第4回
条件式と条件関数

情報処理演習II

Literacy

例題

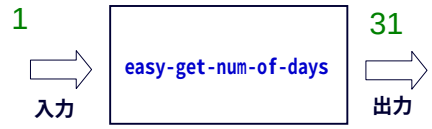
Literacy

例題1. 月の日数

- 月 month を受け取り(数字)、その日数を求める関数 easy-get-num-of-days を作る。ただし、うるう年のことは考えない。
  - まず月ごとに12通りに分けて考えてみよ
  - 日数は3通りしかない。3つの節を持つ cond 式で考えよ
  - 比較演算と論理演算を組み合わせる

Literacy

入力と出力



入力は
1つの数値

出力は
1つの数値

Literacy

easy-get-num-of-days 関数

「関数である」ことを示すキーワード　関数の名前

```
(define (easy-get-num-of-days month)
  (cond
    [(= month 2) 28]
    [(or (= month 4)
         (= month 6)
         (= month 9)
         (= month 11)) 30]
    [else 31]))
```

値を1つ受け取る（入力）

Literacy

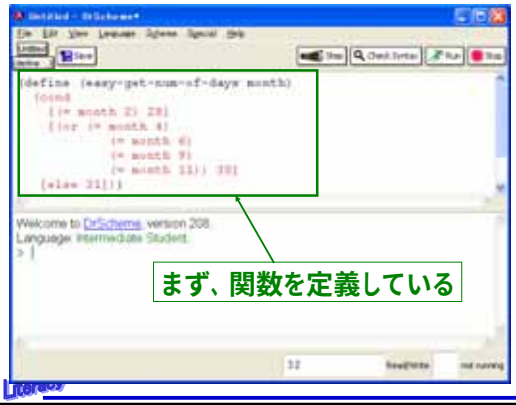
easy-get-num-of-days 関数

節

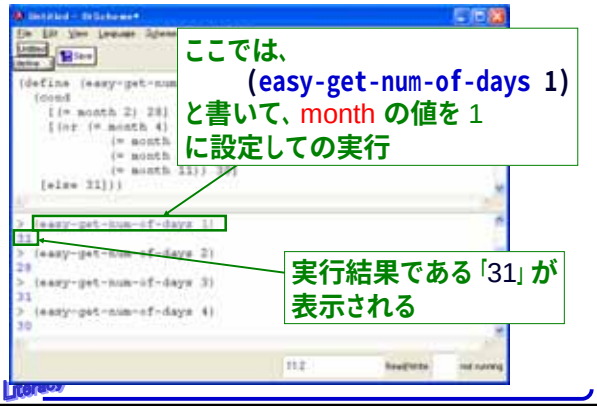
```
(define (easy-get-num-of-days month)
  (cond
    [(= month 2) 28]
    [(or (= month 4)
         (= month 6)
         (= month 9)
         (= month 11)) 30]
    [else 31]))
```

Literacy

easy-get-num-of-days 関数の実行例(1/2)



easy-get-num-of-days 関数の実行例(2/2)



例題2. うるう年の判定

- 西暦年 year から、うるう年であるかを求める関数 is-leap? を作り、実行する
  - うるう年の判定のために、比較演算と論理演算を組み合わせる
  - 西暦年が 4、100、400 の倍数であるかを調べるために remainder を使う

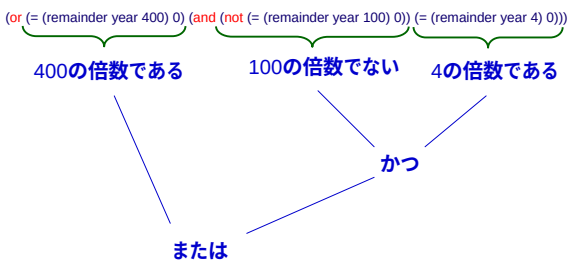
例) 2001 → うるう年でない

2004 → うるう年である

グレゴリオ暦でのうるう年

- うるう年とは： 2月が29日まである年
- うるう年は400年に97回で、1年の平均日数は365.2422日
- うるう年の判定法
  - 4の倍数の年ならうるう年
  - 但し、100の倍数の年で400の倍数でない年はうるう年ではない (4の倍数だが例外)
    - 2005年はうるう年でない(4の倍数でない)
    - 2004年はうるう年(4の倍数)
    - 2000年はうるう年(100の倍数で例外にも当てはまらない)
    - 1900年はうるう年でない(4の倍数で100の倍数だが400の倍数でないので例外に相当)
  - 年がある数の倍数かを調べるためremainderを使う

うるう年の判定式



入力と出力

出力は
true あるいは false 値

is-leap? 関数

「関数である」ことを示すキーワード　関数の名前　値を1つ受け取る(入力)

```
(define (is-leap? year)
  (cond
    [(or (= (remainder year 400) 0)
         (and (not (= (remainder year 100) 0))
              (= (remainder year 4) 0))) true]
    [else false]))
```

