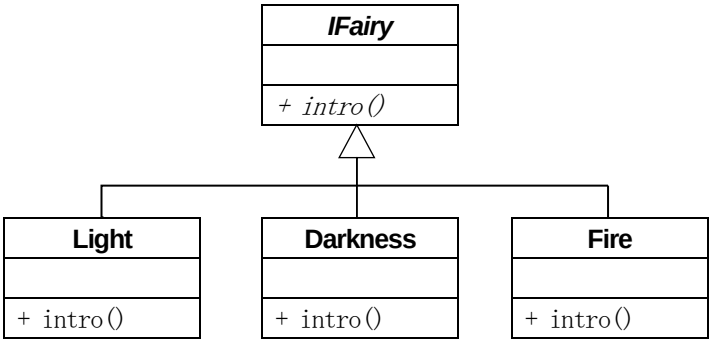
● J2Kad08D「妖精の召喚（抽象クラス版）」

J2Kad08D に妖精を召喚して自己紹介させる処理が作成されている。妖精は光の妖精（Light）、闇の妖精（Darkness）、炎の妖精（Fire）の 3 種類だ。抽象クラス（IFairy）を追加して、自己紹介処理を簡略化せよ。

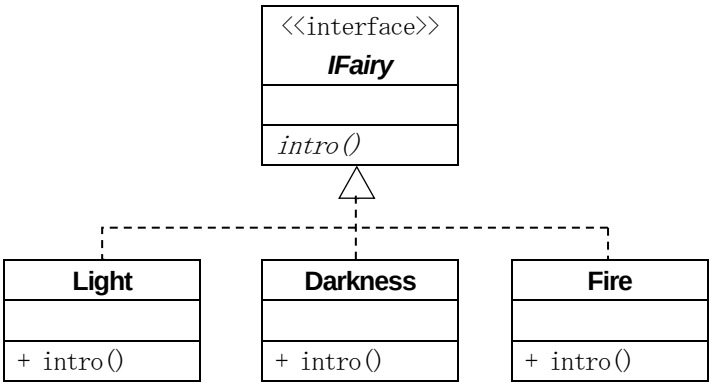


課題完成時の画面

```
妖精を召喚して自己紹介させます！
誰を召喚しますか？（0：光の妖精、1：闇の妖精、2：炎の妖精、-1：やめる）＞0
わたしは光の妖精！この者に祝福を！！

誰を召喚しますか？（0：光の妖精、1：闇の妖精、2：炎の妖精、-1：やめる）＞1
わたしは闇の妖精だ！闇の力を思い知れ！！

誰を召喚しますか？（0：光の妖精、1：闇の妖精、2：炎の妖精、-1：やめる）＞2
わたしは炎の妖精さ！炎の力は気まぐれなのさ！！

誰を召喚しますか？（0：光の妖精、1：闇の妖精、2：炎の妖精、-1：やめる）＞-1
```

● J2Kad08C「妖精の召喚（インターフェイス版）」

J2Kad08C に J2Kad08D と同等の処理が準備されている。IFairy クラスをインターフェイスに変更せよ。



課題完成時の画面

（J2Kad08D と同じ）

● J2Kad08B「激安スーパーECC3 号店！」

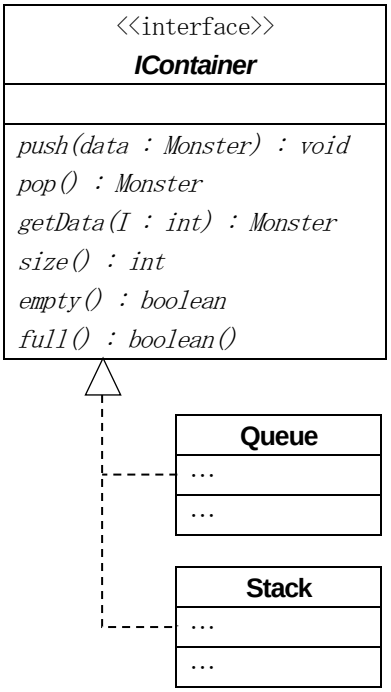
激安スーパーECC が 3 号店を開店した！現在、レジ待ち行列は「待ち行列（キュー）形式」（Queue クラス）だが、3 号店の店長は「キュー形式」と「スタック形式」（Stack クラス）を選択できるようにしたいらしい。Queue クラスと Stack クラスからインターフェイスを抽出し（IContainer クラス）、レジ待ち行列の形式を選択できるようにせよ。

