# **J2Kad07D「ポリモーフィズムの復習」**

ECCがゲームを作ることになった！その名も「ドラゴンファンタジー」。戦士と魔法使いがドラゴン退治に行くという画期的なストーリーだ。とりあえず戦士を表すFighterクラスと魔法使いを表すMageクラス、および動作チェックプログラム（J2Kad07Dクラス）を業者に依頼して作ってもらったが、今後様々なジョブを追加するととんでもないことになるシロモノだった！ポリモーフィズムを使って処理を簡略化せよ。

|  |
| --- |
| **RPGCharacter** |
|  |
| + attack()  + defend()  + useItem() |

|  |
| --- |
| **Fighter** |
|  |
| + attack()  + defend() |

|  |
| --- |
| **Mage** |
|  |
| + attack()  + defend() |

スーパークラスとして

RPGCharacterクラスを作成する。

**リスト1：業者のプログラム（J2Kad07Dクラス）**

public class J2Kad07D {

public static void main(String[] args) {

…

while(true) {

System.out.print("ジョブを選んでください（0：戦士、1：魔法使い、-1：終了）＞");

int job = in.nextInt();

if (job < 0) break;

switch(job) {

default:

case 0: // Fighter

Fighter f = new Fighter();

f.attack();

f.defend();

f.useItem();

break;

case 1: // Mage

Mage m = new Mage();

m.attack();

m.defend();

m.useItem();

break;

}

System.out.println();

}

}

}

恐ろしい処理

**課題完成時の画面**

ジョブを選んでください（0：戦士、1：魔法使い、-1：終了）＞**0**

武器で攻撃します！

盾で防御します！

何かのアイテムを使います！

ジョブを選んでください（0：戦士、1：魔法使い、-1：終了）＞**1**

攻撃の魔法を唱えます！

防御の魔法を唱えます！

何かのアイテムを使います！

ジョブを選んでください（0：戦士、1：魔法使い、-1：終了）＞**-1**

# **J2Kad07C「抽象メソッドと抽象クラス」※J2Kad07Dをコピーして作成 （入門編P.240「実体のない抽象メソッド」、P.237「インスタンスを作れないクラス」）**

J2Kad07Dで追加したRPGCharacterクラスのattackメソッドとdefendメソッドにはそもそもプログラムコードが必要ないことがわかった。

1. RPGCharacterクラスのattackメソッドとdefendメソッドを抽象メソッドにせよ。
2. 新たに僧侶（Priestクラス）を追加し、Priestクラスの動作チェックも追加せよ。動作チェックの手順（メソッドを呼び出す順序）は戦士・魔法使いと同じとする。

|  |
| --- |
| **Priest** |
|  |
| + attack()  + defend() |

|  |
| --- |
| ***RPGCharacter*** |
|  |
| *+ attack()*  *+ defend()*  + useItem() |

抽象クラスおよび抽象メソッドは

*斜体*で記述する

**Priestクラスの仕様（RPGCharacterクラスを継承する）**

|  |  |
| --- | --- |
| メソッド | 仕様 |
| void attack() | 「みんなを励まします！」と表示する。 |
| void defend() | 「神に祈ります！」と表示する。 |

**課題完成時の画面（戦士と魔法使いはJ2Kad07Dと同じ）**

ジョブを選んでください（0：戦士、1：魔法使い、2：僧侶、-1：終了）＞**2**

みんなを励まします！

神に祈ります！

何かのアイテムを使います！

# **J2Kad07B「フック」※J2Kad07Cをコピーして作成**

僧侶（Priestクラス）独自の行動として「治療する」（healメソッド）を追加することになった。また新たに盗賊（Thiefクラス）も追加することになった。

1. Priestにhealメソッドを追加し、動作チェックできるようにせよ。
2. 動作チェックの手順そのものをRPGCharacterで実装し（checkAllActionメソッド）、mainメソッドからはcheckAllActionメソッドのみを呼び出すように修正せよ。
3. Thiefを追加し、Thiefも選択できるようにせよ。なお、Thiefのみ新たに「盗む」（stealメソッド）が追加されるものとする。

