

Javaプログラミング実習

06. 分岐処理1

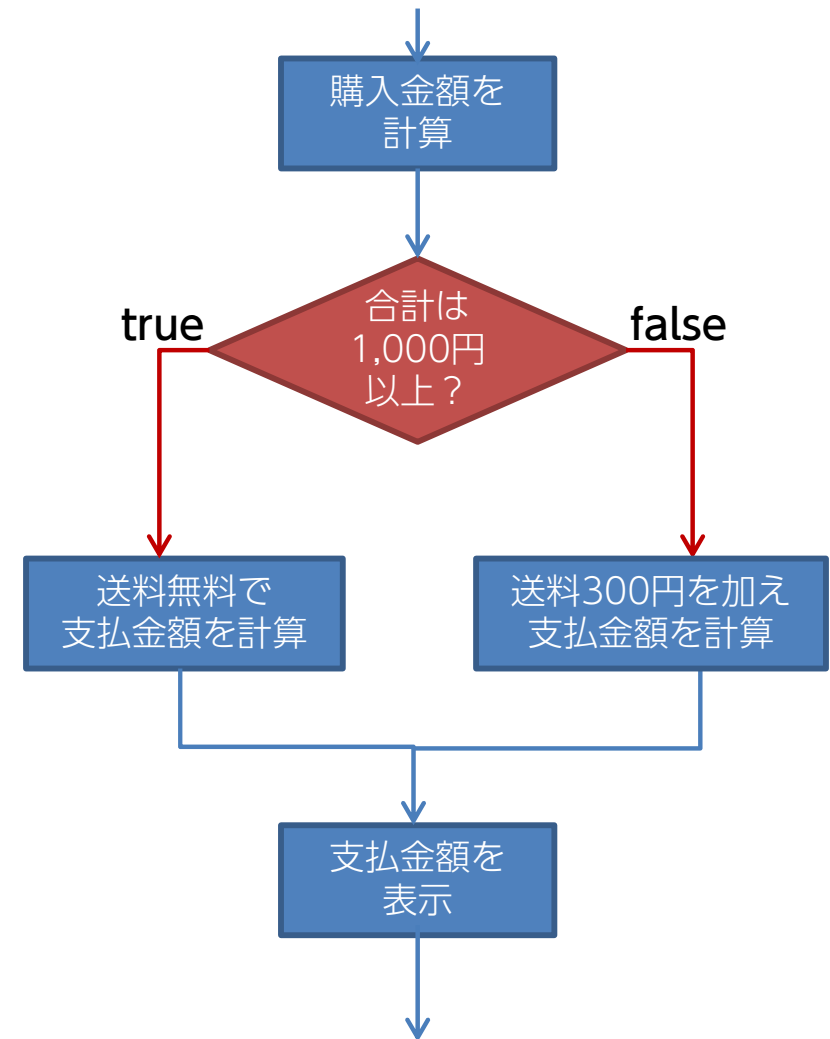
株式会社ジードライブ

今回学ぶこと

- if文を使用した分岐構造の記述方法

分岐構造

- プログラムの作成においては、条件に応じて処理を分けたい場面に遭遇する
 - 購入金額が1,000円以上の場合は送料無料にする
 - 20歳未満の場合は、お酒が購入できないようにする…等
- Javaではif文やswitch文によって、条件に応じて処理を分ける分岐構造を作ることができる



if文

書式

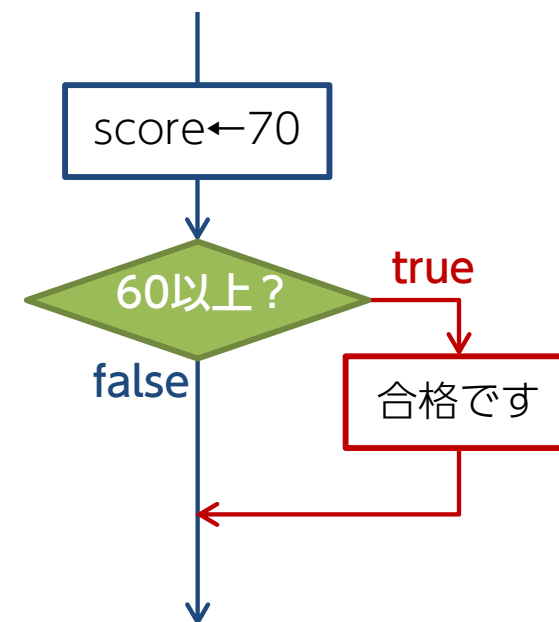
```
if (条件) {  
    // 条件に適合する(条件がtrue)場合の処理  
}
```