Literacy

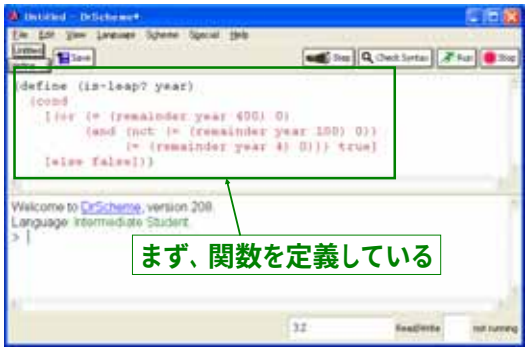
is-leap? 関数

節

```
(define (is-leap? year)
  (cond
    [(or (= (remainder year 400) 0)
         (and (not (= (remainder year 100) 0))
              (= (remainder year 4) 0))) true]
    [else false]))
```

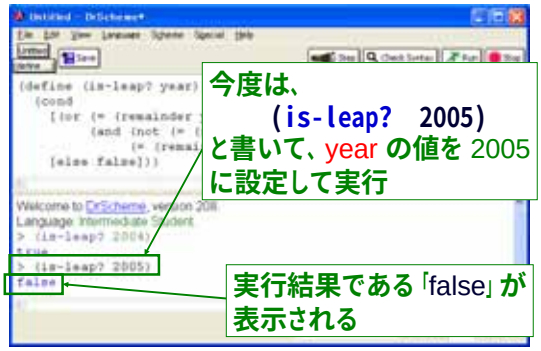
Literacy

is-leap? 関数の実行例(1/2)



Literacy

is-leap? 関数の実行例(2/2)



Literacy

例題3. 曜日を求める

- 西暦の年、月、日から曜日を求める公式であるツエラー(zellar)の公式のプログラムを作る。
  - ツエラーの公式では「1年の起点を3月とし、月は3月から14月までである」と考えている。(うるう年があるので、1年の起点を3月とする方が計算が簡単)
  - 1月、2月は、前年の13月、14月と考え、月は3から14までの数になる。
  - 計算された曜日は「数字」で次の意味になる。

0	1	2	3	4	5	6
日曜	月曜	火曜	水曜	木曜	金曜	土曜

Literacy

ツエラーの公式(数式表現)

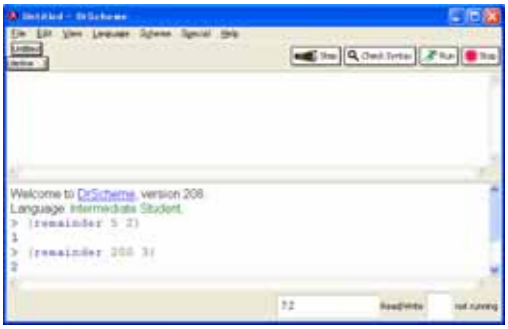
```
[(y+[y/4]+[y/400]-[y/100]+[(13m+8)/5]+d) mod 7]
```

ここで y: 年、d: 日、m: 月(3月を起点とする月で値は3から14までの値をとる(1月、2月は前年の13、14月と考える))

- この計算の結果値が0なら日曜、1なら月曜・・・
- 式中 [...] とあるのは、小数点以下切り捨て
- mod は剰余を計算。例えば 2005 mod 4 は 1

Literacy

remainder



- remainder の意味:
  - 割り算の余りを求める
  - (remainder x y) はツエラーの公式の x mod y に対応

Literacy

quotient

- quotient の意味:
  - 整数の割り算の商を求める
  - (quotient x y) は、ツエラーの公式の [x / y] に対応

Literacy

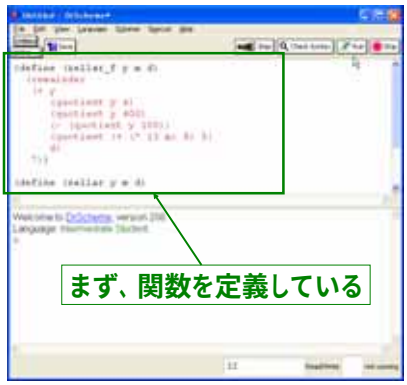
ツエラーの公式のプログラム

[(y+[y/4]+[y/400]-[y/100]+[(13m+8)/5]+d) mod 7]

```
(define (zellar_f y m d)
    (remainder
        (+ y
            (quotient y 4)
            (quotient y 400)
            (- (quotient y 100))
            (quotient (+ (* 13 m) 8) 5)
            d)
        7))
;; zellar : number number number -> number
;; to determine youbi (the day of the week) from year, month, and day.
;; The result is 0 ... 6 meaning 0 is Sunday, 1 Monday, and so on
(define (zellar y m d)
    (cond
        [(or (= m 1) (= m 2)) (zellar_f (- y 1) (+ m 12) d)]
        [else (zellar_f y m d)]))
```