**Priestクラスの追加メソッド**

|  |  |
| --- | --- |
| メンバ | 仕様 |
| void heal() | 「みんなの傷を治します！」と表示する。 |

**Thiefクラスの仕様（RPGCharacterクラスを継承する）**

|  |  |
| --- | --- |
| メンバ | 仕様 |
| void attack() | 「素早く背後にまわって不意打ちします！」と表示する。 |
| void defend() | 「素早く攻撃をかわします！」と表示する。 |
| void steal() | 「何かを盗みます！」と表示する。 |

|  |
| --- |
| ***RPGCharacter*** |
|  |
| + callAllMethod()  *+ attack()*  *+ defend()*  + useItem()  … |

|  |
| --- |
| **Priest** |
|  |
| + attack()  + defend()  + heal() |

|  |
| --- |
| **Thief** |
|  |
| + attack()  + defend()  + steal() |

attack、defend、useItemなどの呼び出しはcallAllMethodから行う。

PriestのhealやThiefのstealを呼び出すためには、RPGCharacter側に工夫が必要。

**課題完成時の画面（戦士と魔法使いはJ2Kad07Dと同じ）**

ジョブを選んでください（0：戦士、1：魔法使い、2：僧侶、3：盗賊、-1：終了）＞**2**

みんなを励まします！

神に祈ります！

何かのアイテムを使います！

みんなの傷を治します！

ジョブを選んでください（0：戦士、1：魔法使い、2：僧侶、3：盗賊、-1：終了）＞**3**

素早く背後にまわって不意打ちします！

素早く攻撃をかわします！

何かのアイテムを使います！

何かを盗みます！

# **J2Kad07A「ECCコーヒー再び！（Template Method）」**

世界に羽ばたくECCがカフェを経営することになった。名付けて「ECCコーヒー」。メニューはコーヒーと紅茶。ところがドリンク生成プログラムを業者に依頼して作ってもらったところ、とんでもないシロモノになってしまった！これでは新しいメニューの追加が面倒だ！

1. スーパークラス（HotDrink）を導入してmainメソッドを簡略化せよ。
2. CoffeeクラスとTeaクラスも簡略化せよ。
3. ココア（Cocoa）を追加せよ。

ココアの作り方

①お湯を沸かす

②ココアパウダーを入れる

③カップに注ぐ

紅茶の作り方

①お湯を沸かす

②ティーバッグを浸す

③カップに注ぐ

コーヒーの作り方

①お湯を沸かす

②コーヒーをドリップする

③カップに注ぐ

ヒント：

サブクラスのメソッドをスーパークラスへ集約するためにはメソッドの仕様（戻り値、名前、引数）を同じにする必要がある。

|  |
| --- |
| **HotDrink** |
|  |
| ??? |

|  |
| --- |
| **Coffee** |
|  |
| + execute()  + boilWater()  + dripCoffee()  + pourInCup() |

|  |
| --- |
| **Tea** |
|  |
| + execute()  + boilWater()  + steepTeabag()  + pourInCup() |

