

sp-2. Scheme の式とプログラム

(Scheme プログラミング)

URL: <https://www.kkaneko.jp/cc/scheme/index.html>

金子邦彦



アウトライン



2-1 Scheme の式

2-2 Scheme の関数

2-3 パソコン演習

2-4 課題

2-1 Scheme の式

Scheme を使ってできること

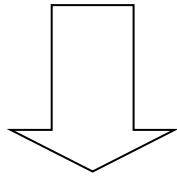


- 計算機能：
 - Scheme の式を入力すると：
計算が行われて、実行結果が表示される
- プログラム機能：
 - Scheme のプログラムを入力すると：
プログラムが記憶され、後で何度でも実行できる

Scheme の計算機能

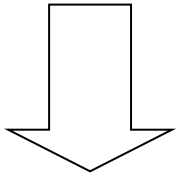


Scheme の式



Scheme の式を入力すると,

コンピュータ
(Scheme 搭載)



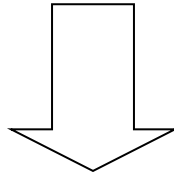
計算が行われて表示される

式の実行結果

Scheme のプログラム機能 (1/2)

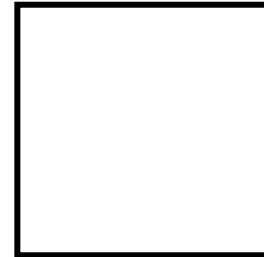
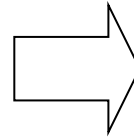


Scheme の
プログラム



Scheme のプログラムを
読み込ませると . . .

コンピュータ
(Scheme 搭載)

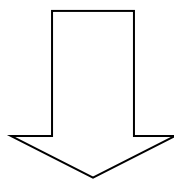


いったん, プログラムが
記憶される

Scheme のプログラム機能

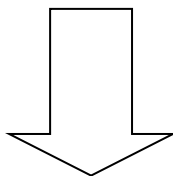
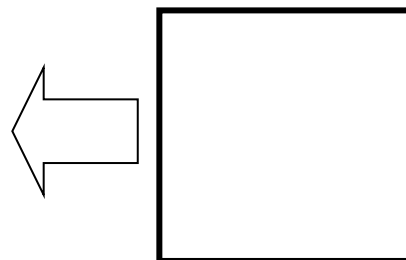


Scheme の式



今度は、読み込ませた
プログラムを実行させると・

コンピュータ
(Scheme 搭載)



記憶されていた
プログラムが使用される

式の実行結果

実行結果が表示される 7

Scheme の式に登場するもの



Scheme の式は, 以下の組み合わせ

- 数値 :
5, -5, 0.5 など
- 変数名
- 四則演算子 :
+, -, *, /
- その他の演算子 :
remainder, quotient, max,
min, abs, sqrt, expt, log, sin,
cos, tan
asin, acos, atan など
- 括弧
(,)
- 関数名
- define

これ以外にもあるが,
適宜授業で触れていく

Scheme の式の例

— 四則演算 —



• (+ 5 5)

• (+ -5 5)

• (+ 0.5 0.5)

• (- 5 5)

• (* 3 4)

• (/ 8 12)

} 負の数も扱える

} 負の数も扱える

} +, -, *, / が使える

「*」は掛け算
の意味

実行結果の例



```
> (+ 5 5)
10
> (+ -5 5)
0
> (+ 0.5 0.5)
1
> (- 5 5)
0
> (* 3 4)
12
> (/ 8 12)
2/3
>
```

式を入力すると、
計算が行われて表示される

15:3

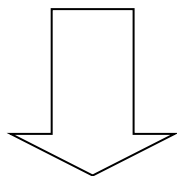
Unlocked

not running

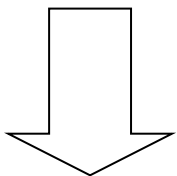
コンピュータが行っていること



Scheme の式



コンピュータ
(Scheme 搭載)



式の実行結果

例えば :

(+ 5 5)

を入力する
と . . .

10

が表示される

Scheme の式の例

— 各種の演算 —



- (remainder 100 15) ;; 100 を 15 で割った剰余(=10)
- (quotient 100 15) ;; 100 を 15 で割った商(=6)
- (max 3 5) ;; 3, 5 の大きい方 (=5)
- (min 3 5) ;; 3, 5 の小さい方 (=3)
- (abs -10) ;; -10 の絶対値 (=10)

実行結果の例



```
> (remainder 100 15)
```

```
10
```

```
> (quotient 100 15)
```

```
6
```

```
> (max 3 5)
```

```
5
```

```
> (min 3 5)
```

```
3
```

```
> (abs -10)
```

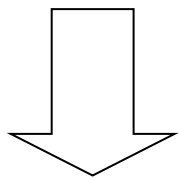
```
10
```

式を入力すると、
計算が行われて表示される

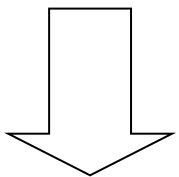
コンピュータが行っていること



Scheme の式



コンピュータ
(Scheme 搭載)



式の実行結果

例えば :

(remainder 100

15)

を入力すると . . .

10

が表示される

Scheme の式の例

— 各種の演算 —



- (sqrt 2) ;; 2 の平方根 $\sqrt{2}$
 - (expt 2 3) ;; 2 の 3 乗
 - (log 4) ;; $\log_e 4$ (eを底とする)
 - (sin 0.785) ;; $\sin 0.785$
 - (cos 0.785) ;; $\cos 0.785$
 - (tan 0.785) ;; $\tan 0.785$
 - (asin (/ (sqrt 2) 2)) ;; $\sin^{-1} \frac{\sqrt{2}}{2}$
 - (acos (/ (sqrt 2) 2)) ;; $\cos^{-1} \frac{\sqrt{2}}{2}$
 - (atan 1) ;; $\tan^{-1} 1$
- 三角関数の
単位はラジアン

実行結果の例



```
> (sqrt 2)
#i1.4142135623730951
> (expt 2 3)
8
> (log 4)
#i1.3862943611198906
> (sin 0.785)
#i0.706825181105366
> (cos 0.785)
#i0.7073882691671998
> (tan 0.785)
#i0.9992039901050427
> (asin (/ (sqrt 2) 2))
#i0.7853981633974484
> (acos (/ (sqrt 2) 2))
#i0.7853981633974483
> (atan 1)
#i0.7853981633974483
```

式を入力すると、
計算が行われて表示される

コンピュータが行っていること

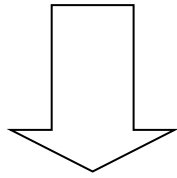


例えば：

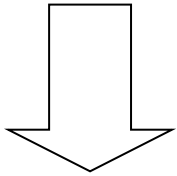
(sqrt 2)

を入力する
と・・・

Scheme の式



コンピュータ
(Scheme 搭載)



