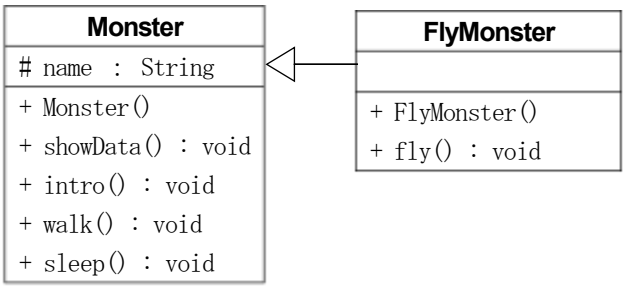
● J2Kad05D「クラスの継承」

（入門編 P.193「継承の概念」、P.194「継承の親子関係」、P.195「継承を行うための extends」、P.197「フィールドとメソッドの継承」）

Monster クラス（ピカチュウとムックル）を使った処理が作成されている。ムックルを FlyMonster クラスに変更し fly メソッドの呼び出しを追加せよ。

```
Monster muku = new Monster();
     ↓
FlyMonster muku = new FlyMonster();
```



FlyMonster クラスの仕様（Monster クラスを継承する）

メソッド	仕様
コンストラクタ	「FlyMonster クラスのコンストラクタが呼び出されました！」と表示する。
void fly()	「～が飛ぶよ！びゅ～ん!!」（～は名前）と表示する。

課題作成前の画面（Before）

```
Monster クラスのコンストラクタが呼び出されました！
おいらの名前はピカチュウ。
趣味は散歩。特技はどこでも眠れることだよ。
てくてく・・・
ぐうぐう・・・

Monster クラスのコンストラクタが呼び出されました！
おいらの名前はムックル。
趣味は散歩。特技はどこでも眠れることだよ。
てくてく・・・
ぐうぐう・・・
```

課題完成時の画面（After）

```
Monster クラスのコンストラクタが呼び出されました！
おいらの名前はピカチュウ。
趣味は散歩。特技はどこでも眠れることだよ。
てくてく・・・
ぐうぐう・・・

Monster クラスのコンストラクタが呼び出されました！
FlyMonster のコンストラクタが呼び出されました！
おいらの名前はムックル。
趣味は散歩。特技はどこでも眠れることだよ。
てくてく・・・
ぐうぐう・・・
ムックルが飛ぶよ！びゅ～ん！
```

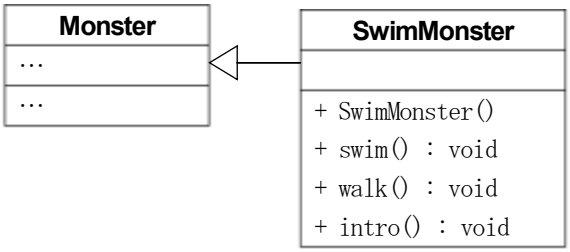
コンストラクタの呼び出し順と fly メソッドの呼び出しを確認すること。

● J2Kad05C「オーバーライド」

（入門編 P.199「メソッドのオーバーライド」、P.202「super でスーパークラスのメソッドを呼び出す」）

Monster クラス（コイキング）を使った処理が作成されている。コイキングを SwimMonster クラスに変更し swim メソッドの呼び出しを追加せよ。

```
Monster king = new Monster();
     ↓
SwimMonster king = new SwimMonster();
```



SwimMonster クラスの仕様（新規作成、Monster クラスを継承する）

メソッド	仕様
コンストラクタ	「SwimMonster クラスのコンストラクタが呼び出されました！」と表示する。
void swim()	「～が泳ぐよ！ぶくぶく･･･」（～は名前）と表示する。
void walk()	「尾びれだと歩けないよ～」と表示する。
void intro()	Monster クラスの intro メソッドを呼び出した後、「泳ぎも得意さ！」と表示する。

課題完成時の画面

```
Monster クラスのコンストラクタが呼び出されました！
SwimMonster のコンストラクタが呼び出されました！
おいらの名前はコイキング。
趣味は散歩。特技はどこでも眠れることだよ。
泳ぎも得意さ！
尾びれだと歩けないよ～
ぐうぐう・・・
コイキングが泳ぐよ！ぶくぶく・・・っ！？
```

