# **J2Kad24D「世界にはばたくECCフーズ！（ループの復習）」**

世界にはばたくECCフーズは外食チェーン店を次々と買収している。現在傘下にあるのは、ECCコーヒーとECCドーナツ、さらに他のチェーン店の買収も計画している。そこで全チェーン店のメニューを表示する処理を作成したい。まずはfor文を使ってECCコーヒーとECCドーナツのすべてのメニューを表示する処理を作成せよ。ただしECCコーヒー・ECCドーナツともに元は別々の会社だったため、メニュー管理方式が異なるので注意すること。

**リスト1：MenuItemクラス（ファイル「MenuItem.java」）**

メニュー名と値段を格納するクラス。printメソッドで表示する。

public class MenuItem {

private String name; // メニュー名

private int price; // 値段

public MenuItem(String name, int price) {

this.name = name;

this.price = price;

}

public void print() { System.out.println(name + "：" + price + "円"); }

}

**リスト2：ECCコーヒーのメニュー（ファイル「CafeMenu.java」）**

MenuItemの配列で管理している。配列の値がnullのとき、それ以上メニューはない。

public class CafeMenu {

private MenuItem[] menu = new MenuItem[100];

public CafeMenu() {

menu[0] = new MenuItem("ドリップコーヒー", 390);

menu[1] = new MenuItem("アールグレイ", 430);

menu[2] = new MenuItem("オレンジジュース", 220);

menu[3] = null; // 終了コード

}

public MenuItem[] getMenu() { return menu; } // menu配列のゲッター

}

**リスト3：ECCドーナツのメニュー（ファイル「DonutMenu.java」）**

メニュー名と値段をそれぞれ別々の配列で管理している。値段の値が-1のとき、それ以上メニューはない。

なぜ、こんな方法で管理しているのか知る由もない（買収したときすでにこうなっていた）。

public class DonutMenu {

private String[] names = new String[100]; // メニュー名

private int[] prices = new int[100]; // 値段

public DonutMenu() {

names[0] = "ハニーディップ"; prices[0] = 120;

names[1] = "ハニーチュロ"; prices[1] = 130;

names[2] = "チョコリング"; prices[2] = 140;

prices[3] = -1; // 終了コード

}

public String[] getNames() { return names; } // メニュー名のゲッター

public int[] getPrices() { return prices; } // 値段のゲッター

}

**リスト4：メニュー表示処理（ファイル「J2Kad24D.java」）**

public class J2Kad24D {

public static void main(String[] args) {

…

while (true) {

System.out.print("どのメニューを表示しますか？（0：ECCコーヒー、1：ECCドーナツ、-1：終了）＞");

int shop = Integer.parseInt(in.next());

if (shop < 0) break;

switch(shop) {

default:

case 0: // ECCコーヒー

**ECCコーヒーの全メニューを表示する**

break;

case 1: // ECCドーナツ

**ECCドーナツの全メニューを表示する**

break;

}

System.out.println();

}

}

}

**課題完成時の画面**

世界にはばたくECCフーズ！

ただいまM&Aで拡大中！！

どのメニューを表示しますか？（0：ECCコーヒー、1：ECCドーナツ、-1：終了）＞**0**

ドリップコーヒー：390円

アールグレイ：430円

オレンジジュース：220円

どのメニューを表示しますか？（0：ECCコーヒー、1：ECCドーナツ、-1：終了）＞**1**

ハニーディップ：120円

ハニーチュロ：130円

チョコリング：140円

どのメニューを表示しますか？（0：ECCコーヒー、1：ECCドーナツ、-1：終了）＞**-1**

# **J2Kad24C「世界にはばたくECCフーズ！（ループの本質）」※J2Kad24Dをコピーして作成**

ECCコーヒー・ECCドーナツともにメニュー一覧のデータ構造が異なるため、それぞれにメニュー表示処理を作る必要がある。「う～ん、これは面倒だ」と思っていたところ、「私がECCコーヒーのメニューを教えて差し上げましょう！」という人物が現れた。その名もカフェイテレータ（CafeIterator）。このカフェイテレータに尋ねればECCコーヒーのメニューをひとつずつ教えてくれるそうだ！CafeIteratorを使ってECCコーヒーのメニュー表示処理を修正せよ。同じくECCドーナツに関してもDonutIteratorを導入してメニュー表示処理を修正せよ。

**CafeIteratorクラス（ファイル「CafeMenu.java」に作成）**

|  |  |
| --- | --- |
| メンバ | 説明 |
| private MenuItem[] menu; | Caféメニュー配列への参照。コンストラクタで設定。 |
| private int i = 0; | 配列のインデックス。初期値は0。 |
| public CafeIterator(MenuItem[] menu) | コンストラクタ。menuへの参照を設定する。 |
| public boolean hasNext() | i番目のメニューがあればtrue、なければfalseを返す。 |
| public MenuItem next() | i番目のメニューを返し、iの値を1増やす。 |

**DonutIteratorクラス（ファイル「DonutMenu.java」に作成）**

|  |  |
| --- | --- |
| メンバ | 説明 |
| private String[] names; | Donutのメニュー名配列への参照。コンストラクタで設定。 |
| private int[] prices; | Donutの値段配列への参照。コンストラクタで設定。 |
| private int i = 0; | 配列のインデックス。初期値は0。 |
| public DonutIterator(String[] names, int[] prices) | コンストラクタ。namesとpricesへの参照を設定する。 |
| public boolean hasNext() | i番目のメニューがあればtrue、なければfalseを返す。 |
| public MenuItem next() | i番目のメニューをMenuItemとして返し、iの値を1増やす。 |

