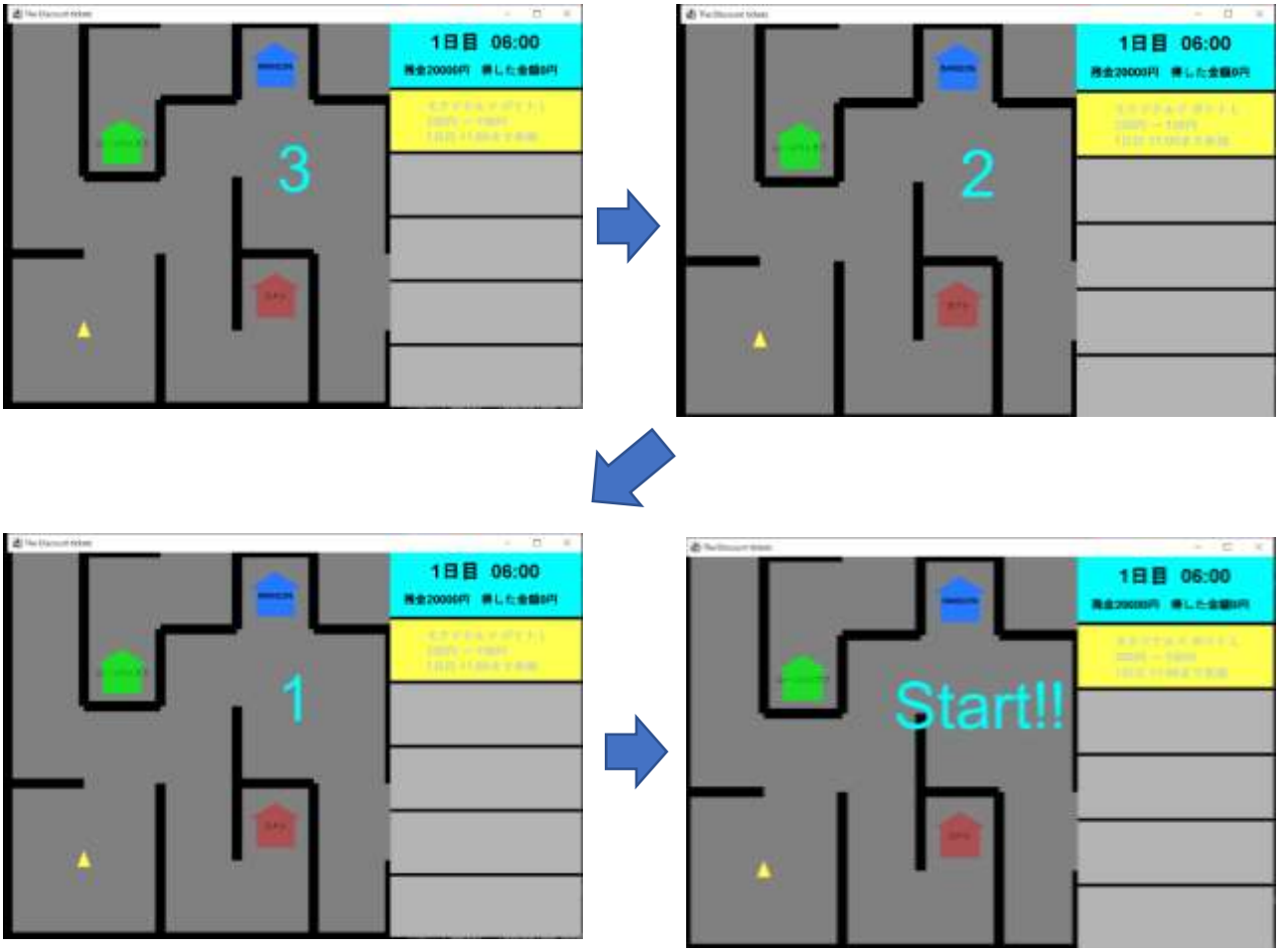


また，プレーヤーが店と接触すると，その店のクーポンの文字が黒色に変わった．この状態でクーポンをクリックするとそのクーポンは消え，支払った額だけ残金が減り，得した金額が増加した．

