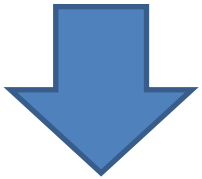
プログラミング基礎

第6.5回

オブジェクト指向の概念
演習問題ヒント

個人的には

- クラスはまず，「使う側」から考えたほうがクラスにどのような属性，メソッドが必要か考えやすい



- 必要なメソッドが完成していると仮定して，mainメソッドから書き始める
- その後必要になったメソッドの中身を記述する

問題1

```
public class BulkBallSimulation {
   public static void main(String[] args) {
      int x, y;         // ボールの位置 (x座標, y座標)
      int vx, vy;       // ボールのx軸方向, y軸方向の速度
      int width, height; // ボールの動く枠の幅と高さ

      // 初期設定
      x = 0; y = 0;
      vx = 1; vy = 1;
      width = 80; height = 40;

      for (int t = 0; t <= 100; t++) {     // t:経過時間(0〜100秒まで)
         // 移動を行った結果ボールが枠外に出そうなら進む方向を逆転
         if (x + vx > width || x + vx < 0)
            vx = -vx;
         // ボールの位置 (x座標) の計算
         x = x + vx;

         // 移動を行った結果ボールが枠外に出そうなら進む方向を逆転
         if (y + vy > height || y + vy < 0)
            vy = -vy;
         // ボールの位置 (y座標) の計算
         y = y + vy;

         System.out.println(t + "秒後のボールの座標: (" + x + ", " + y + ")");
      }
   }
}
```

問題:
ボールをクラスにして,

mainメソッド:
ボールを作り,
(ボールクラスのインスタンスを作り,)
初期情報を与え, 動くよう指示する
(具体的にどのように動くかは
ボールクラス内で処理する)

ボールクラス:
与えられた情報を基に, 初期化,
動くよう指示されたら動く

ように書き換える.

問題1

```
public class BulkBallSimulation {
  public static void main(String[] args) {
    int x, y;        // ボールの位置 (x座標, y座標)
    int vx, vy;      // ボールのx軸方向, y軸方向の速度
    int width, height; // ボールの動く枠の幅と高さ

    // 初期設定
    x = 0; y = 0;
    vx = 1; vy = 1;
    width = 80; height = 40;

    for (int t = 0; t <= 100; t++) {     // t:経過時間(0〜100秒まで)
      // 移動を行った結果ボールが枠外に出そうなら進む方向を逆転
      if (x + vx > width || x + vx < 0)
        vx = -vx;
      // ボールの位置 (x座標) の計算
      x = x + vx;

      // 移動を行った結果ボールが枠外に出そうなら進む方向を逆転
      if (y + vy > height || y + vy < 0)
        vy = -vy;
      // ボールの位置 (y座標) の計算
      y = y + vy;

      System.out.println(t + "秒後のボールの座標: (" + x + ", " + y + ")");
    }
  }
}
```

ボールに関する情報（位置，速度）
ボールが動くために必要な情報（枠の幅，高さ）
⇒ボールクラスの**属性**とする

また，インスタンスを作らなければいけない
例：
Ball ball;
ball = new Ball();

問題1

```
public class BulkBallSimulation {
   public static void main(String[] args) {
      int x, y;        // ボールの位置 (x座標, y座標)
      int vx, vy;      // ボールのx軸方向, y軸方向の速度
      int width, height; // ボールの動く枠の幅と高さ

      // 初期設定
      x = 0; y = 0;
      vx = 1; vy = 1;
      width = 80; height = 40;

      for (int t = 0; t <= 100; t++) {     // t:経過時間(0〜100秒まで)
         // 移動を行った結果ボールが枠外に出そうなら進む方向を逆転
         if (x + vx > width || x + vx < 0)
            vx = -vx;
         // ボールの位置 (x座標) の計算
         x = x + vx;

         // 移動を行った結果ボールが枠外に出そうなら進む方向を逆転
         if (y + vy > height || y + vy < 0)
            vy = -vy;
         // ボールの位置 (y座標) の計算
         y = y + vy;

         System.out.println(t + "秒後のボールの座標: (" + x + ", " + y + ")");
      }
   }
}
```

位置，速度，動く範囲の設定（初期化）
例：
ball.SetPosition(0, 0);
ball.SetVelocity(1, 1);
ball.SetBoundary(80, 40);