例

```
int score = 70;  
if (score >= 60) {  
    System.out.println("合格です");  
}
```



「変数scoreの中身が60以上」という条件に適合するので「合格です」と表示される



if文

- if文のカッコ内に記述する条件では、>や<=のような関係演算子を使用する
- 条件がtrueの場合に実行するステートメントが1つだけの場合は { } は省略できる
 - 基本的には省略しない方がよい

省略した場合の記述例

```
int score = 70;  
if (score >= 60)  
    System.out.println("合格です");
```

練習

- 練習06-1

boolean型と関係演算子

- 関係演算子による式を評価した結果はboolean型になる
 - boolean型は true / false のどちらかの値しかとらない

```
System.out.println(3 == 5); // false
```

関係演算子	意味	例	評価結果
==	等しい	3 == 5	false
!=	等しくない	3 != 5	true
>	より大きい	3 > 5	false
>=	以上	3 >= 5	false
<	より小さい	3 < 5	true
<=	以下	3 <= 5	true

論理演算子

- 複数の条件が同時に成立した場合、複数の条件のいずれかが一つが成立した場合、などの状況を表す際に使用する演算子

論理演算子

演算子	意味	例	評価結果
&&	AND (且つ)	<code>3 > 5 && 1 < 2</code>	false
	OR (または)	<code>3 > 5 1 < 2</code>	true
!	NOT (否定)	<code>!(3 > 5)</code>	true

論理演算子の使用例

例

```
int score = 70;
if (score >= 80) {
    System.out.println("優秀です");
}
if (score >= 60 && score < 80) {
    System.out.println("合格です");
}
if (score < 60) {
    System.out.println("不合格です");
}
```

練習

- 練習06-2

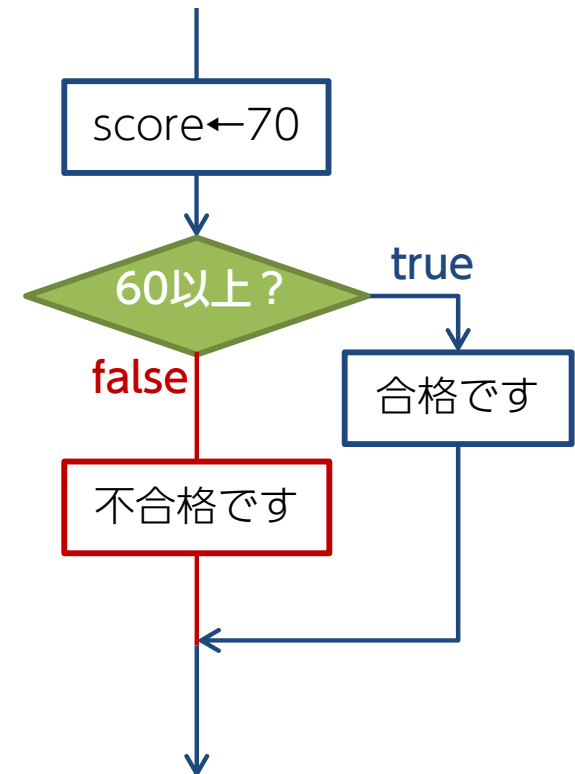
else

- elseのブロックには、条件に適合しない(false)場合に実行する処理を記述する

例

```
int score = 70;  
if (score >= 60) {  
    System.out.println("合格です");  
} else {  
    System.out.println("不合格です");  
}
```