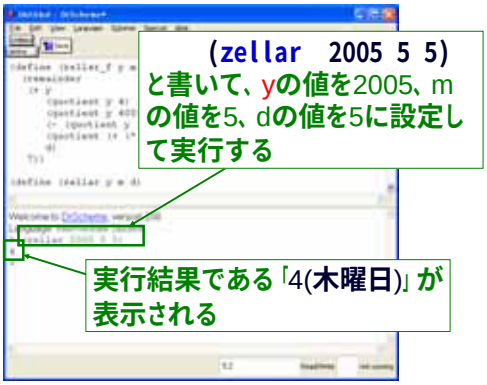
Literacy

ツェラーの公式の実行例(1/2)



Literacy

ツェラーの公式の実行例(2/2)



Literacy

実習

Literacy

実習の進め方

- 資料を見ながら、「実習」を行ってみる
- その後、各自、「課題」に挑戦する
  - 各自で自習 ＋ 巡回指導
  - 遠慮なく質問してください
- 自分のペースで先に進んで構いません

Literacy

DrScheme の使用

- DrScheme の起動
  - プログラム → PLT Scheme → DrScheme
- 今日の実習では「Intermediate Student」に設定
  - Language
  - → Choose Language
  - → Intermediate Student
  - → OK ボタン

Literacy

DrScheme の使用 / ステップ実行

- プログラム実行の振る舞いを観察するツール
- 「定義用ウインドウ」のみを使用
  (通常の実行と異なる)
- 「Intermediate Student」に設定する必要あり
  - Language → Choose Language
  - → Intermediate Student
  - → OK ボタン

Literacy

実習1. 条件関数のステップ実行

- 関数 easy-get-num-of-days (例題1) について、実行結果に至る過程を見る
  - (easy-get-num-of-days 1) から 31 に至る過程を見る
  - DrScheme のステップ実行機能を使用

```
(define (easy-get-num-of-days month)
  (cond
    [(= month 2) 28]
    [(or (= month 4)
         (= month 6)
         (= month 9)
         (= month 11)) 30]
    [else 31]))
```

Literacy

「実習1. 条件関数のステップ実行」の手順

1. 次を「定義用ウインドウ」で、実行しなさい
   - Intermediate Student で実行すること
   - 入力した後に、Run ボタンを押す

```
(define (easy-get-num-of-days month)
  (cond
    [(= month 2) 28]
    [(or (= month 4)
         (= month 6)
         (= month 9)
         (= month 11)) 30]
    [else 31]))
(easy-get-num-of-days 1)
```

例題1と同じ

ステップ実行したいので、定義用ウインドウだけを使う

2. DrScheme を使って、ステップ実行の様子を確認しなさい (Step ボタン、Next ボタンを使用)

Literacy
- 理解しながら進むこと

(easy-get-num-of-days 1) から 31 に至る過程 (1/3)

(easy-get-num-of-days 1) 最初の式

```
= (cond
    [(= 1 2) 28]
    [(or (= 1 4)
         (= 1 6)
         (= 1 9)
         (= 1 11)) 30]
    [else 31]))
```

(cond [(= month 2) 28] [(or (= month 4) (= month 6) (= month 9) (= month 11)) 30] [else 31])) に month = 1 が代入される

```
= (cond
    [false 28]
    [(or (= 1 4)
         (= 1 6)
         (= 1 9)
         (= 1 11)) 30]
    [else 31]))
```

(= 1 2) → false

```
= (cond
    [(or (= 1 4)
         (= 1 6)
         (= 1 9)
         (= 1 11)) 30]
    [else 31]))
```

[false 28] → 消える

コンピュータ内部での計算

(easy-get-num-of-days 1) から 31 に至る過程 (2/3)

```
= (cond
   [(or false
        (= 1 6)
        (= 1 9)
        (= 1 11)) 30]
   [else 31]))
= (cond
   [(or (= 1 9)
        (= 1 11)) 30]
   [else 31]))
= (cond
   [(or false
        (= 1 11)) 30]
   [else 31]))
= (cond
   [(or (= 1 11)) 30]
   [else 31]))
= (cond
   [(or false) 30]
   [else 31]))
```

コンピュータ内部での計算

(= 1 4) → false

false → 消える

(= 1 9) → false

false → 消える

(= 1 11) → false

(easy-get-num-of-days 1) から 31に至る過程 (3/3)

```
= (cond
   [false 30]
   [else 31]))
= 31
```