Queue クラスの仕様（作成済み）

メンバ	仕様
Queue(int size)	データ数 size のキューを作る
void push(Monstaer data)	キューへデータを格納する
Monster pop()	キューからデータを取り出す
Monster getData(int i)	i 番目のモンスターを返す
int size()	並んでいるモンスター数を返す
boolean empty()	誰も並んでいなければ true を返す
boolean full()	キューがいっぱいのとき true を返す

Stack クラスの仕様（作成済み）

メンバ	仕様
Stack(int size)	データ数 size のスタックを作る
void push(Monstaer data)	スタックへデータを格納する
Monster pop()	スタックからデータを取り出す
Monster getData(int i)	i 番目のモンスターを返す
int size()	並んでいるモンスター数を返す
boolean empty()	誰も並んでいなければ true を返す
boolean full()	スタックがいっぱいのとき true を返す



課題完成時の画面（スタックを選択した場合）

```
いらっしゃい！激安スーパーECC3 号店です！！
レジ待ち行列をスタック形式と待ち行列形式から選べます！
どちらにしますか？（0：スタック、それ以外：待ち行列）＞0
スタックを作りました！
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
ウインディがやってきた！
ミュウがやってきた！
スピアーがやってきた！

現在のレジ待ち行列です！
0：ウインディ
1：ミュウ
2：スピアー

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
スピアーは帰っていった！！
```

スタックを選ぶとあとから並んだモンスターから先にレジを打つ。
待ち行列（キュー）を選択すると先に並んだモンスターから順にレジを打つ。

● J2Kad08A「図形を描こう！」

図形の描画と消去を行う処理が準備されている。Triangle クラスで二等辺三角形、Rectangle クラスで長方形、Circle クラスで円の描画ができる。また Eraser クラスで図形の消去ができる。

1．インターフェイス IShape を定義して、図形描画処理を簡略化せよ。必要であれば各描画クラス（Triangle、Rectangle、Circle、Eraser）を修正してもよい。
2．正方形を描画する Square クラスを追加せよ。

Triangle
・・・
Triangle(・・・) drawTriangle(c ： Canvas)

Rectangle
・・・
Rectangle(・・・) drawRectangle(c ： Canvas)

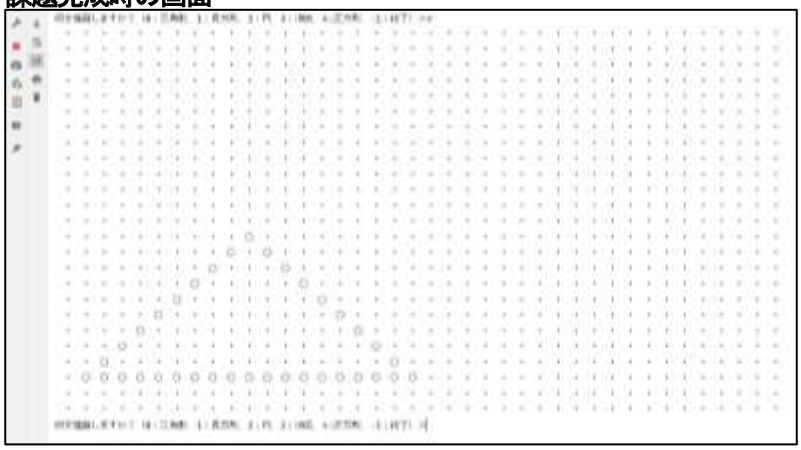
Circle
・・・
Circle(・・・) drawCircle(c ： Canvas)

Eraser
・・・
Eraser(・・・) erase(c ： Canvas)