← Monster クラスの intro メソッドの呼び出し

← オーバーライド

● J2Kad05B「月面着陸ゲーム！（新型ロケット）」

（入門編 P.205「デフォルトコンストラクタ」、P.206「サブクラスのコンストラクタの動作」、P.207「スーパークラスのコンストラクタの呼び出し」）

「宇宙を~~われらのもの~~平和に！」を合言葉に ECC 航空宇宙局では月面探査に挑戦している。しかし何度挑戦しても失敗ばかり（JChallenge18A「月面着陸ゲーム！」）。そこで今度は新型ロケットを投入することにした。最新式のエンジンを積んでいる。この新型ロケットを使って月面着陸を成功させ、まずは月を~~われらのもの~~平和にせよ。

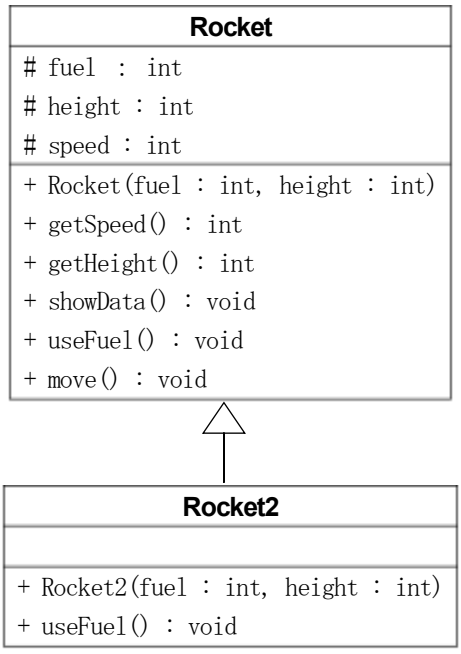
1. JChallenge18A の完成版相当のプログラムが準備されている。ゲーム部分は J2Kad05B クラス、ロケットは Rocket クラスとして作成されている。これを実行し月面着陸がいかに難しいかを確認せよ。
2. Rocket クラスを継承して Rocket2 クラス（新型ロケット）を作成し、Rocket2 クラスで月面着陸に挑戦せよ。

Rocket2 クラスの仕様（新規作成、Rocket クラスを継承）

メソッド	仕様
コンストラクタ	Rocket クラスのコンストラクタを呼び出し、「新型エンジンだ！」と表示する。
void useFuel()	Rocket クラスの useFuel メソッドと同じ。ただし逆噴射による減速は「7」とする。

課題完成時の画面

```
ロケットを作った！
新型エンジンだ！
--------------------------------
[燃料]：15　[落下速度]：0　　[高度]：300
--------------------------------
逆噴射しますか？（1：する、それ以外：しない）＞0
　　⋮
　(中略)
　　⋮
--------------------------------
[燃料]：7　　[落下速度]：12　[高度]：10
--------------------------------
逆噴射しますか？（1：する、それ以外：しない）＞1
燃料を使った！

--------------------------------
[燃料]：6　　[落下速度]：9　　[高度]：1
--------------------------------
逆噴射しますか？（1：する、それ以外：しない）＞1
燃料を使った！

--------------------------------
[燃料]：5　　[落下速度]：6　　[高度]：-5
--------------------------------
おめでとう！着陸成功！！
```



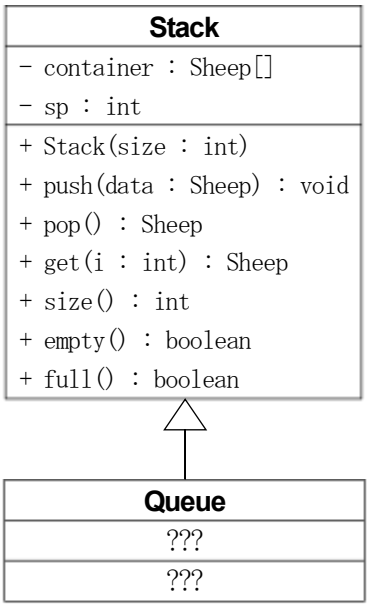
Rocket で逆噴射しても落下速度は 1 しか減らないが
最新式エンジンの Rocket2 だと 3 減る。

● J2Kad05A「レジ待ち行列③」

ECC が激安スーパーの 3 号店を開業した！今度の客は Monster ではなく Sheep だ。ところがレジ待ち行列をまたもやスタック形式にしてしまった！！ これで 3 回目だ。しかしすでに 1 号店にてキュー形式を導入済みだ（J2Kad04X1「レジ待ち行列①」）。3 号店もキュー形式に変更せよ。その際、キュー（Queue クラス）はスタック（Stack クラス）を継承するようにせよ。