式の実行結果

#i1.4142135623730951

が表示される

#i とあるのは「近似値」という意味

Scheme の式の例

— 入れ子になった括弧 —

$$(2+2) * \frac{(3+5) * (30/10)}{2}$$

Scheme 言語で書くと： 

```
(* (+ 2 2)
  (/ (* (+ 3 5)
        (/ 30 10))
    2))
```

Scheme の式

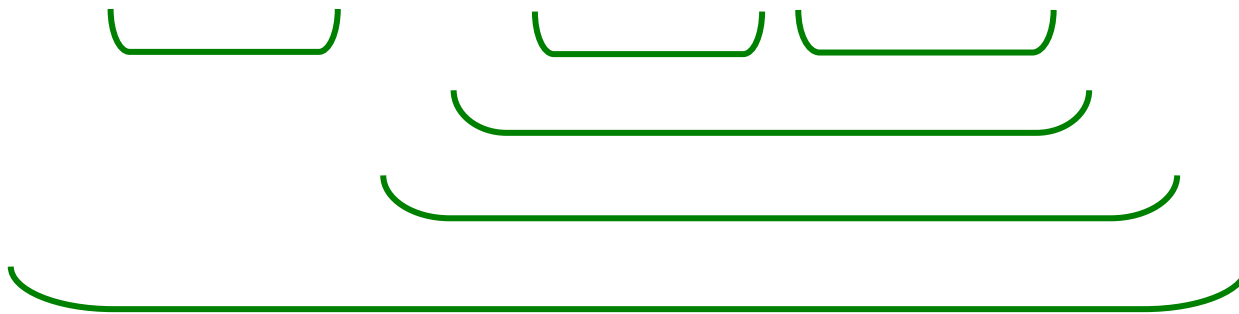
「*」 とあるのは、「乗算」の意味

括弧の意味



$$(2 + 2) * \frac{(3 + 5) * (30 / 10)}{2}$$

(* (+ 2 2) (/ (* (+ 3 5) (/ 30 10)) 2))



括弧が、計算の「単位」を表現

実行結果の例



Untitled (define ...)

Check Syntax Step Execute Break

式を入力すると

```
> (* (+ 2 2)
      (/ (* (+ 3 5)
            (/ 30 10))
        2))
```

48

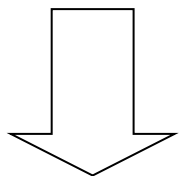
実行結果が表示される

8:3 Unlocked not running

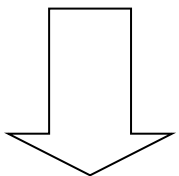
コンピュータが行っていること



Scheme の式



コンピュータ
(Scheme 搭載)



式の実行結果

例えば :

```
(* (+ 2 2)
  (/ (* (+ 3 5)
        (/ 30 10))
    2))
```

を入力する
と...

48

が表示される

2-2 Scheme の関数

プログラムとは



計算等の実行手順を記述したもの

例

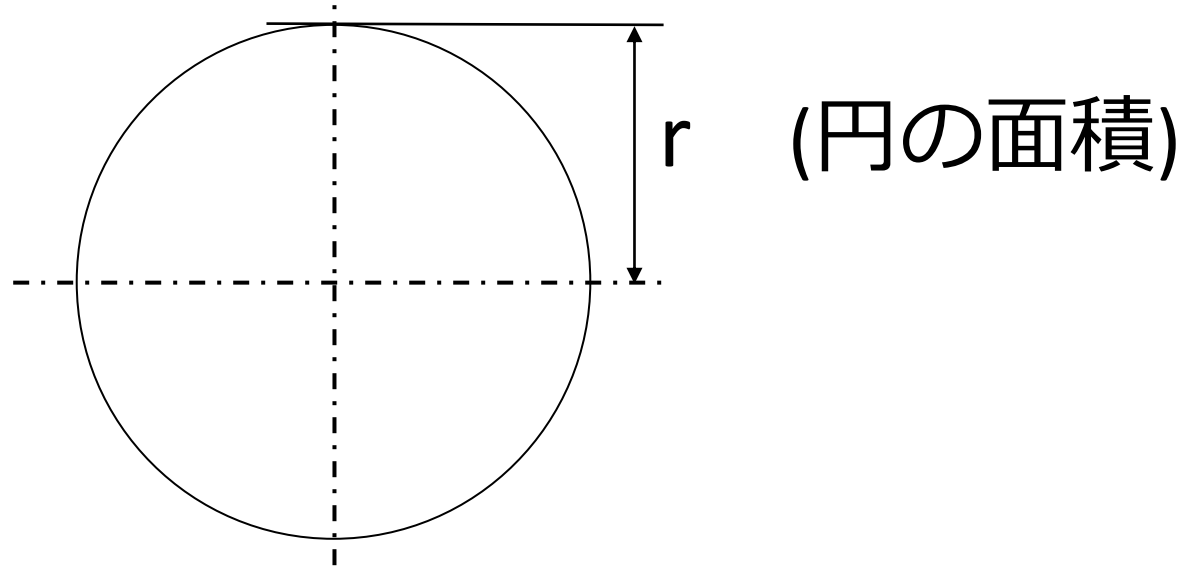
```
(define (area-of-disk r)
  (* 3.14
    (* r r)))
```

} Scheme のプログラム

円の面積を求めるプログラム

- 円の半径 r の値から, 円の面積
「 $(* 3.14 (* r r))$ 」を計算

円の面積



半径 r の円の面積は $3.14 r^2$

円の面積を求めるプログラム



```
(define (area-of-disk r)
  (* 3.14
    (* r r)))
```

この部分は Scheme の式
(変数 r を使用)