問題1

```
public class BulkBallSimulation {
   public static void main(String[] args) {
      int x, y;        // ボールの位置 (x座標, y座標)
      int vx, vy;      // ボールのx軸方向, y軸方向の速度
      int width, height; // ボールの動く枠の幅と高さ

      // 初期設定
      x = 0; y = 0;
      vx = 1; vy = 1;
      width = 80; height = 40;

      for (int t = 0; t <= 100; t++) {     // t:経過時間(0〜100秒まで)
         // 移動を行った結果ボールが枠外に出そうなら進む方向を逆転
         if (x + vx > width || x + vx < 0)
            vx = -vx;
         // ボールの位置 (x座標) の計算
         x = x + vx;

         // 移動を行った結果ボールが枠外に出そうなら進む方向を逆転
         if (y + vy > height || y + vy < 0)
            vy = -vy;
         // ボールの位置 (y座標) の計算
         y = y + vy;

         System.out.println(t + "秒後のボールの座標: (" + x + ", " + y + ")");
      }
   }
}
```

ボールを動かす処理
⇒ボールクラス内部で行えばよい

例 :
ball.Move();

問題1

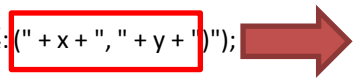
```
public class BulkBallSimulation {
   public static void main(String[] args) {
      int x, y;        // ボールの位置 (x座標, y座標)
      int vx, vy;      // ボールのx軸方向, y軸方向の速度
      int width, height; // ボールの動く枠の幅と高さ

      // 初期設定
      x = 0; y = 0;
      vx = 1; vy = 1;
      width = 80; height = 40;

      for (int t = 0; t <= 100; t++) {     // t:経過時間(0〜100秒まで)
         // 移動を行った結果ボールが枠外に出そうなら進む方向を逆転
         if (x + vx > width || x + vx < 0)
            vx = -vx;
         // ボールの位置 (x座標) の計算
         x = x + vx;

         // 移動を行った結果ボールが枠外に出そうなら進む方向を逆転
         if (y + vy > height || y + vy < 0)
            vy = -vy;
         // ボールの位置 (y座標) の計算
         y = y + vy;

         System.out.println(t + "秒後のボールの座標: (" + x + ", " + y + ")");
      }
   }
}
```

ボールを動かす処理
⇒ボールクラス内部で行えばよい

例 :
ball.Move();

問題1

```
public class BulkBallSimulation {
   public static void main(String[] args) {
      int x, y;        // ボールの位置 (x座標, y座標)
      int vx, vy;      // ボールのx軸方向, y軸方向の速度
      int width, height; // ボールの動く枠の幅と高さ

      // 初期設定
      x = 0; y = 0;
      vx = 1; vy = 1;
      width = 80; height = 40;

      for (int t = 0; t <= 100; t++) {     // t:経過時間(0〜100秒まで)
         // 移動を行った結果ボールが枠外に出そうなら進む方向を逆転
         if (x + vx > width || x + vx < 0)
            vx = -vx;
         // ボールの位置 (x座標) の計算
         x = x + vx;

         // 移動を行った結果ボールが枠外に出そうなら進む方向を逆転
         if (y + vy > height || y + vy < 0)
            vy = -vy;
         // ボールの位置 (y座標) の計算
         y = y + vy;

         System.out.println(t + "秒後のボールの座標:(" + x + ", " + y + ")");
      }
   }
}
```



ボール位置は，クラスの属性としたので，
属性の値を得るメソッドが必要

例：

```
int x, y;
x = ball.GetX();
y = ball.GetY();
```

問題1

まとめると

```
public class BulkBallSimulation {
   public static void main(String[] args) {
      Ball ball;
      ball = new Ball();
      // 初期設定
      ball.SetPosition(0, 0);
      ball.SetVelocity(1, 1);
      ball.SetBoundary(80, 40);

      for (int t = 0; t <= 100; t++) {     // t:経過時間(0〜100秒まで)
         ball.Move();

         int x, y;
         x = ball.GetX();
         y = ball.GetY();
         System.out.println(t + "秒後のボールの座標: (" + x + ", " + y + ")");
      }
   }
}
```