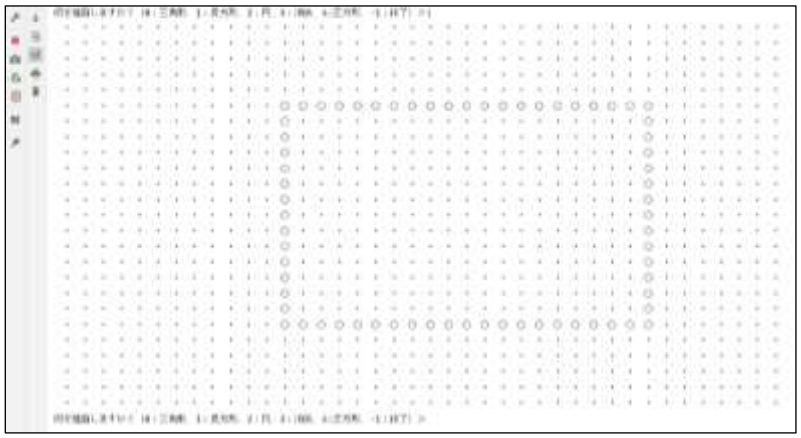
入力	描画する図形	使用するクラス
0	二等辺三角形を描画する。左下座標：(1, 22)、底辺の長さ：18、高さ：9	Triangle
1	長方形を描画する。左上座標：(12, 5)、横幅：20、高さ：14	Rectangle
2	円を描画する。中心座標：(29, 9)、半径：8	Circle
3	図形を消去する	Eraser
4	正方形を描画する。左上座標：(1, 1)、辺の長さ：16	Square

ヒント1：ポリモーフィズムを使うためにはメソッドの仕様を同じにする必要がある。
ヒント2：正方形（Square）は長方形（Rectangle）を継承すれば作成できる。