円の面積を求めるプログラム



```
(define (area-of-disk r)
  (* 3.14
    (* r r)))
```

値を1つ（名前はr）
受け取る

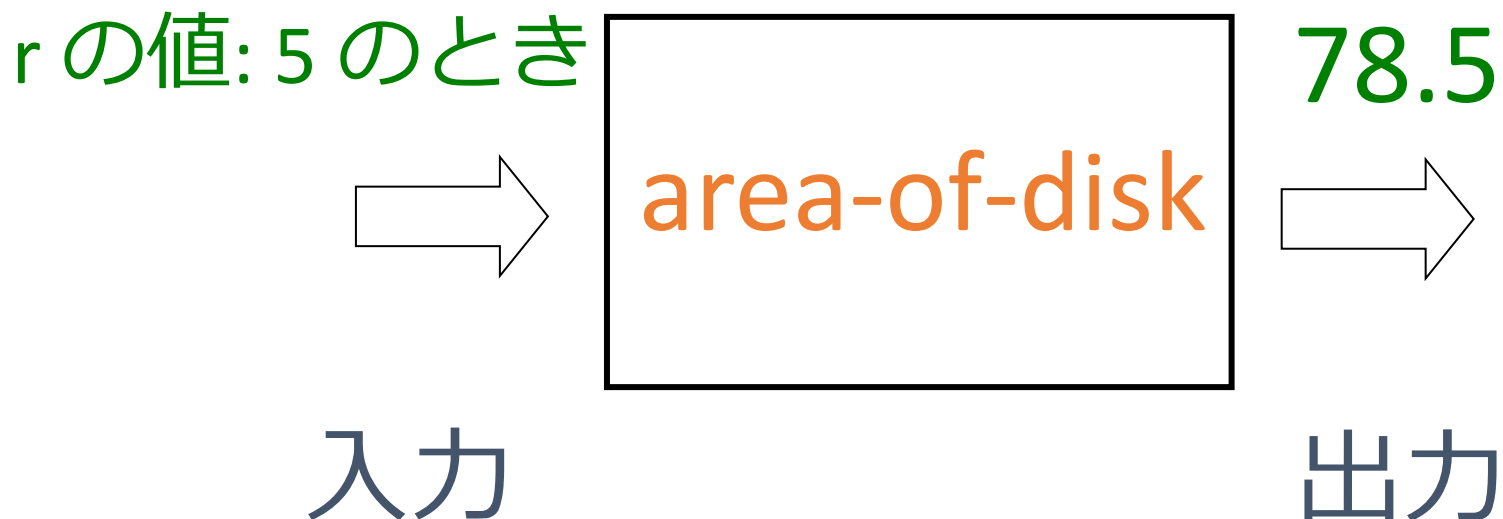
（rのことをパラメータ
という）

rの値から「(* 3.14 (* r r))」を計算

関数としてのプログラム



- プログラムは「関数」と見立てることができる
- 入力と出力がある



関数 = プログラムの単位

「関数である」ことを示すキーワード 関数の名前

```
(define (area-of-disk r)
  (* 3.14
    (* r r)))
```

1つの関数

rの値から「(* 3.14 (* r r))」を計算 (出力) 値を1つ受け取る (入力)