継承して処理を集約したい。

**課題完成時の画面**

（J2Kad07Sを参照）

# **J2Kad07S「ECCコーヒー完成！」※J2Kad07Aをコピーして作成**

新メニューとして「3：ゆず茶」（Yuzuクラス）と「4：ミルクティー」（MilkTeaクラス）を追加せよ。

ミルクティーの作り方

①お湯を沸かす

②ティーバッグを浸す

③カップに注ぐ

④ミルクを加える

ゆず茶の作り方

①お湯を沸かす

②ゆずジャムを入れる

③カップに注ぐ

④はちみつを加える

**課題完成時の画面（J2Kad07A＆J2Kad07S）※「3：ゆず茶」と「4：ミルクティー」はJ2Kad07S**

ECCコーヒーへようこそ！

門外不出のレシピで作るから、おいしいよ！！

ご注文は？（0：コーヒー、1；紅茶、2：ココア、3：ゆず茶、4：ミルクティー、-1：店を出る）＞**0**

お湯を沸かしました！

コーヒーをドリップしました！

カップに注ぎました！

お待たせしました！ごゆっくりどうぞ！

ご注文は？（0：コーヒー、1；紅茶、2：ココア、3：ゆず茶、4：ミルクティー、-1：店を出る）＞**1**

お湯を沸かしました！

ティーバッグを浸しました！

カップに注ぎました！

お待たせしました！ごゆっくりどうぞ！

ご注文は？（0：コーヒー、1；紅茶、2：ココア、3：ゆず茶、4：ミルクティー、-1：店を出る）＞**2**

お湯を沸かしました！

ココアパウダーを入れました！

カップに注ぎました！

お待たせしました！ごゆっくりどうぞ！

ご注文は？（0：コーヒー、1；紅茶、2：ココア、3：ゆず茶、4：ミルクティー、-1：店を出る）＞**3**

お湯を沸かしました！

ゆずジャムを入れました！

カップに注ぎました！

はちみつを加えました！

お待たせしました！ごゆっくりどうぞ！

ご注文は？（0：コーヒー、1；紅茶、2：ココア、3：ゆず茶、4：ミルクティー、-1：店を出る）＞**4**

お湯を沸かしました！

ティーバッグを浸しました！

カップに注ぎました！

ミルクを加えました！

お待たせしました！ごゆっくりどうぞ！

# **J2Kad07X「直線描画のアルゴリズム」※実行画面を別ウインドウにしてください**

画面に画像を描画するためのCanvasクラスが準備されている。Canvasクラスに直線描画（drawLineメソッド）を追加し、三角形の描画を行え。なお、drawLineメソッド内において変数を使う場合は、データ型はint型のみとする。

ヒント：「ブレゼンハムのアルゴリズム」「直線描画のアルゴリズム」などで検索すること。

**Canvasクラスの仕様（drawLineメソッドを追加する）**

|  |  |
| --- | --- |
| メンバ | 仕様 |
| boolean[][] pixel | 仮想画面に該当する2次元配列。値がtrueのとき「○」falseのとき「・」を表示する。 |
| Canvas(int width, int height) | コンストラクタ。高さheight×横幅widthでpixelを生成する（初期値はfalse）。 |
| void show() | 仮想画面pixelを画面に表示する。 |
| boolean inBound(int x, int y) | 座標(x, y)がpixelの範囲内ならtrue、範囲外ならfalseを返す。 |
| void set(int x, int y) | 座標(x, y)がpixelの範囲内なら点を打つ「○」。 |
| void reset(int x, int y) | 座標(x, y)がpixelの範囲内なら点を消す「・」。 |
| void drawLine(int x1, int y1,  int x2, int y2) | 座標(x1, y1)から座標(x2, y2)まで直線を引く（「○」を打つ）。  **※変数を使う場合はint型のみOK。** |

**mainメソッドの仕様（三角形の描画）**

1. drawLineメソッドを使って三角形を描画する。三角形の頂点は以下の通り。  
     
   (10, 17)、(30, 17)、(20, 2)

**課題完成時の画面（実装するアルゴリズムによっては1マス程度「○」がずれることもあり）**

　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・

　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・

　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　○　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・

　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　○　・　○　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・

　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　○　・　・　・　○　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・

　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　○　・　・　・　○　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・

　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　○　・　・　・　・　・　○　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・

　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　○　・　・　・　・　・　・　・　○　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・

　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　○　・　・　・　・　・　・　・　○　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・

　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　○　・　・　・　・　・　・　・　・　・　○　・　・　・　・　・　・　・　・　・　・　・　・　・　・

　・　・　・　・　・　・　・　・　・　・　・　・　・　・　○　・　・　・　・　・　・　・　・　・　・　・　○　・　・　・　・　・　・　・　・　・　・　・　・　・

　・　・　・　・　・　・　・　・　・　・　・　・　・　・　○　・　・　・　・　・　・　・　・　・　・　・　○　・　・　・　・　・　・　・　・　・　・　・　・　・

　・　・　・　・　・　・　・　・　・　・　・　・　・　○　・　・　・　・　・　・　・　・　・　・　・　・　・　○　・　・　・　・　・　・　・　・　・　・　・　・

　・　・　・　・　・　・　・　・　・　・　・　・　○　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　○　・　・　・　・　・　・　・　・　・　・　・

　・　・　・　・　・　・　・　・　・　・　・　・　○　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　○　・　・　・　・　・　・　・　・　・　・　・

　・　・　・　・　・　・　・　・　・　・　・　○　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　○　・　・　・　・　・　・　・　・　・　・

　・　・　・　・　・　・　・　・　・　・　○　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　○　・　・　・　・　・　・　・　・　・