課題完成時の画面

```
      ⋮
   (中略)
      ⋮
現在のレジ待ち行列です！
0：ユークリッド
1：フィボナッチ
2：レントゲン
3：パスカル

何をしますか？（0：客を呼び込む、1：レジを打つ、-1：店をたたむ）＞0
ガウスがやってきた！

現在のレジ待ち行列です！
0：ユークリッド
1：フィボナッチ
2：レントゲン
3：パスカル
4：ガウス

何をしますか？（0：客を呼び込む、1：レジを打つ、-1：店をたたむ）＞1
ユークリッドは帰っていった！！

現在のレジ待ち行列です！
0：フィボナッチ
1：レントゲン
2：パスカル
3：ガウス

何をしますか？（0：客を呼び込む、1：レジを打つ、-1：店をたたむ）＞
```



Queueにすると先頭のユークリッドから処理される。

● J2Kad05S「ECC コーヒー！」

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。そこでドリンク生成処理（J2Kad05S）を作成するためコーヒーと紅茶の作り方をクラス（Coffee と Tea）にしたところ、いくつか共通部分が見つかった。今後さらにメニューを追加することを考えるとこのままでは効率が悪い。

1. スーパークラス（HotDrink）を導入して処理の無駄を省け（共通部分を HotDrink へ持って行く）。
2. ココア（Cocoa）とミルクティー（MilkTea）を追加せよ。

各ドリンクの作り方

コーヒーの作り方	紅茶の作り方	ココアの作り方	ミルクティーの作り方
①お湯を沸かす	①お湯を沸かす	①お湯を沸かす	①お湯を沸かす
②コーヒーをドリップする	②ティーバッグを浸す	②ココアパウダーを加える	②ティーバッグを浸す
③カップに注ぐ	③カップに注ぐ	③カップに注ぐ	③カップに注ぐ
			④ミルクを加える

Coffee クラスと Tea クラス

Coffee
+ boilWater() + dripCoffee() + pourInCup()

Tea
+ boilWater() + steepTeabag() + pourInCup()

課題完成時の画面

```
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：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：ミルクティー、-1：店を出る）>3
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！

ご注文は？（0：コーヒー、1；紅茶、2：ココア、3：ミルクティー、-1：店を出る）>-1
ありがとうございました！
```

● J2Kad05X「ECC 苦情処理センター②」

ECC 苦情処理センターでは受け付けた苦情を 0 から 99 までの番号に分類して処理している。ここで働くのは優秀なスタッフばかりだ。受け付けた苦情を対応順に処理していく。各スタッフの処理をクラスにして作成したところ、苦情対応できるかどうか判断する箇所以外はほぼ同じ処理だった。スーパークラス（Handler クラス）を導入して各スタッフクラスを簡略化せよ。

課題完成時の画面

```
ここはECC 苦情処理センターです！
優秀なスタッフが対応します！

のび太がスタンバイした！
ジャイアンがスタンバイした！
スネ夫がスタンバイした！
出木杉がスタンバイした！

苦情番号：58 を受け付けた！
どうしますか？（1：対応する、それ以外：もうやだ）＞1
のび太：専門外です・・・
ジャイアン：専門外です・・・
スネ夫：専門外です・・・
出木杉：私が対応します！

苦情番号：10 を受け付けた！
どうしますか？（1：対応する、それ以外：もうやだ）＞1
のび太：私が対応します！

苦情番号：78 を受け付けた！
どうしますか？（1：対応する、それ以外：もうやだ）＞1
のび太：専門外です・・・
ジャイアン：専門外です・・・
スネ夫：私が対応します！

苦情番号：65 を受け付けた！
どうしますか？（1：対応する、それ以外：もうやだ）＞1
のび太：専門外です・・・
ジャイアン：私が対応します！

苦情番号：82 を受け付けた！
どうしますか？（1：対応する、それ以外：もうやだ）＞0
おつかれさまでした！
```

スタッフ	対応できる番号
のび太	20 未満のとき処理 OK
ジャイアン	5 の倍数のとき処理 OK
スネ夫	3 の倍数のとき処理 OK
出木杉	どんな番号でも処理 OK