ヒント：

ループ構造は「データがあるかどうか（hasNext）」と「データを取得する（next）」で構成することができる。

while( データがある？ ) {

データを取得して処理をする

}

**課題完成時の画面**

（J2Kad24Dと同じ）

# **J2Kad24B1「世界にはばたくECCフーズ！（ループの共通化）」※J2Kad24Cをコピーして作成**

CafeIteratorとDonutIteratorを使うとメニュー表示のループがほぼ同じ形になった。MenuIteratorインターフェイスを導入し（ファイル「MenuItem.java」に作成）、ループ処理を共通化せよ。

**課題完成時のクラス図**

|  |
| --- |
| <<interface>>  ***MenuIterator*** |
|  |
| *hasNext() : boolean*  *next() : MenuItem* |

|  |
| --- |
| **CafeIterator** |
| … |
| … |

|  |
| --- |
| **DonutIterator** |
| … |
| … |

**課題完成時の画面**

（J2Kad24Dと同じ）

# **J2Kad24B2「世界にはばたくECCフーズ！（Iteratorパターン）」※J2Kad24B1をコピー**

CafeMenuとDonutMenuにそれぞれのイテレータを生成する処理（iteratorメソッド）を作成し、Menuインターフェイス（ファイル「MenuItem.java」に新規作成）を導入してイテレータ取得処理を共通化せよ。

**CafeMenuに追加するメソッド（Menuインターフェイスを実装）**

|  |  |
| --- | --- |
| メソッド | 説明 |
| public MenuIterator iterator() | CafeIteratorを生成して返す。 |

**DonutMenuに追加するメソッド（Menuインターフェイスを実装）**

|  |  |
| --- | --- |
| メソッド | 説明 |
| public MenuIterator iterator() | DonutIteratorを生成して返す。 |

**課題完成時のクラス図**

|  |
| --- |
| <<interface>>  ***Menu*** |
|  |
| *iterator() : MenuIterator* |

|  |
| --- |
| **CafeMenu** |
| … |
| … |

|  |
| --- |
| **DonutMenu** |
| … |
| … |

**課題完成時の画面**

（J2Kad24Dと同じ）

# **J2Kad24A「世界にはばたくECCフーズ！（M&A）」※J2Kad24B2をコピーして作成**

世界にはばたくECCフーズが今度はあのハンバーガーチェーンを買収した！ハンバーガーチェーンのメニュー表示を追加せよ。なお、バーガーチェーン（BurgerMenu）にもイテレータ（BurgerIterator）を導入すること。

**リスト1：BurgerMenuクラス（ファイル「BurgerMenu.java」）**

ArrayListで管理している。

public class BurgerMenu {

private ArrayList<MenuItem> menu = new ArrayList<>();

public BurgerMenu() {

menu.add(new MenuItem("ハンバーガー", 150));

menu.add(new MenuItem("チーズバーガー", 180));

menu.add(new MenuItem("ビッグマック", 410));

}

public ArrayList<MenuItem> getMenu() { return menu; }

}

**課題完成時の画面**

世界にはばたくECCフーズ！

ただいまM&Aで拡大中！！

どのメニューを表示しますか？（0：ECCコーヒー、1：ECCドーナツ、2：ECCバーガー、-1：終了）＞**2**

ハンバーガー：150円

チーズバーガー：180円

ビッグマック：410円

どのメニューを表示しますか？（0：ECCコーヒー、1：ECCドーナツ、2：ECCバーガー、-1：終了）＞

# **J2Kad24S「世界にはばたくECCフーズ！（匿名クラス）」※J2Kad24Aをコピーして作成**

CafeIterator・DonutIterator・BurgerIteratorはそれぞれのショップの店員（内部の人間）だった！どうりで各ショップの内部事情に通じていたわけだ。

1. 各イテレータを対応するメニューの内部クラスにせよ（iteratorメソッド内で定義する）。なお、内部クラスにすればprivateなメニューデータを直接参照できるため、簡略化することができる。
2. さらに①で作成した内部クラスを匿名クラスにせよ。

* J2Kad24Aをコピーして作成するが、J2Kad24Sは特に修正する箇所はない（とりあえずAとは別の課題ということで）。
* 各Iteratorクラスは内部クラス・匿名クラスにすると不要になるが、削除せずに残しておくこと（他の課題でエラーが出るので）。

ヒント：内部クラス・匿名クラスについては以下を参照、もしくは検索すること

・（もし教科書を持ってきていれば）実践編P.125～P.130

・J2Kad19D「内部クラス」、J2Kad19C「匿名クラス（無名クラス）」

**課題完成時の画面**

（J2Kad24Aと同じ）

# **J2Kad24X「世界にはばたくECCフーズ！（拡張for文対応）」 ※パッケージpac24xのファイルを修正**

パッケージpac24xに業者が作った恐ろしいメニュー表示処理が準備されている。これまでの知識を動員してリファクタリングせよ。良い子は恐ろしいコードを見てはいけないが、課題を作るために今回は許可する。なお、メニュー表示処理は拡張for文を使って記述すること。

ヒント：

Iterableインターフェイス（←検索）とIteratorインターフェイス（←検索）を実装すれば、拡張for文を使って記述することができる。

**課題完成時の画面**

（J2Kad24Aと同じ）