あとはBallクラスを作りましょう

問題1

まとめると

```
public class BulkBallSimulation {
   public static void main(String[] args) {
      Ball ball;
      ball = new Ball();
      // 初期設定
      ball.SetPosition(0, 0);
      ball.SetVelocity(1, 1);
      ball.SetBoundary(80, 40);

      for (int t = 0; t <= 100; t++) {     // t:経過時間(0～100秒まで)
         ball.Move();

         int x, y;
         x = ball.GetX();
         y = ball.GetY();
         System.out.println(t + "秒後のボールの座標: (" + x + ", " + y + ")");
      }
   }
}
```

クラス，属性の書き方，使い方は第4回資料参照

メソッドの書き方，使い方は第5回資料参照

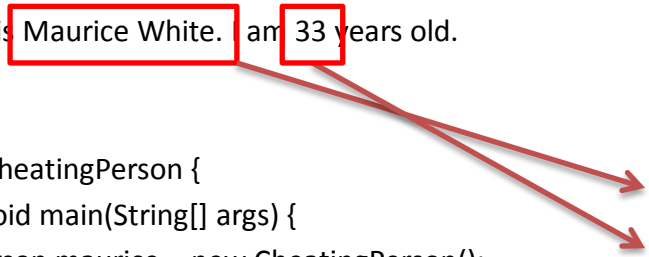
あとはBallクラスを作りましょう

問題2

年齢を聞かれるとサバをよむ人を作る.

実行結果:

Hello, my name is Maurice White. I am 33 years old.



```
class IntroduceCheatingPerson {
   public static void main(String[] args) {
      CheatingPerson maurice = new CheatingPerson();

      maurice に対して名前と実年齢を登録 ;

      // maurice の自己紹介を行う
      System.out.println("Hello, my name is " + ..... );
   }
}
```

クラスの属性には何が必要か？

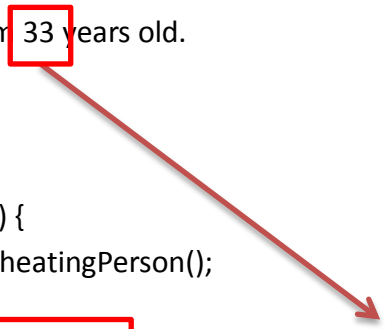
- **名前**
- サバをよんだ年齢は実年齢から算出するので，**実年齢**が必要

問題2

実行結果:

Hello, my name is Maurice White. I am 33 years old.

```
class IntroduceCheatingPerson {
   public static void main(String[] args) {
      CheatingPerson maurice = new CheatingPerson();

      maurice に対して名前と実年齢を登録 ;

      // maurice の自己紹介を行う
      System.out.println("Hello, my name is " + ..... );
   }
}
```



例:

```
maurice.SetName("Maurice White");
//サバをよんで33歳なので
//実年齢は34歳
maurice.SetYears(34);
```

問題2

実行結果:
Hello, my name is Maurice White. I am 33 years old.

```
class IntroduceCheatingPerson {
   public static void main(String[] args) {
      CheatingPerson maurice = new CheatingPerson();

      maurice に対して名前と実年齢を登録 ;

      // maurice の自己紹介を行う
      System.out.println("Hello, my name is " + ..... );
   }
}
```

実行結果のとおり出力する

例:

```
String name = maurice.GetName();
int sabaYears = maurice.GetSabaYears();
System.out.println("Hello, my name is"
 + name + ". I am "
 + sabaYears + " years old.");
```

問題2

実行結果:

Hello, my name is Maurice White. I am 33 years old.

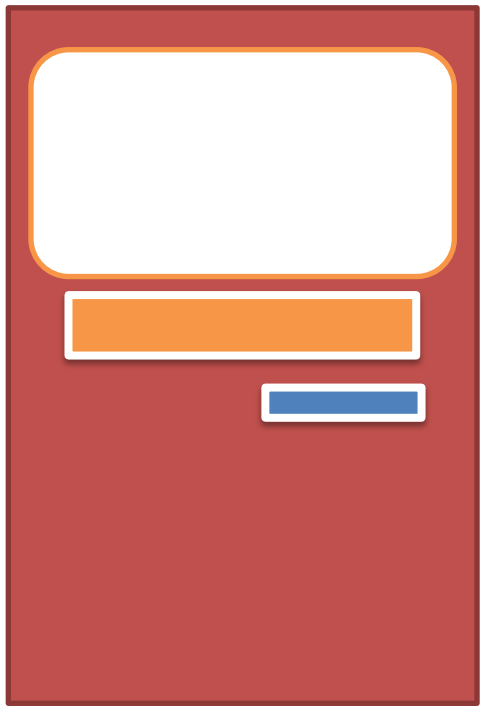
まとめると

```
class IntroduceCheatingPerson {
   public static void main(String[] args) {
      CheatingPerson maurice = new CheatingPerson();

      maurice.SetName("Maurice White");
      //サバをよんで33歳なので実年齢は34歳
      maurice.SetYears(34);

      // maurice の自己紹介を行う
      String name = maurice.GetName();
      int sabaYears = maurice.GetSabaYears();
      System.out.println("Hello, my name is" + name + ". I am "  + sabaYears + " years old.");
   }
}
```