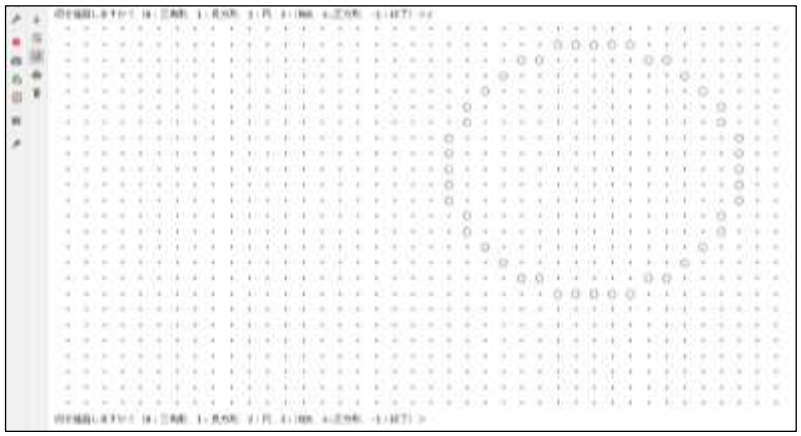
課題完成時の画面



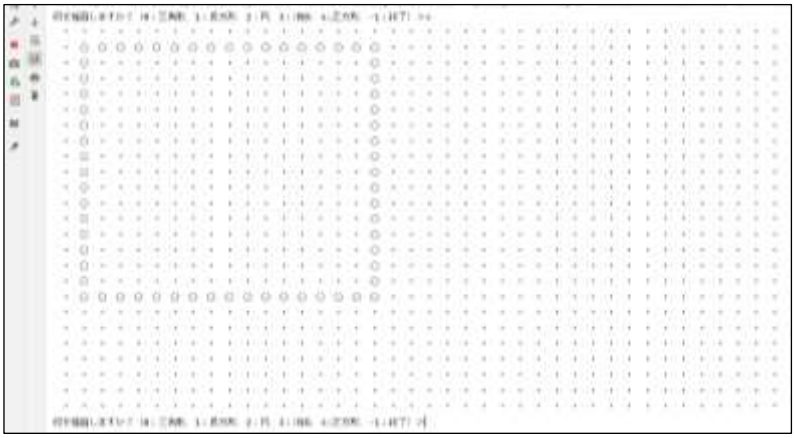
三角形を描画



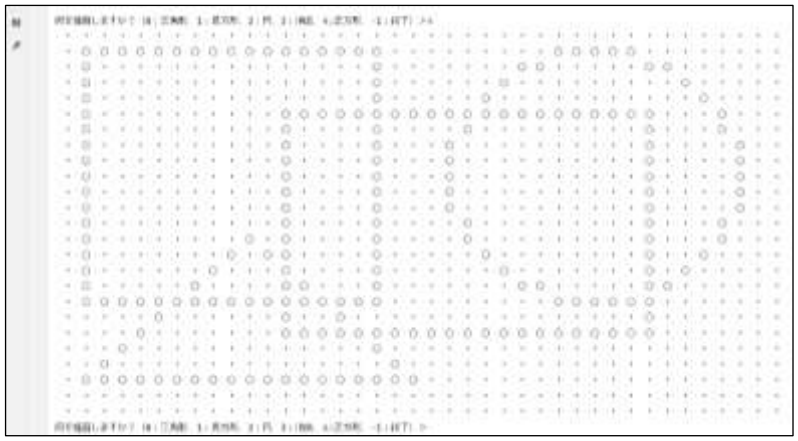
長方形を描画



円を描画



正方形を描画



すべて描画

● J2Kad08S「新型エアコンの開発」

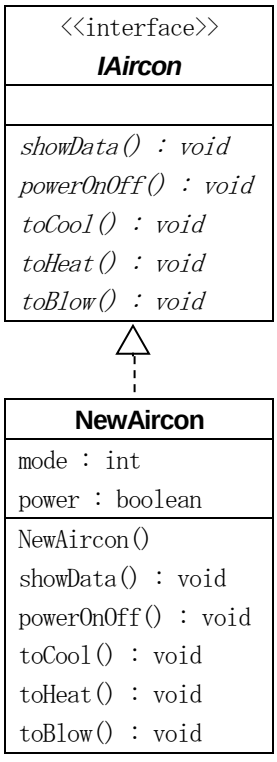
世界的な家電メーカーECC ホームエレクトロニクスがエアコン操作の画期的なインターフェイス IAircon を発表した！ボタンを押すだけで「電源 ON/OFF」「冷房」「暖房」「送風」を切り換えられるというスグレモノだ！このインターフェイスで操作するエアコンを開発し（NewAricon クラス）、IAircon を使って動作確認を行え。

NewAricon クラスでの実装（IAircon に基づいて実装する）

メンバ	仕様
int mode	運転モード（0：冷房、1：暖房、2：送風）、初期値：冷房
boolean power	電源（true：ON、false：OFF）、初期値：OFF
コンストラクタ	mode と power に初期値を設定する
void showData()	電源 ON のとき「電源：ON、運転モード～」（～は現在のモード）と表示する。 電源 OFF のとき「電源：OFF」と表示する。
void powerOnOff()	電源 ON/OFF を切り換えて「電源を入れました！」または「電源を切りました！」と表示する。
void toCool()	電源 OFF のとき「電源が入っていません！」と表示する。 電源 ON のとき運転モードを冷房にして「冷房に切り換えました！」と表示する。
void toHeat()	電源 OFF のとき「電源が入っていません！」と表示する。 電源 ON のとき運転モードを暖房にして「暖房に切り換えました！」と表示する。
void toBlow()	電源 OFF のとき「電源が入っていません！」と表示する。 電源 ON のとき運転モードを送風にして「送風に切り換えました！」と表示する。