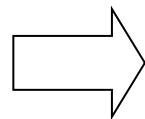
2-3 パソコン演習

- 資料を見ながら、「例題」を行ってみる
- 各自、「課題」に挑戦する
- 自分のペースで先に進んで構いません

Scheme プログラミングの手順



Scheme の関数を
作り, コンピュータ
に読み込ませる

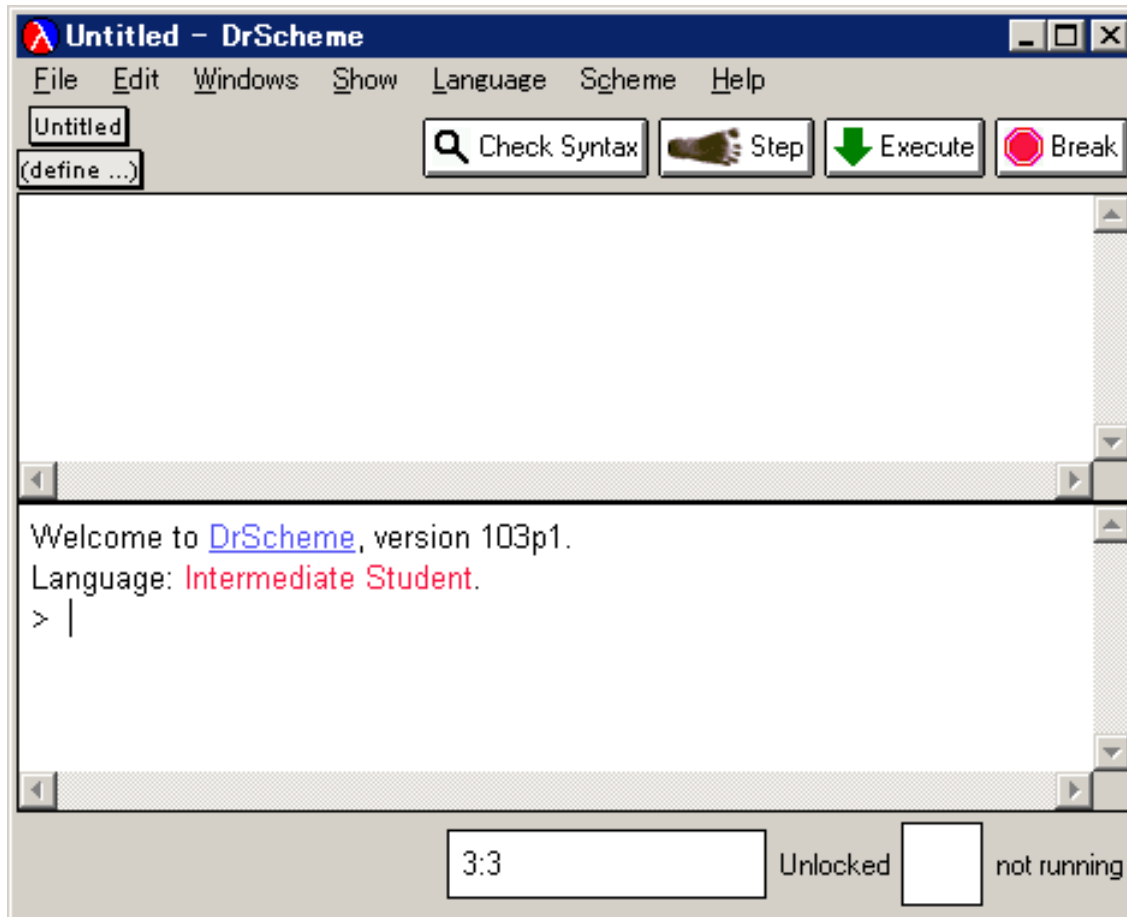


作った関数を実行
して, 実行結果を
得る

Scheme の関数の定義

Scheme の式の実行

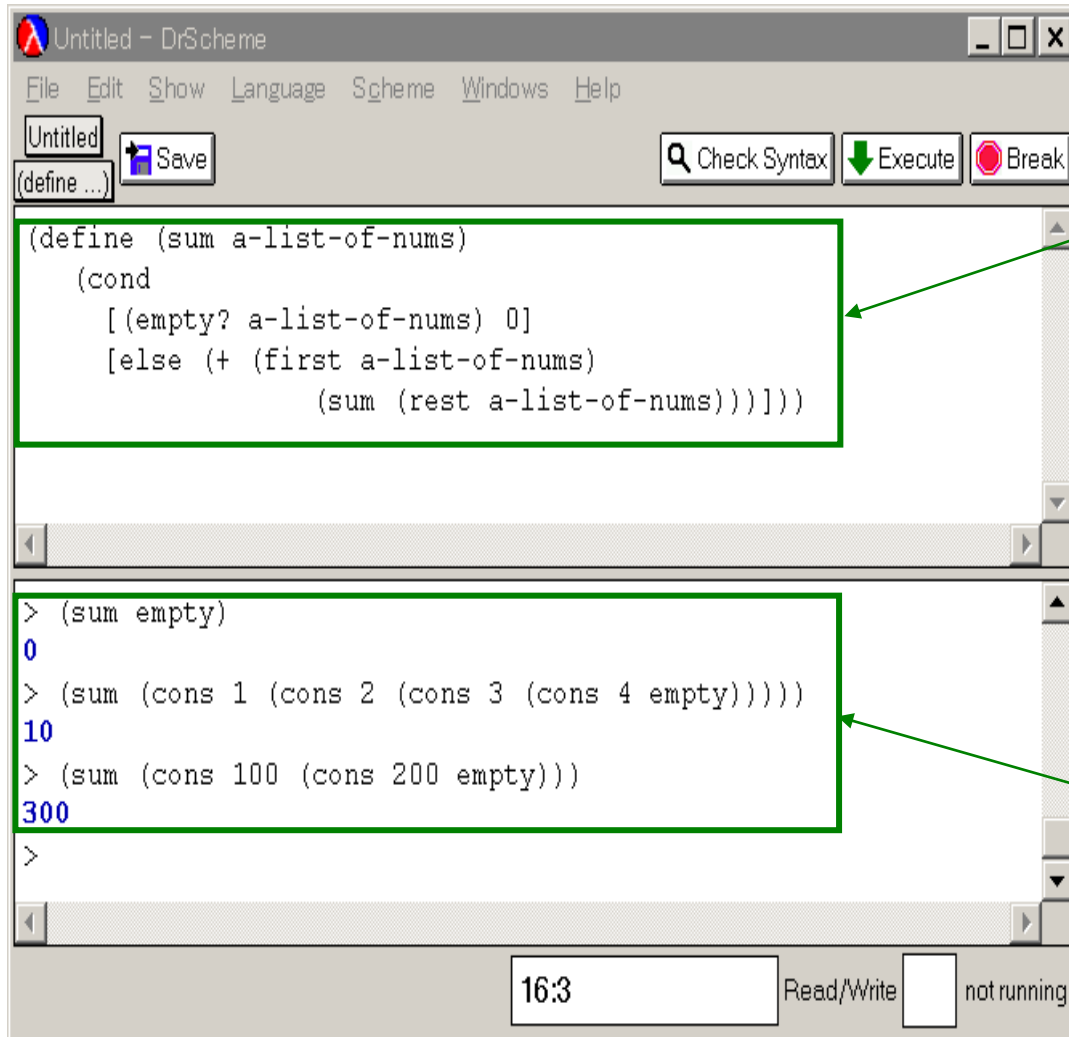
DrScheme の 2 つのウィンドウ



定義用ウィンドウ

実行用ウィンドウ

DrScheme の 2 つのウィンドウ



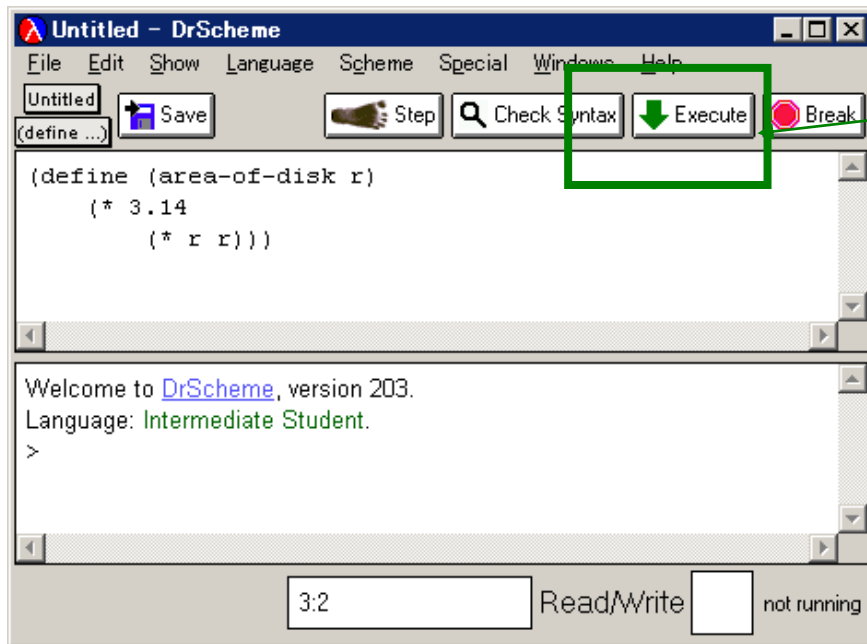
The screenshot shows the DrScheme IDE with two main windows. The top window is the editor, containing a Scheme function definition for a recursive sum function. The bottom window is the REPL, showing the results of executing several Scheme expressions. The status bar at the bottom indicates the current time is 16:3 and the system is in Read/Write mode, not running.

```
Untitled - DrScheme
File Edit Show Language Scheme Windows Help
Untitled Save Check Syntax Execute Break
(define (sum a-list-of-nums)
  (cond
    [(empty? a-list-of-nums) 0]
    [else (+ (first a-list-of-nums)
              (sum (rest a-list-of-nums)))]))
> (sum empty)
0
> (sum (cons 1 (cons 2 (cons 3 (cons 4 empty)))))
10
> (sum (cons 100 (cons 200 empty)))
300
>
16:3 Read/Write not running
```