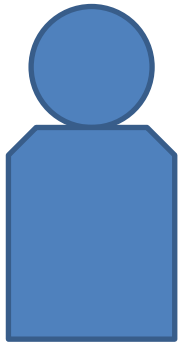
あとはCheatingPersonクラスを作りましょう

問題3

- 自販機（クラス）を作る



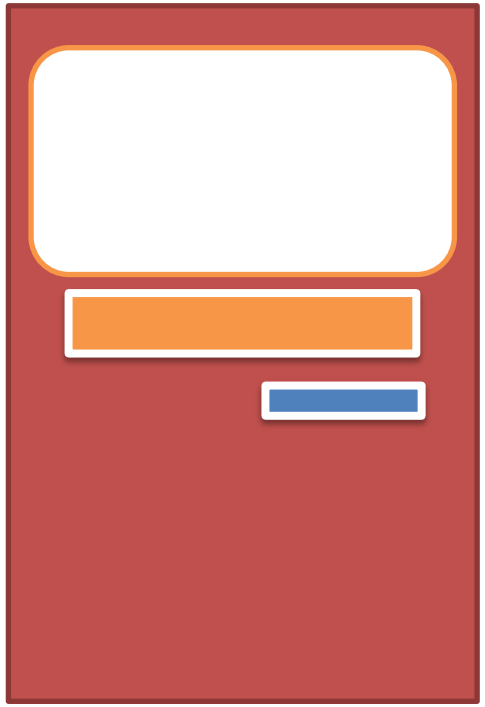
自販機



客

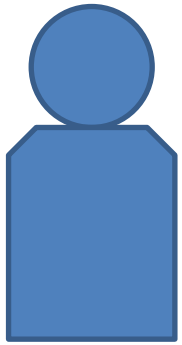
問題3

- 自販機（クラス）を作る



入力
・金を投入する
・購入ボタンを押す

出力
・合計投入金額の表示
・不足金額，もしくはつり銭を表示

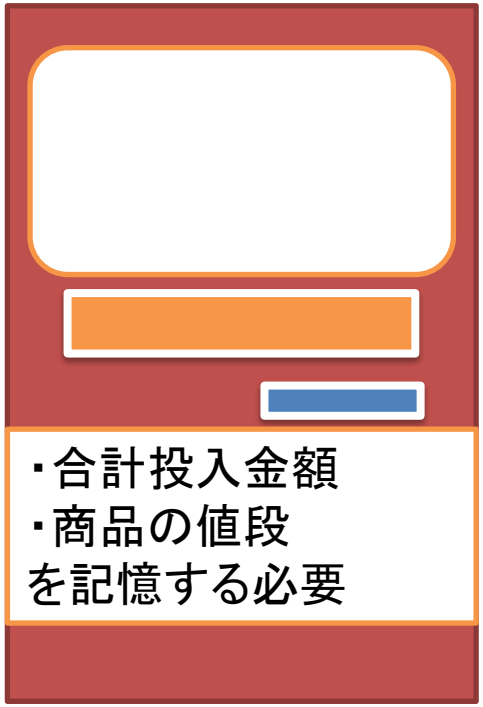


自販機

客

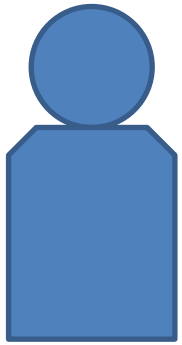