課題完成時の画面

```
ECC ホームエレクトロニクスが新型エアコンを開発した！
電源：OFF
どうしますか？（0：電源 ON/OFF、1：冷房、2：暖房、3：送風、-1：終了）＞0
電源を入れました！

電源：ON、モード：冷房
どうしますか？（0：電源 ON/OFF、1：冷房、2：暖房、3：送風、-1：終了）＞1
冷房に切り換えました！

電源：ON、モード：冷房
どうしますか？（0：電源 ON/OFF、1：冷房、2：暖房、3：送風、-1：終了）＞2
暖房に切り換えました！

電源：ON、モード：暖房
どうしますか？（0：電源 ON/OFF、1：冷房、2：暖房、3：送風、-1：終了）＞3
送風に切り換えました！

電源：ON、モード：送風
どうしますか？（0：電源 ON/OFF、1：冷房、2：暖房、3：送風、-1：終了）＞0
電源を切りました！

電源：OFF
どうしますか？（0：電源 ON/OFF、1：冷房、2：暖房、3：送風、-1：終了）＞1
電源が入っていません！
```

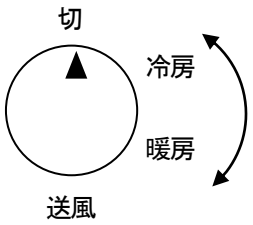


● J2Kad08X「旧式エアコン（Adapter パターン）」

ECC ホームエレクトロニクスには「つまみ」を回して操作する旧式エアコン（OldAircon）の在庫が山ほどあった！これを IAircon インターフェイスで操作できるようにして売りさばきたい！！OldAircon を IAircon で操作できるように処理を作成せよ。ただし OldAircon はすでに製品として出来上がってしまっているので OldAircon に手を加えることはできないものとする（もちろん IAircon も変更不可）。

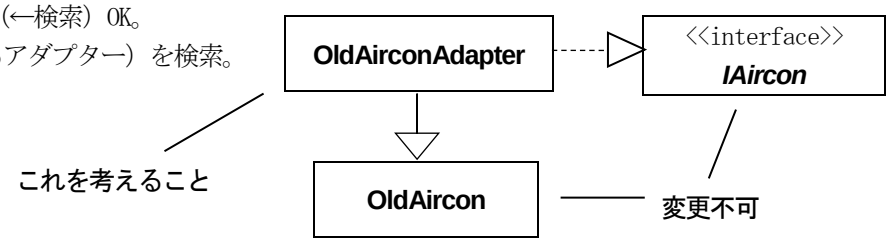
OldAricon クラスの仕様（変更不可）

メンバ	仕様
コンストラクタ	つまみを[切]（OFF）にする。
int getKnob()	現在のつまみ（ノブ）の位置を返す。 （[切]：OFF、[冷房]：COOL、[暖房]：HEAT、[送風]：BLOW）
void showData()	現在のつまみの位置（運転モード）を表示する。
void turnRight()	つまみを右（時計回り）へひとつ回す。
void turnLeft()	つまみを左（反時計回り）へひとつ回す。



「つまみ」を回して切り換える。
[切]⇔[冷房]⇔[暖房]⇔[送風]

ヒント 1：インターフェイスは多重継承（←検索）OK。
ヒント 2：Adapter パターン（継承によるアダプター）を検索。



課題完成時の画面

```
新型エアコンと同じ操作で旧式エアコンを動かします！
ただいま[切]です。
どうしますか？（0：電源 ON/OFF、1：冷房、2：暖房、3：送風、-1：終了）＞0
つまみを右に回した！

ただいま[冷房]です。
どうしますか？（0：電源 ON/OFF、1：冷房、2：暖房、3：送風、-1：終了）＞2
つまみを右に回した！

ただいま[暖房]です。
どうしますか？（0：電源 ON/OFF、1：冷房、2：暖房、3：送風、-1：終了）＞3
つまみを右に回した！

ただいま[送風]です。
どうしますか？（0：電源 ON/OFF、1：冷房、2：暖房、3：送風、-1：終了）＞0
つまみを左に回した！
つまみを左に回した！
つまみを左に回した！

ただいま[切]です。
どうしますか？（0：電源 ON/OFF、1：冷房、2：暖房、3：送風、-1：終了）＞-1
```

つまみが[切]のときに
電源 ON/OFF をすると
[冷房]で動くものとする。