false時の処理がない場合は
elseブロックを省略できる

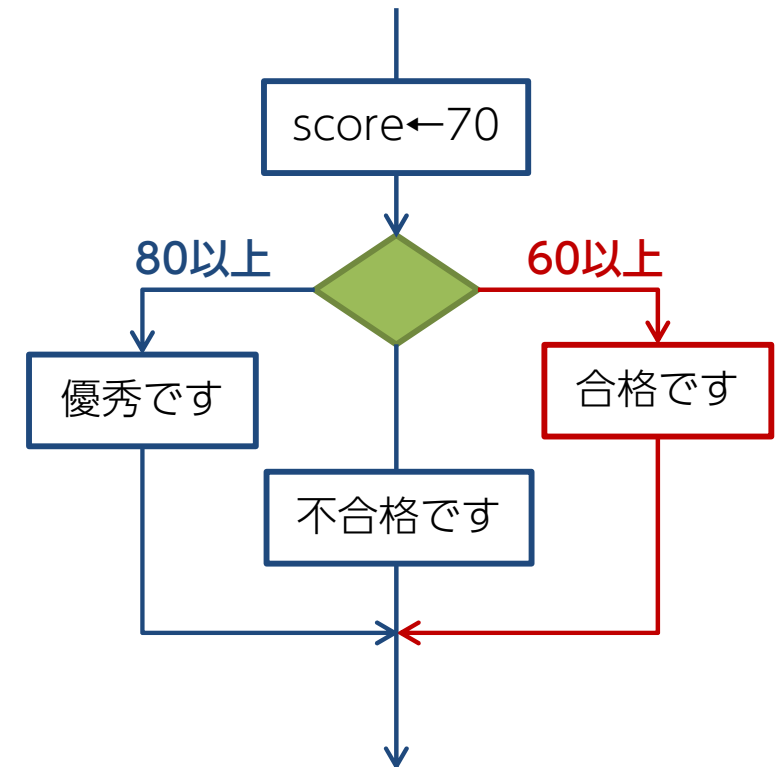


else if

- elseとifを組み合わせることで、複数の条件を表現できる

例

```
int score = 70;  
if (score >= 80) {  
    System.out.println("優秀です");  
} else if (score >= 60) {  
    System.out.println("合格です");  
} else {  
    System.out.println("不合格です");  
}
```

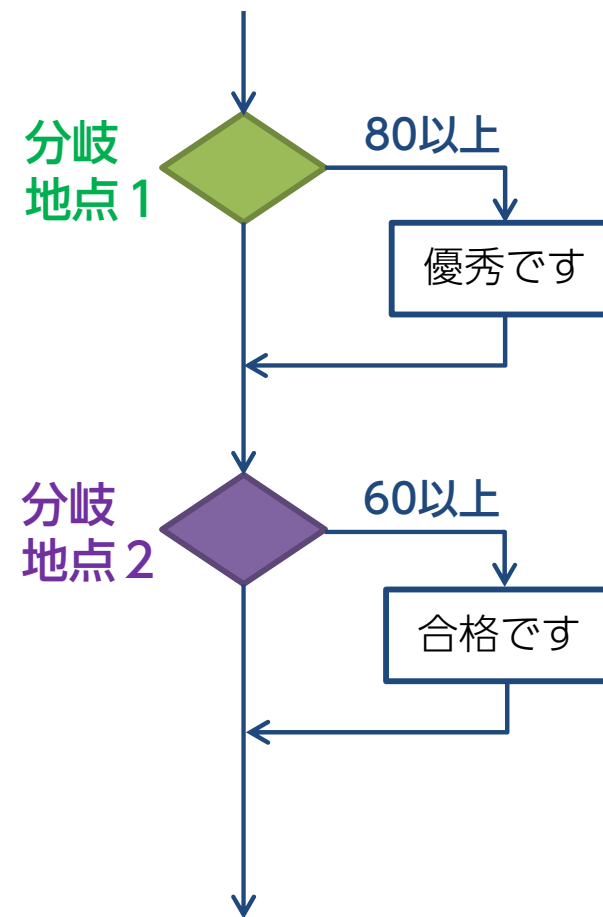


複数のif

- else ifとは異なり、複数のifでは分岐地点が複数できる

例

```
if (score >= 80) {  
    System.out.println("優秀です");  
}  
if (score >= 60) {  
    System.out.println("合格です");  
}
```



