関数型言語（OCaml 演習）(2)

条件文、デザインレシピ

浅井 健一

お茶の水女子大学

真偽値（bool 型）と比較・論理演算

```
# true ;;
- : bool = true
# false ;;
- : bool = false
# 2 < 3 ;;
- : bool = true
# 2 > 3 ;;
- : bool = false
```

```
# 2 = 3 ;;
- : bool = false
# 2 + 3 = 5 ;;
- : bool = true
# 2.71 >= 3.14 ;;
- : bool = false
# "hello" < "world" ;;
- : bool = true
```

比較演算：<, <=, >, >=, =, <>

同じ型のデータなら（ほぼ）全てのデータを比較可能。

論理演算：&&（かつ），||（または），not（否定）

条件文（if 文）

if 条件 then 式$_1$ else 式$_2$

条件部が true になったら式$_1$ の値を、false になったら式$_2$ の値を返す。

```
# if 2 < 3 then "a" else "b" ;;
- : string = "a"
# let abs_value x = if x > 0.0 then x
                                else -. x ;;
val abs_value : float -> float = <fun>
# abs_value 3.14 ;;
- : float = 3.14
# abs_value (-0.5) ;;
- : float = 0.5
```

条件文（if 文）

if 条件 then 式$_1$ else 式$_2$

条件部が true になったら式$_1$ の値を、false になったら式$_2$ の値を返す。

```
# if 2 < 3 then "a" else "b" ;;
- : string = "a"
# let abs_value x = if x > 0.0 then x
                                else -. x ;;
val abs_value : float -> float = <fun>
```

- 条件部は bool 型でなくてはならない。
- 式$_1$ と式$_2$ は同じ型でなくてはならない。

関数定義のためのデザインレシピ

超重要

目的	関数の目的を考え、ヘッダを作成する。
例	関数の入出力の例を作成する。
本体	関数本体を作成する。
テスト	作った関数の動作を確認する。

関数定義のためのデザインレシピ

超重要

目的	関数の目的を考え、ヘッダを作成する。
例	関数の入出力の例を作成する。
本体	関数本体を作成する。
テスト	作った関数の動作を確認する。

問題

ひとつ 126 円（税込み）のチョコレートをなるべくたくさん買いたい。所持金を与えられたときに、いくつ買えるかを返す関数 `chocolate` を定義せよ。

目的

関数の目的を考え、ヘッダを作成する。

```
(* 目的：所持金が与えられたとき 126 円の
     チョコレートをいくつ買えるかを求める *)
(* chocolate : int -> int *)
let chocolate x = 0
```

問題

ひとつ 126 円（税込み）のチョコレートをなるべくたくさん買いたい。所持金を与えられたときに、いくつ買えるかを返す関数 chocolate を定義せよ。

例

関数の入出力の例を作成する。

```
(* テスト *)
let test1 = chocolate 100 = 0
let test2 = chocolate 252 = 2
let test3 = chocolate 500 = 3
```

問題

ひとつ 126 円（税込み）のチョコレートをなるべくたくさん買いたい。所持金を与えられたときに、いくつ買えるかを返す関数 `chocolate` を定義せよ。

本体

関数本体を作成する。

```
(* 目的：所持金が与えられたとき 126 円の
   チョコレートをいくつ買えるかを求める *)
(* chocolate : int -> int *)
let chocolate x = x / 126
```

問題

ひとつ 126 円（税込み）のチョコレートをなるべくたくさん買いたい。所持金を与えられたときに、いくつ買えるかを返す関数 `chocolate` を定義せよ。

テスト

作った関数の動作を確認する。

```
# #use "chocolate.ml" ;;
val chocolate : int -> int = <fun>
val test1 : bool = true
val test2 : bool = true
val test3 : bool = true
```

問題

ひとつ 126 円（税込み）のチョコレートをなるべくたくさん買いたい。所持金を与えられたときに、いくつ買えるかを返す関数 `chocolate` を定義せよ。

関数定義のためのデザインレシピ

超重要

目的	関数の目的を考え、ヘッダを作成する。
例	関数の入出力の例を作成する。
本体	関数本体を作成する。
テスト	作った関数の動作を確認する。

問題

ひとつ 126 円（税込み）のチョコレートをなるべくたくさん買いたい。所持金を与えられたときに、いくつ買えるかを返す関数 `chocolate` を定義せよ。

まとめ

- 真偽値（bool 型）。true false
- 比較演算。<, <=, >, >=, =, <>
- 論理演算。&&, ||, not
- 条件文。if 条件 then 式$_1$ else 式$_2$
- 条件部は bool 型。式$_1$ と 式$_2$ は同じ型。

デザインレシピ

目的	関数の目的を考え、ヘッダを作成する。
例	関数の入出力の例を作成する。
本体	関数本体を作成する。
テスト	作った関数の動作を確認する。