関数を編集し、
コンピュータに
読み込ませている

読み込んだ関
数を実行させて
いる

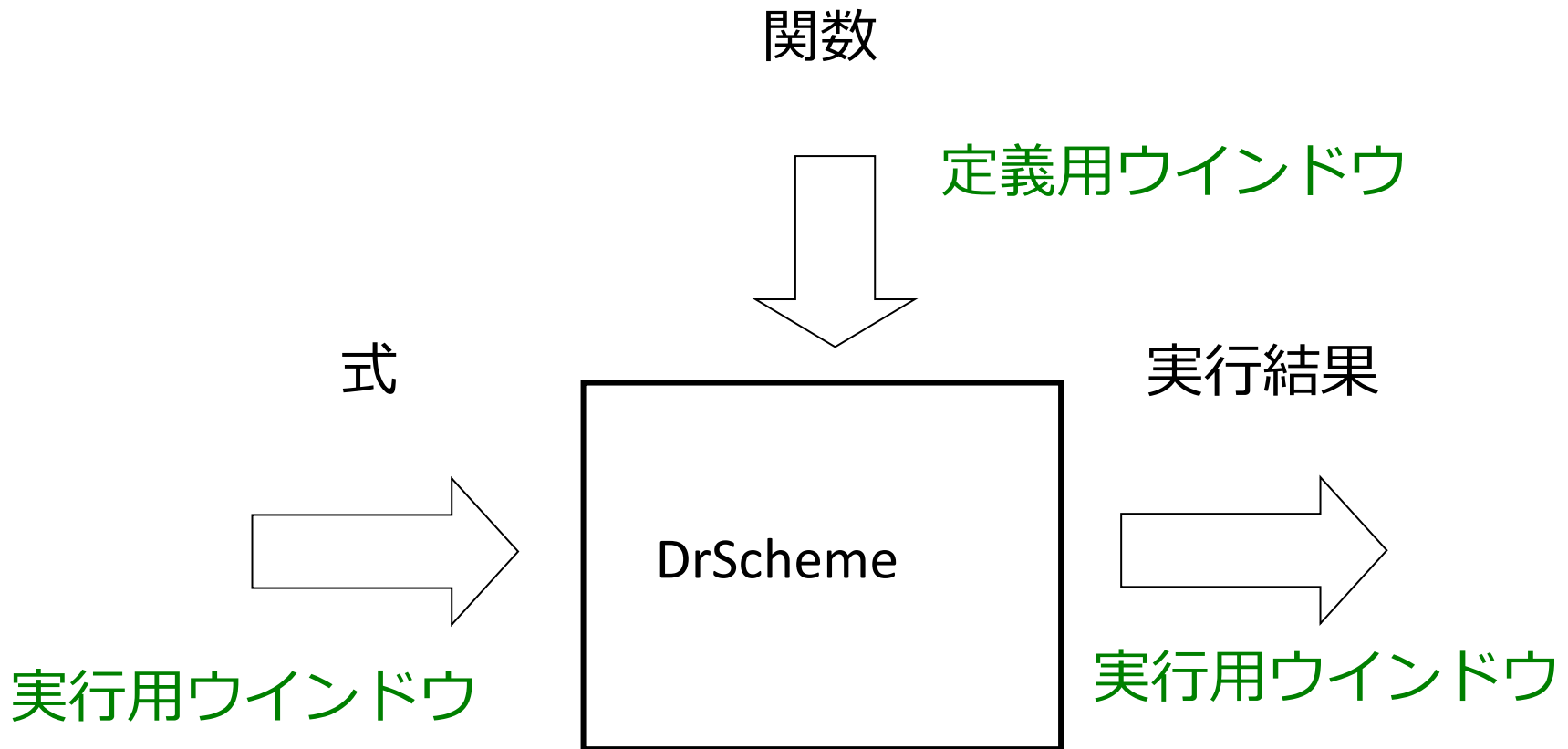
DrScheme の Execute ボタン



Execute ボタン

- 定義を行ったら
「Execute ボタン」を押す
- 関数の定義内容を
コンピュータが読み込む
- 「実行用ウィンドウ」の
中身はクリアされる

DrScheme の 2 つのウィンドウ



DrScheme でのプログラム保存法



- 何日かかけてプログラム作成したいとき
→ プログラムを保存する必要あり
- DrScheme の「Save 機能」を活用すること
 - ファイル名は「英語」で付けることを勧める



Untitled - DrScheme

File Edit Windows Show Language Scheme Help

- New (Ctrl+N)
- New Project
- Open... (Ctrl+O)
- Open Project...
- Open URL...
- Revert
- Save Definitions (Ctrl+S)**
- Save Definitions As...
- Save Other
- Print Definitions... (Ctrl+P)
- Print Interactions...
- Bring project to the front (Ctrl+J)
- Close (Ctrl+W)
- Exit (Ctrl+Q)

```
(year)
(remainder year 400) 0)
(not (= (remainder year 100) 0))
(= (remainder year 4) 0))) true]
```

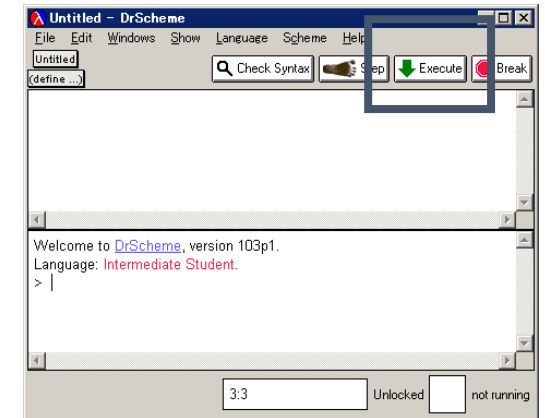
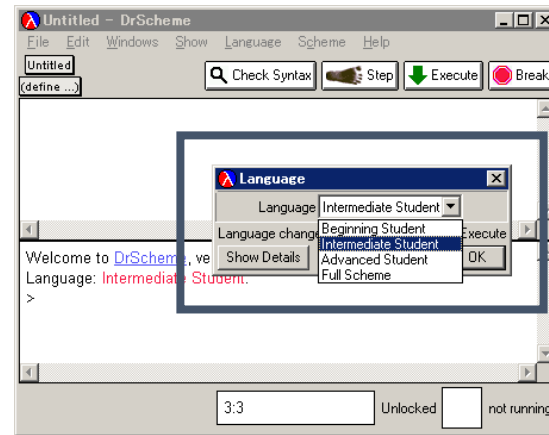
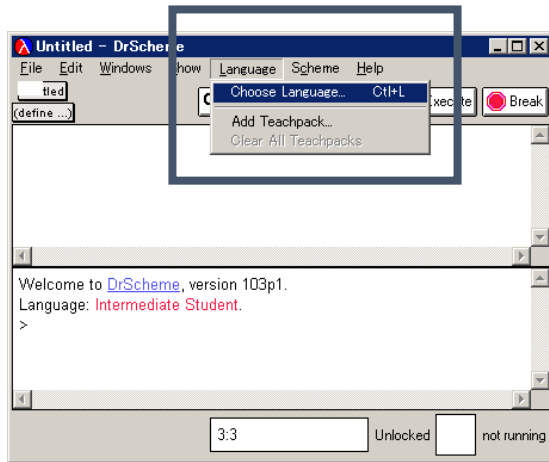
Language: Intermediate Student.

3:3 Unlocked not running

保存では、「File」→
「Save Definitions」を行う

- DrScheme の起動
プログラム → PLT Scheme →
DrScheme
- 今日のパソコン演習では「Intermediate Student」
に設定
Language
→ Choose Language
→ Intermediate Student
→ Execute ボタン