(or false) → false

コンピュータ内部での計算

出力結果

Literacy

実習2．条件関数のステップ実行

- 関数 is-leap? (例題2) について、実行結果に至る過程を見る
  - (is-leap? 2005) から false に至る過程を見る
  - DrScheme のステップ実行機能を使用

```
(define (is-leap? year)
  (cond
    [(or (= (remainder year 400) 0)
         (and (not (= (remainder year 100) 0))
              (= (remainder year 4) 0))) true]
    [else false]))
```

Literacy

「実習2．条件関数のステップ実行」の手順

1. 次を「定義用ウインドウ」で、実行しなさい
   - Intermediate Student で実行すること
   - 入力した後に、Run ボタンを押す

例題2と同じ

```
(define (is-leap? year)
  (cond
    [ (or  (= (remainder year 400) 0)
           (and (not (= (remainder year 100) 0))
                (= (remainder year 4) 0))) true]
    [else false]))
(is-leap? 2005)
```

ステップ実行したいので、定義用ウインドウだけを使う

2. DrScheme を使って、ステップ実行の様子を確認しなさい (Step ボタン、Next ボタンを使用)
   - 理解しながら進むこと

Literacy

(is-leap? 2005) から true が得られる過程 (1/3)

(is-leap? 2005) 最初の式

```
= (cond
    [(or (= (remainder 2005 400) 0)
         (and (not (= (remainder 2005 100) 0))
              (= (remainder 2005 4) 0))) true]
    [else false]))
= (cond
    [(or (= 5 0)
         (and (not (= (remainder 2005 100) 0))
              (= (remainder 2005 4) 0))) true]
    [else false]))
= (cond
    [(or false
         (and (not (= (remainder 2005 100) 0))
              (= (remainder 2005 4) 0))) true]
    [else false]))
```

(cond [(or (= (remainder year 400) 0) (and (not (= (remainder year 100) 0)) (= (remainder year 4) 0))) true] [else false]) に year = 2005 が代入される

(remainder 2005 400) → 5

(= 5 0) → false

コンピュータ内部での計算

(is-leap? 2005) から true が得られる過程 (2/3)

```
= (cond
    [(or (and (not (= (remainder 2005 100) 0))
              (= (remainder 2005 4) 0))) true]
    [else false]))
= (cond
    [(or (and (not (= 5 0))
              (= (remainder 2005 4) 0))) true]
    [else false]))
= (cond
    [(or (and (not false)
              (= (remainder 2005 4) 0))) true]
    [else false]))
= (cond
    [(or (and true
              (= (remainder 2005 4) 0))) true]
    [else false]))
```

(or false 式) → (or 式)

(remainder 2005 100) → 5

(= 5 0) → false

(not false) → true

コンピュータ内部での計算

(is-leap? 2005) から true が得られる過程 (3/3)

```
= (cond
    [(or (and (= (remainder 2005 4) 0))) true]
    [else false]))                          (and true 式) → (and 式)
= (cond
    [(or (and (= 1 0))) true]    (remainder 2005 4) → 1
    [else false]))
= (cond
    [(or (and false)) true]  (= 1 0) → false
    [else false]))
= (cond
    [(or false) true]
    [else false]))           (and false) → false
= (cond
    [false true]             (or false) → false
    [else false]))                          コンピュータ内部での計算
= false                                     実行結果
```

課題

Literacy

課題1

- x = 5 のとき、次の Scheme の式の実行結果は何であるか、答えなさい
  - 実際に DrScheme で実行しなさい
  - true あるいは false の値である

1. (> x 1)
2. (and (> 10 x) (> x 1))
3. (= x (* x x))

Literacy

課題2

- 次の値を求める関数 foo2 を定義し、実行結果を報告しなさい
  - 「X mod 7」は、X を 7 で割った時の余り (剰余)。例えば 2005 mod 7 は 3 である。
  - remainderを使いなさい。

$$[(20x + 8) / 7] \bmod 10$$

式中の [...] は、小数点以下切り捨てを意味する。

Literacy

課題3

- 関数 foo2 を定義し、実行結果を報告しなさい
  - cond を使うこと

```
50グラム以下なら    → 120
75グラム以下なら    → 140
100グラム以下なら   → 160
150グラム以下なら   → 200
```

Literacy

課題4. 月の日数

- 年 year (数値) と, 月 month (数値) を受け取り, その日数を求める関数 get-num-of-days を定義し, 実行結果を報告しなさい.
  - うるう年のことも考慮せよ

例えば, 2004年の2月は29日まである

  - 例題1の easy-get-num-of-days と 例題2の is-leap? を適切に利用する

Literacy

課題のヒント

- ここにあるのは「間違い」の例です。同じ間違いをしないこと。

1. 関数名の付け方の間違い

```
(define (decide price w)
    ... 以下省略
```

⇒ 「decide price」では無く、
「decide_price」のように1単語で書くこと

2. ≦、≧をプログラム中に使うことはできない

⇒ 代わりに、<=、>= を使うこと

Literacy