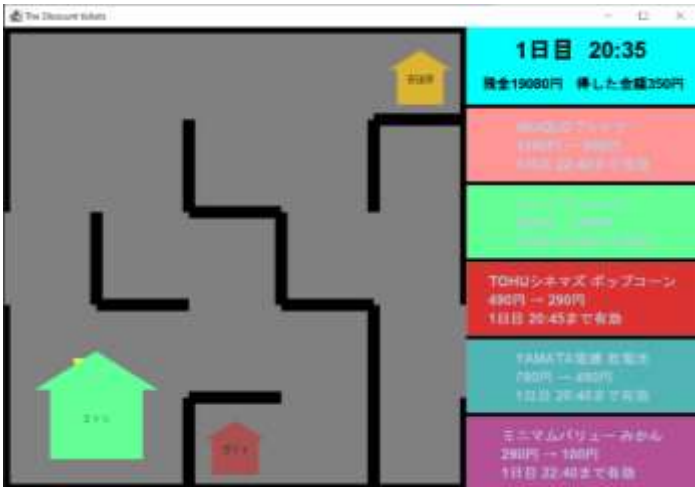


さらに該当するクーポンが複数ある場合は，その全てのクーポンが使用可能になった．また，有効期限が過ぎたクーポンは自動的に削除された．

4 日目の 6:00 になるとスタート画面に戻り，ゲーム名の代わりにスコアが表示された．S キーを押すとカウントダウン始まり，ゲームが再スタートした．



また，営業時間外の店と接触しても，その店のクーポンは使用可能な状態にはならなかった．
（ミトリの営業時間は 11:00～20:00）



残金より高額な買い物をするとゲームは終了し，スタート画面に切り替わった．

～参考資料～

・【簡単理解】Java での乱数の使い方まとめ　エンジニアの入り口
<https://eng-entrance.com/java-math-random>
・BoxLayout クラス　Let's プログラミング
<https://www.javadrive.jp/tutorial/boxlayout/>
・インセットの値を取得する　Let's プログラミング
<https://www.javadrive.jp/tutorial/insets/index3.html>
・ボタンに表示される文字列のフォントを設定する　Let's プログラミング
<https://www.javadrive.jp/tutorial/jbutton/index4.html>
・ボタンの前景色/背景色の設定と背景の透明/非透明の切り替え　Let's プログラミング
<https://www.javadrive.jp/tutorial/jbutton/index2.html>
・ボタンのサイズをピクセル単位で設定する　Let's プログラミング
<https://www.javadrive.jp/tutorial/jbutton/index5.html>
・ボタンに表示される文字列の水平及び垂直位置を設定する　Let's プログラミング
<https://www.javadrive.jp/tutorial/jbutton/index7.html>
・ボタンの有効/無効を切り替える　Let's プログラミング
<https://www.javadrive.jp/tutorial/jbutton/index14.html>
・ActionEvent　Let's プログラミング
<https://www.javadrive.jp/tutorial/event/index2.html>
・Swing によるクライアントデザイン②　【サラリーマン応援サイト】Japan IT エンジニア
<http://japanengineers.seesaa.net/article/383406375.html>
・GUI 部品にフォーカスを奪われた場合の対処：Java Tips　～Java プログラミング編　@IT
<https://atmarkit.itmedia.co.jp/ait/articles/0501/19/news131.html>
・【Java】キー入力処理　のんぽぐ軟弱クリエイターの備忘録
<https://nompor.com/2017/12/10/post-1924/>
・中間コンテナ　Java 入門
<http://msugai.fc2web.com/java/Swing/intermediate.html>
・【Java】JFrame のサイズにはタイトルバーや枠線も含まれている！　ゆーじの Unity 開発日記　<https://unity-yuji.xyz/java-jframe-size-insets/>
・ウィンドウを表示する前にタイトルバーの高さを取得する方法　ゆーじの Unity 開発日記　<https://unity-yuji.xyz/java-get-title-bar-before-show-window/>
・Java で使えるフォント名　MLTLab のプログラミングブログ
<https://www.mltlab.com/wp/archives/67>