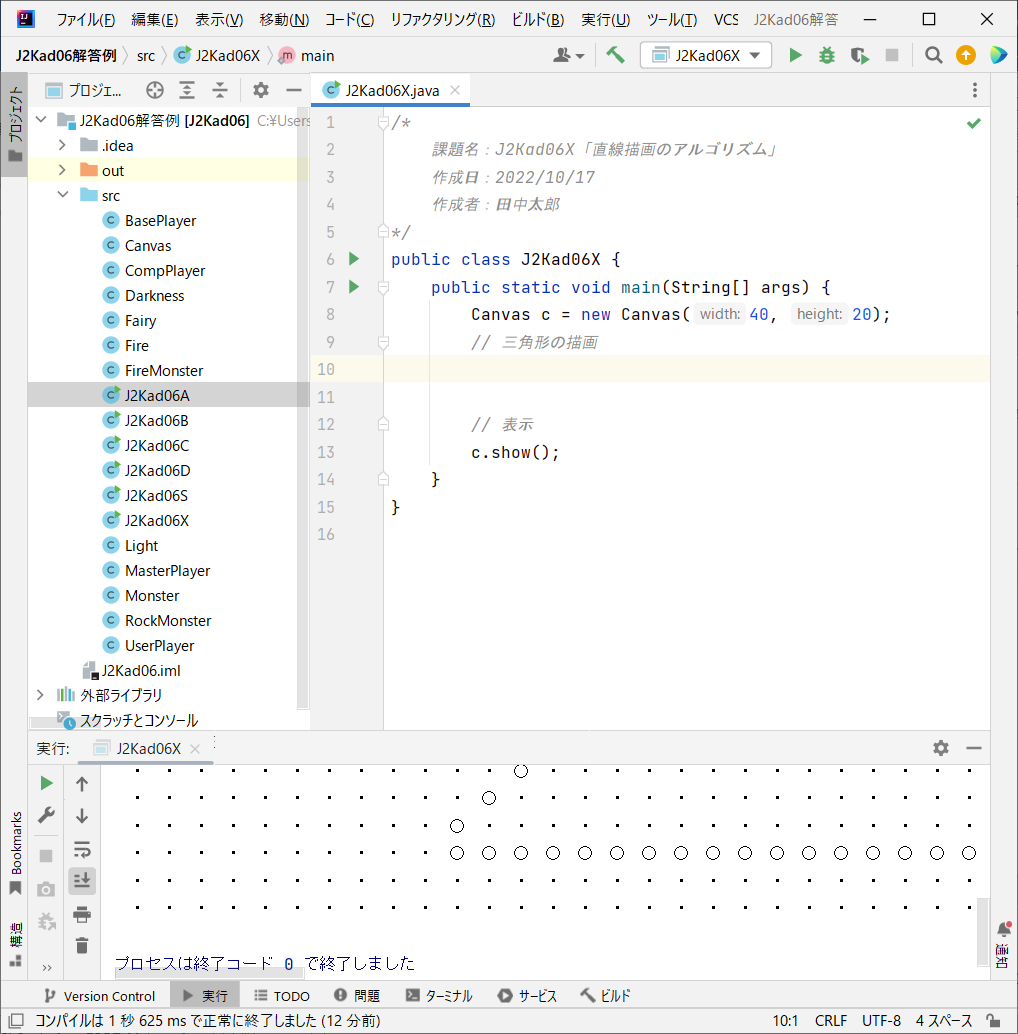
　・　・　・　・　・　・　・　・　・　・　○　○　○　○　○　○　○　○　○　○　○　○　○　○　○　○　○　○　○　○　○　・　・　・　・　・　・　・　・　・

　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・

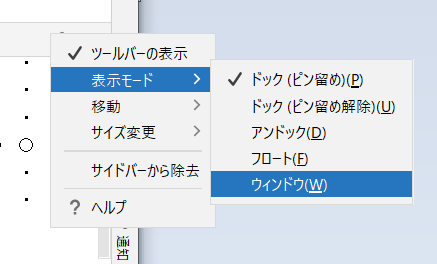
　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・　・

# **実行画面を別ウインドウにする方法**

1. プログラムを実行する。
2. 実行画面右上の⚙をクリックする。



1. ポップアップメニューが表示されるので、[表示モード]→[ウインドウ]を選択する。



1. 実行画面が別ウインドウになるので好みの大きさ（例えば全画面表示）に設定する。