比較の順番

- `else if` を使う場合、比較の順番を間違えると期待する結果が得られないことがある
 - 比較は上部の記述から行われるため、基本的には数値の大きい順、または小さい順に条件を記述する

例

```
int score = 90;  
if (score >= 60) {  
    System.out.println("合格です");  
} else if (score >= 80) {  
    System.out.println("優秀です");  
}
```

60点以上という条件を先に満たしてしまうため「優秀です」とは表示されず「合格です」と表示される

練習

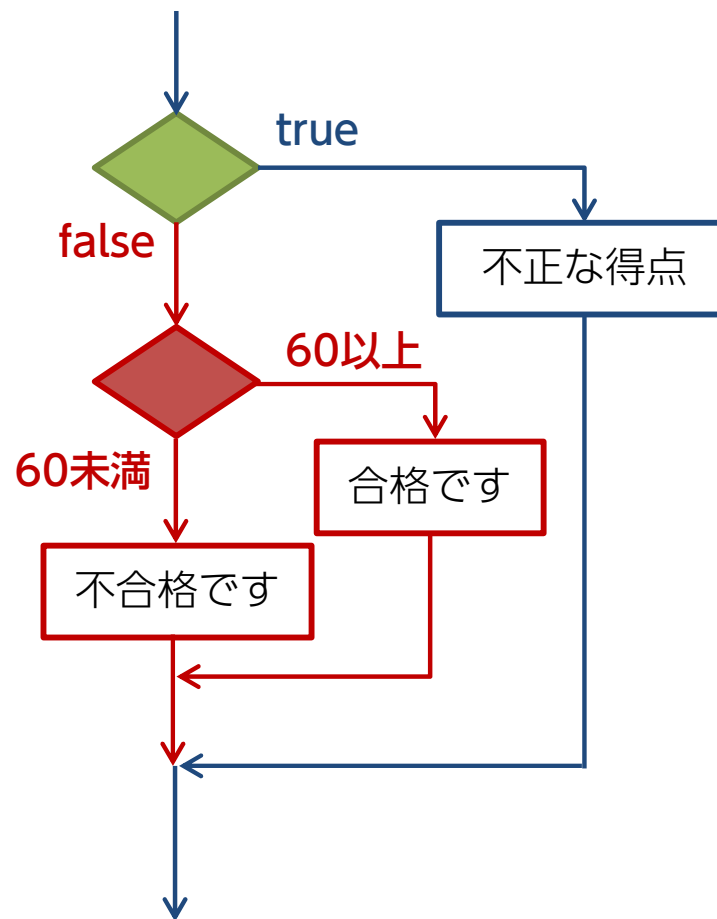
- 練習06-3

If文の入れ子

- 分岐構造の中に分岐を作ることも可能

例

```
if (score > 100 || score < 0) {  
    System.out.println("不正な得点です");  
} else {  
    if (score >= 60) {  
        System.out.println("合格です");  
    } else {  
        System.out.println("不合格です");  
    }  
}
```



練習

- 練習06-4

文字列の比較

- 文字列同士が等しいか否かを判定するには、Stringクラスの`equals()`メソッドを利用する

書式 (文字列Aと文字列Bの比較)

文字列A.`equals`(文字列B)

文字列とequalsをドットでつなぐ

例

```
System.out.println("日本語がわかりますか? yes/no")
String lang = scanner.nextLine();
if (lang.equals("yes")) {
    System.out.println("こんにちは");
} else {
    System.out.println("Hello");
}
```

文字列の比較

- 文字列同士が等しくないことを示す場合、「！」を使用する

例

```
System.out.println("パスワードを入力してください");  
String password = scanner.nextLine();  
  
if (!password.equals("abcd")) {  
    System.out.println("パスワードが不正です");  
}
```

練習

- 練習06-5