問題3

- 自販機（クラス）を作る



入力
・金を投入する
・購入ボタンを押す

出力
・合計投入金額の表示
・不足金額，もしくはつり銭を表示



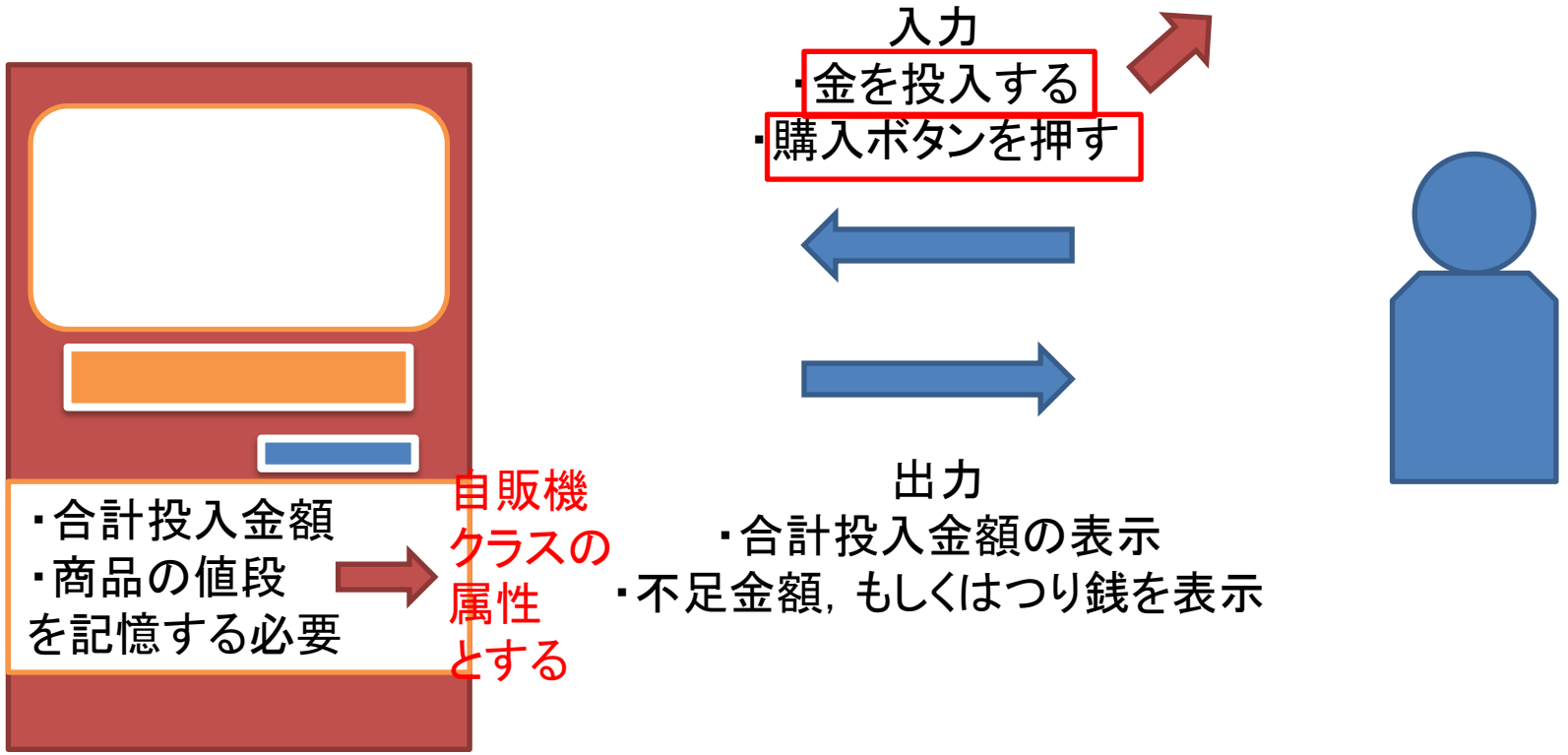
自販機

客

問題3

- 自販機（クラス）を作る

この2つを自販機クラスのメソッドにすればよい



問題3

```
class VendingMachine
{
    //属性
    合計投入金額;
    商品の値段;

    //メソッド
    お金を入れる(InsertCoinメソッド);
    購入ボタンを押す(Buyメソッド);
}
```

問題3

- 実行結果例:

$ java VendingMachineUser
100 円を入れます。
-----> 投入合計金額は 100 円です。

購入ボタンを押します。
-----> 20 円不足しています。

50 円を入れます。
-----> 投入合計金額は 150 円です。

購入ボタンを押します。
-----> ジュースを出します。
-----> 30 円のおつりを出します。

問題3

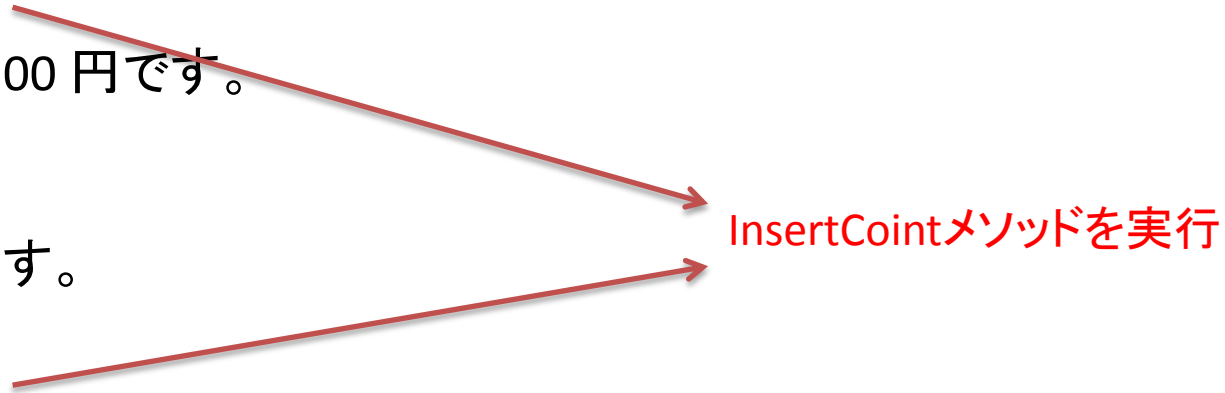
- 実行結果例:

$ java VendingMachineUser
100 円を入れます。
-----> 投入合計金額は 100 円です。

購入ボタンを押します。
-----> 20 円不足しています。

50 円を入れます。
-----> 投入合計金額は 150 円です。

購入ボタンを押します。
-----> ジュースを出します。
-----> 30 円のおつりを出します。



問題3

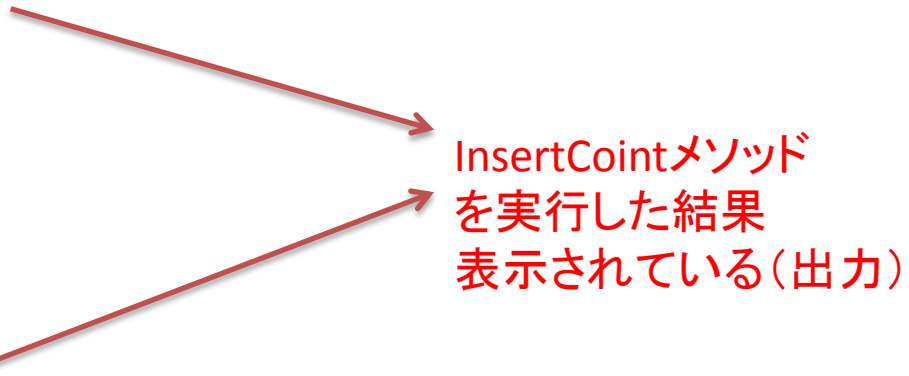
- 実行結果例:

$ java VendingMachineUser
100 円を入れます。
-----> 投入合計金額は 100 円です。

購入ボタンを押します。
-----> 20 円不足しています。

50 円を入れます。
-----> 投入合計金額は 150 円です。

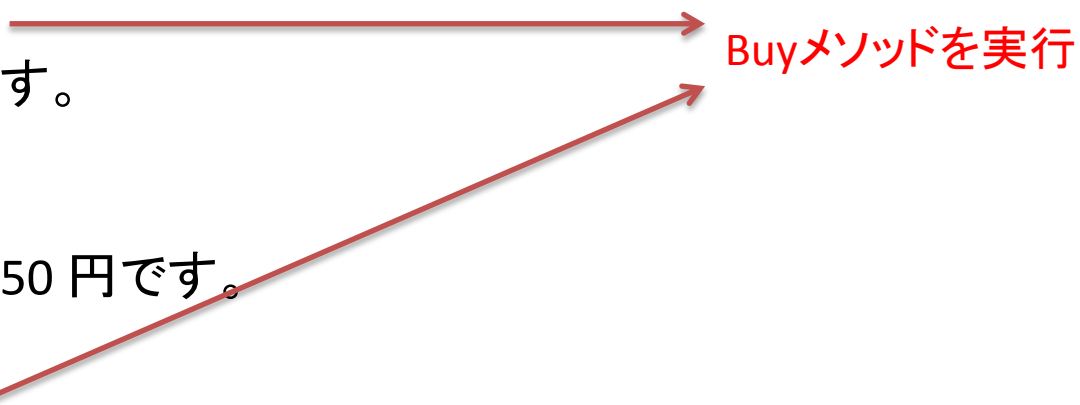
購入ボタンを押します。
-----> ジュースを出します。
-----> 30 円のおつりを出します。



問題3

- 実行結果例:

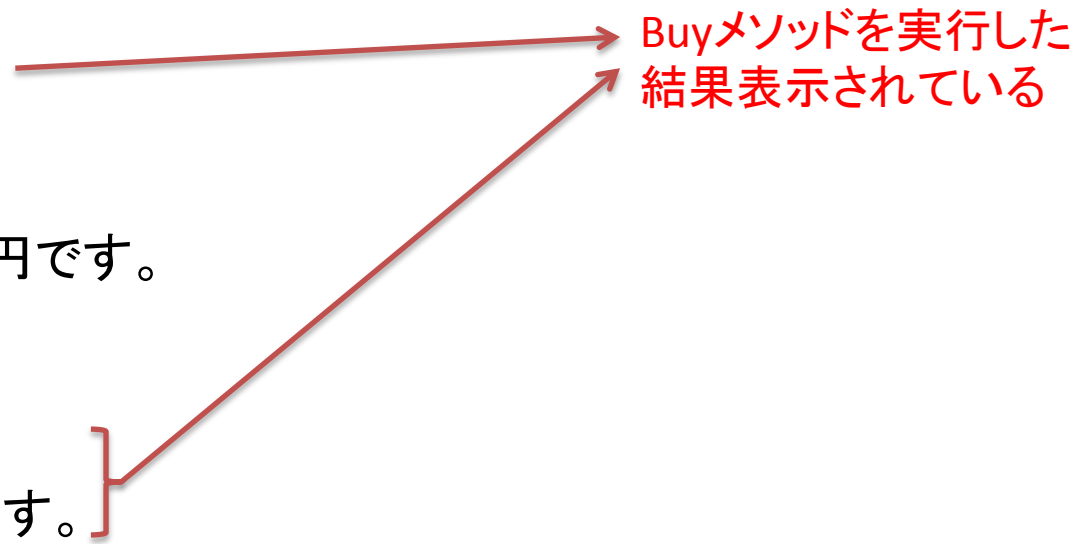
```
$ java VendingMachineUser
100 円を入れます。
-----> 投入合計金額は 100 円です。

購入ボタンを押します。
-----> 20 円不足しています。

50 円を入れます。
-----> 投入合計金額は 150 円です。

購入ボタンを押します。
-----> ジュースを出します。
-----> 30 円のおつりを出します。
```



問題3

- 実行結果例:

```
$ java VendingMachineUser
100 円を入れます。
-----> 投入合計金額は 100 円です。

購入ボタンを押します。
-----> 20 円不足しています。

50 円を入れます。
-----> 投入合計金額は 150 円です。

購入ボタンを押します。
-----> ジュースを出します。
-----> 30 円のおつりを出します。
```



問題3

```
public class VendingMachineUser
{
   public static void main(String[] args)
   {
      VendingMachine v;
      v = new VendingMachine();

      int firstCoins = 100;
      System.out.println(firstCoins + "円を入れます。");
      v.InsertCoins(firstCoins);

      System.out.println("購入ボタンを押します。 ");
      v.Buy();

       int secondCoins = 50;
      System.out.println(secondCoins + "円を入れます。");
      v.InsertCoins(secondCoins);

      System.out.println("購入ボタンを押します。 ");
      v.Buy();
   }
}
```

あとはVendingMachineクラスを作りましょう