「Intermediate Student」に設定



Language
→ Choose Language

Intermediate Student
を選択し、
「OK」をクリック

最後に Execute ボタン

例題 1 . 簡単な数式



- 次の Scheme の式を DrScheme の実行用ウィンドウに入力し, 実行してみる

$$(2 + 2) * \frac{(3 + 5) * (30 / 10)}{2}$$

- Scheme 言語で書くと：

```
(* (+ 2 2)
  (/ (* (+ 3 5)
        (/ 30 10))
     2))
```

Scheme の式

「例題 1. 簡単な数式」の手順

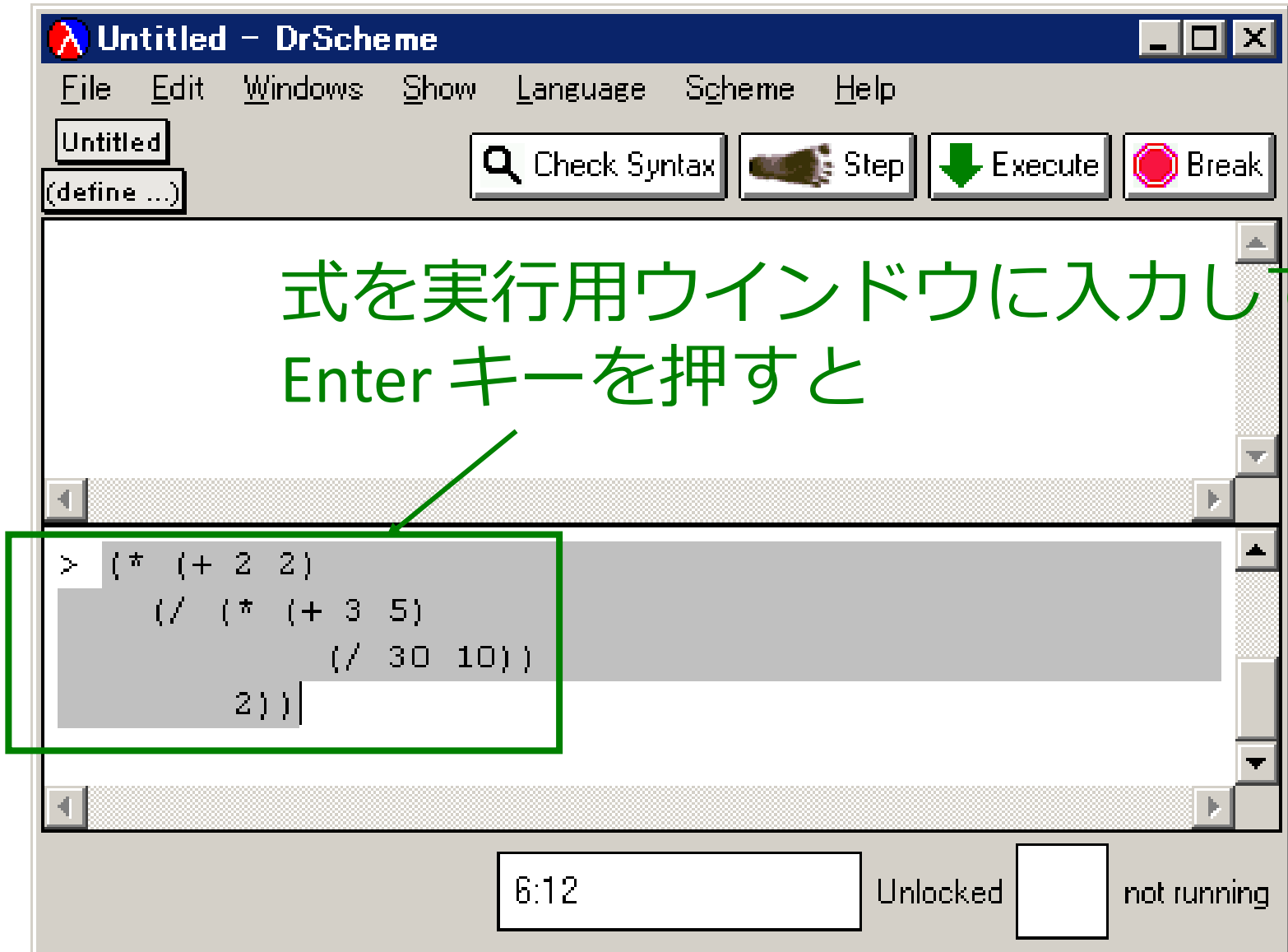


1. 次の式を「実行用ウィンドウ」で、実行しなさい

```
(* (+ 2 2)
 (/ (* (+ 3 5)
 (/ 30 10))
 2))
```

☆ 次は、例題 2 に進んでください

「例題 1. 簡単な数式」の結果 (1/2)

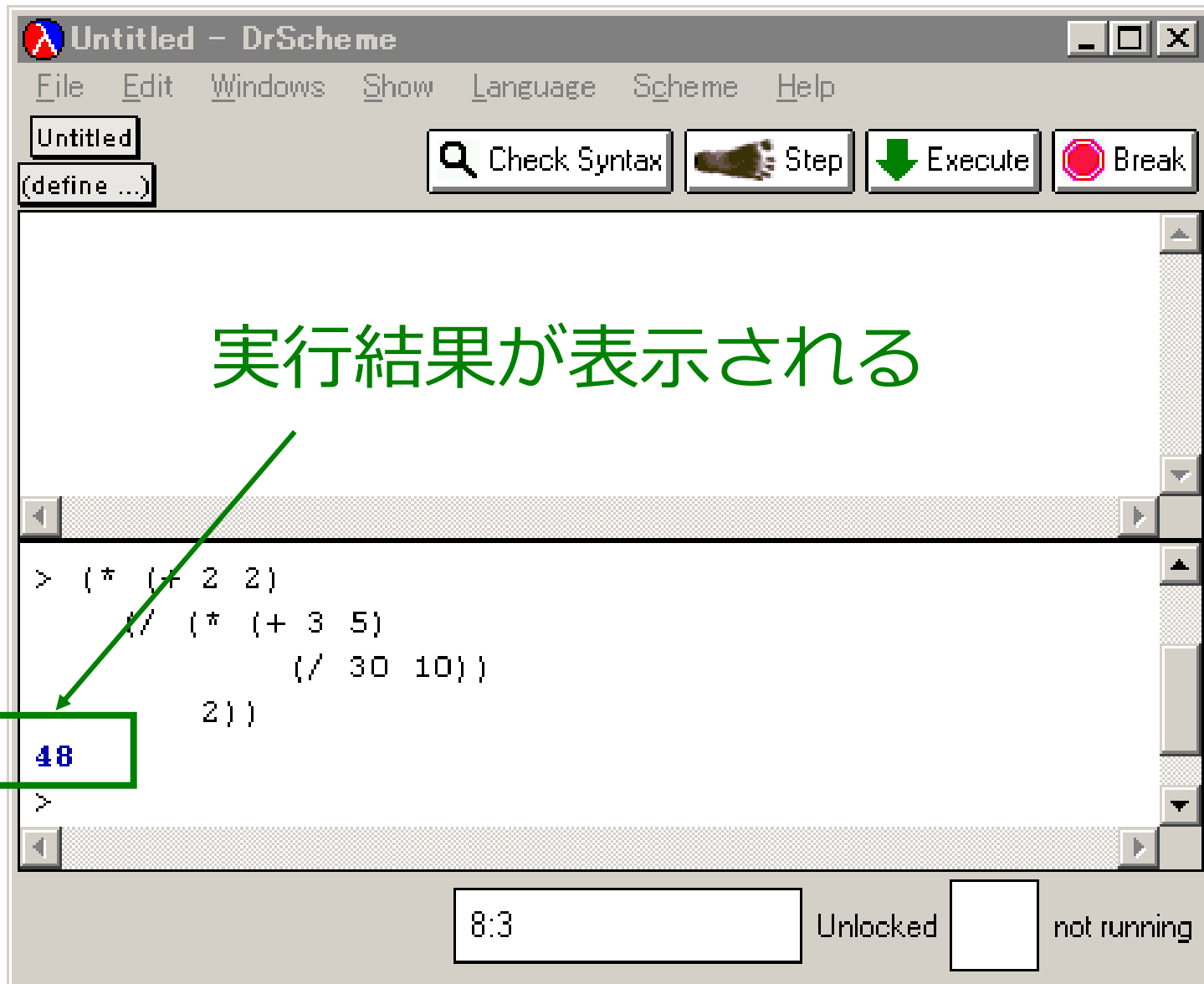


式を実行用ウィンドウに入力して,
Enter キーを押すと

```
> (* (+ 2 2)
      (/ (* (+ 3 5)
            (/ 30 10))
         2))
```

6:12 Unlocked not running

「例題 1. 簡単な数式」の結果 (2/2)



実行結果が表示される

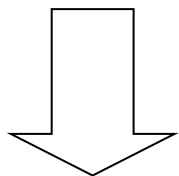
```
> (* (+ 2 2)
      (/ (* (+ 3 5)
            (/ 30 10))
         2))
48
>
```

8:3 Unlocked not running

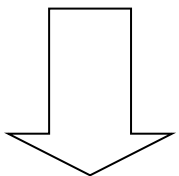
コンピュータが行っていること



Scheme の式



コンピュータ
(Scheme 搭載)



式の実行結果

例えば :

```
(* (+ 2 2)
  (/ (* (+ 3 5)
        (/ 30 10))
    2))
```

を入力する
と...



48

が表示される

よくある間違い



- 「スペース（空白文字）」に意味がある

間違いの例 1

```
(* (+2 2)
  (/ (* (+ 3 5)
        (/ 30 10))
    2))
```

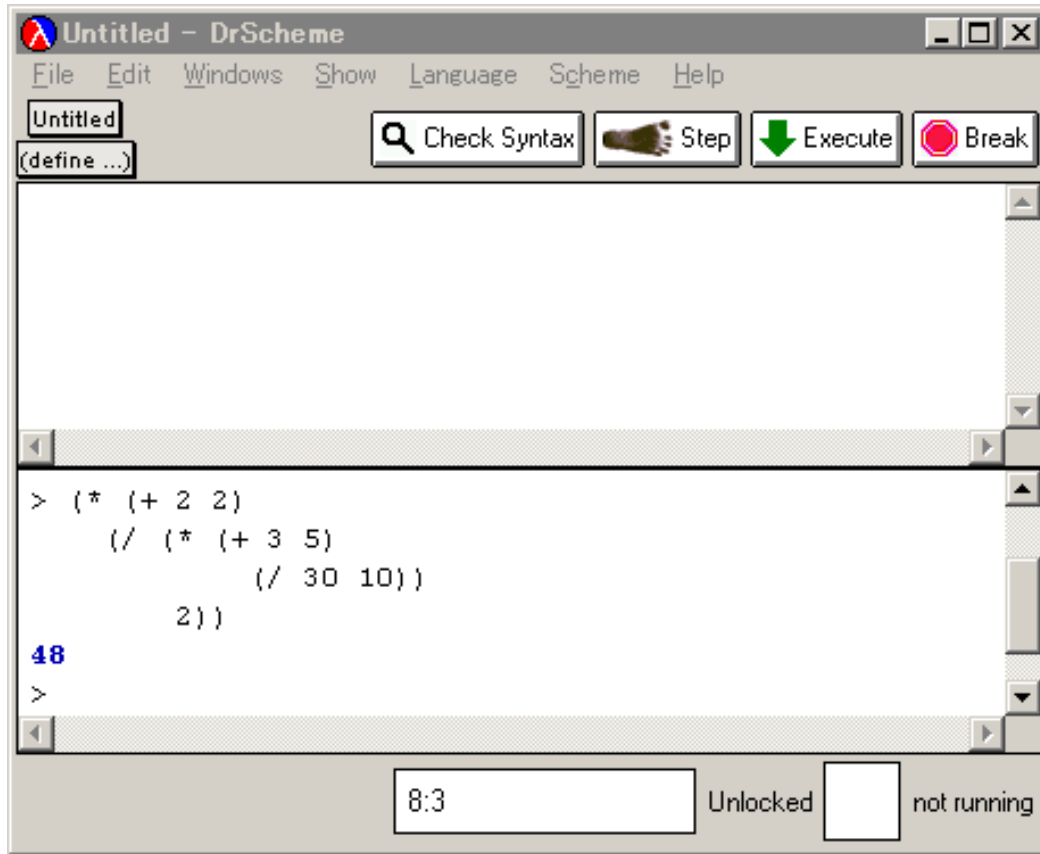
+の後にスペースが無い

間違いの例 2

```
(* (+ 2 2)
  (/ (* (+ 3 5)
        (/ 30 10))
    2))
```

*の後にスペースが無い

定義用ウィンドウ



簡単な数式の実行
では、定義用ウィンド
は使用しない

+ , - , * , / , sqrt , expt , remainder など
の基本的な演算は、すでに、コンピュータ内
に組み込み済み

例題 2 . 円の面積

- 円の半径 r から面積を求める関数 **area-of-disk** を書き, 実行する

例) $5 \rightarrow 78.5$

- 関数の名前 : **area-of-disk**
- パラメータ : r

「例題 2. 円の面積」の手順



1. 次を「定義用ウィンドウ」で，実行しなさい
 - 入力した後に，Execute ボタンを押す

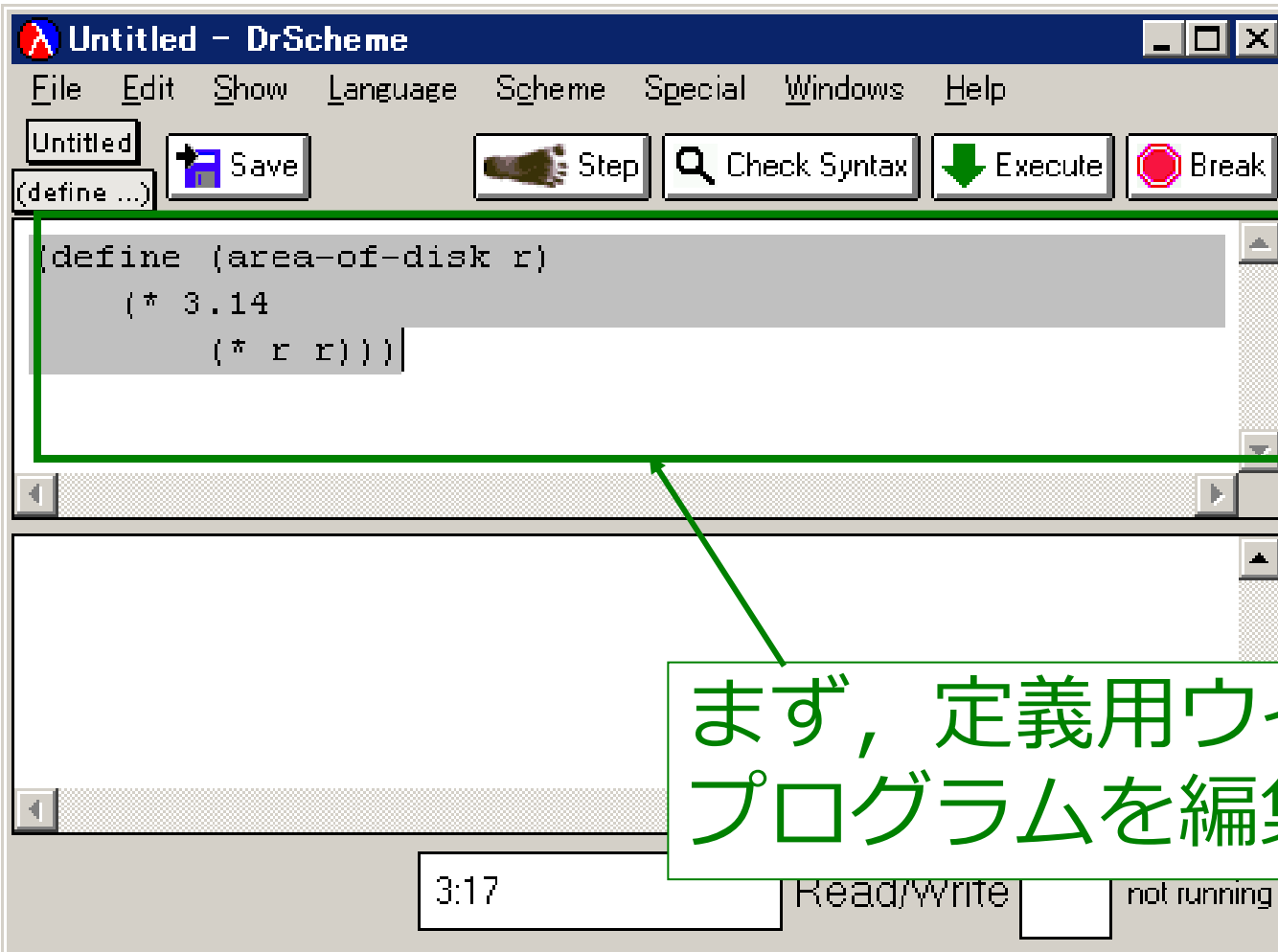
```
(define (area-of-disk r)
  (* 3.14
    (* r r)))
```

2. その後，次を「実行用ウィンドウ」で実行しなさい

```
(area-of-disk 5)
```

☆ 次は，例題 3 に進んでください

「例題 2. 円の面積」の結果(1/4)



The screenshot shows the DrScheme IDE window titled "Untitled - DrScheme". The menu bar includes File, Edit, Show, Language, Scheme, Special, Windows, and Help. The toolbar contains buttons for "Untitled", "Save", "Step", "Check Syntax", "Execute", and "Break". The code editor contains the following Scheme code:

```
(define (area-of-disk r)
  (* 3.14
     (* r r)))
```

The code editor is highlighted with a green border. A green arrow points from a text box to the code editor.

3:17 Read/Write not running

まず、定義用ウィンドウでプログラムを編集している

「例題 2 . 円の面積」の結果(2/4)



The screenshot shows the DrScheme IDE interface. The title bar reads "Untitled - DrScheme". The menu bar includes "File", "Edit", "Show", "Language", "Scheme", "Special", "Windows", and "Help". The toolbar contains buttons for "Save", "Step", "Check Syntax", "Execute", and "Break". The "Execute" button is highlighted with a green box. The code editor contains the following Scheme code:

```
(define (area-of-disk r)
  (* 3.14
    (* r r)))
```

The console window at the bottom shows the following text:

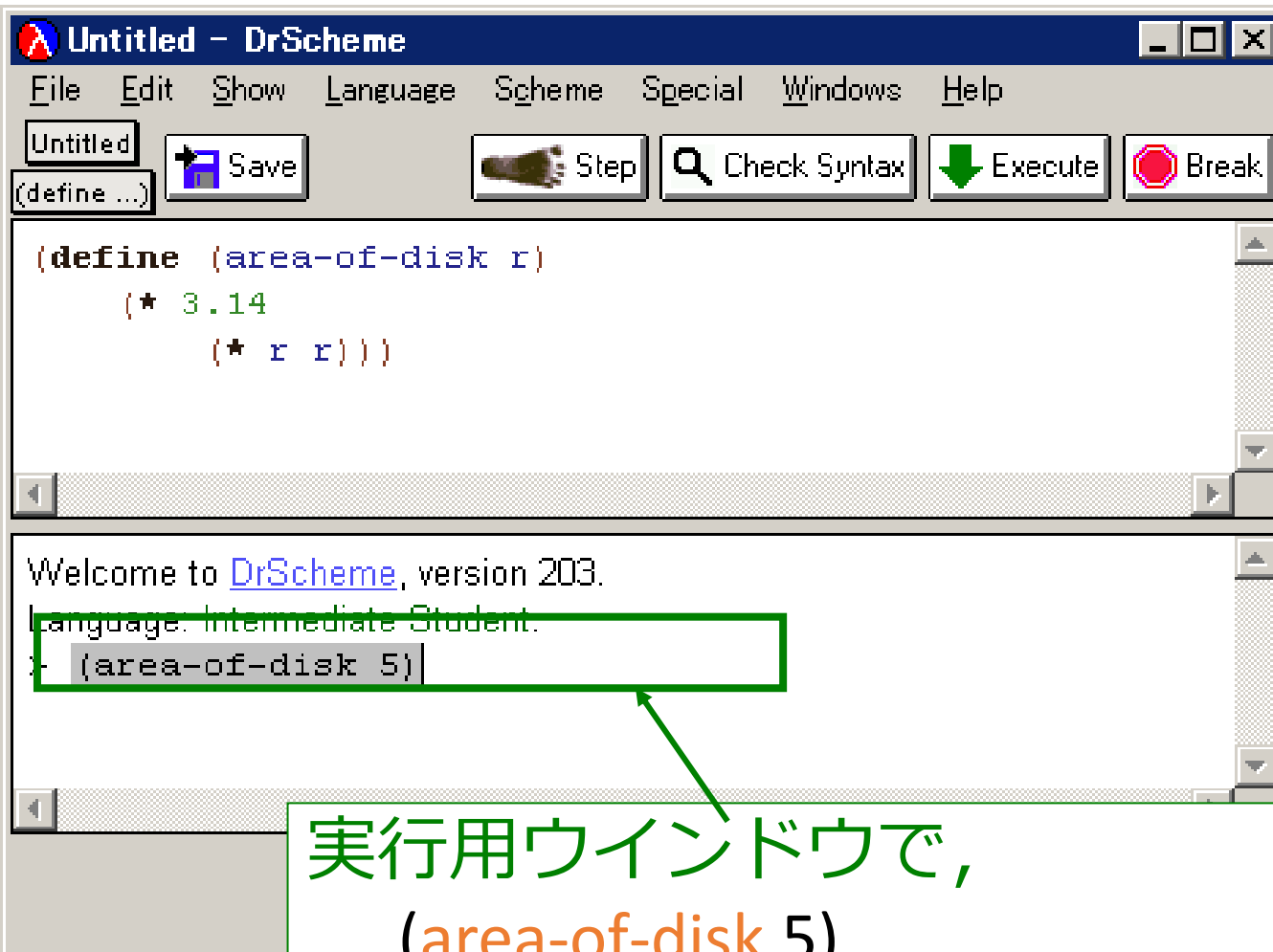
```
Welcome to DrScheme, version 203.
Language: Intermediate Student.
>
```

The status bar at the bottom shows "3:2" and "Read/Write".

Execute ボタンを押すと、定義用ウインドウに書いたプログラムが、コンピュータに読み込まれる

このとき、実行用ウインドウの中身はクリアされる

「例題 2. 円の面積」の結果(3/4)

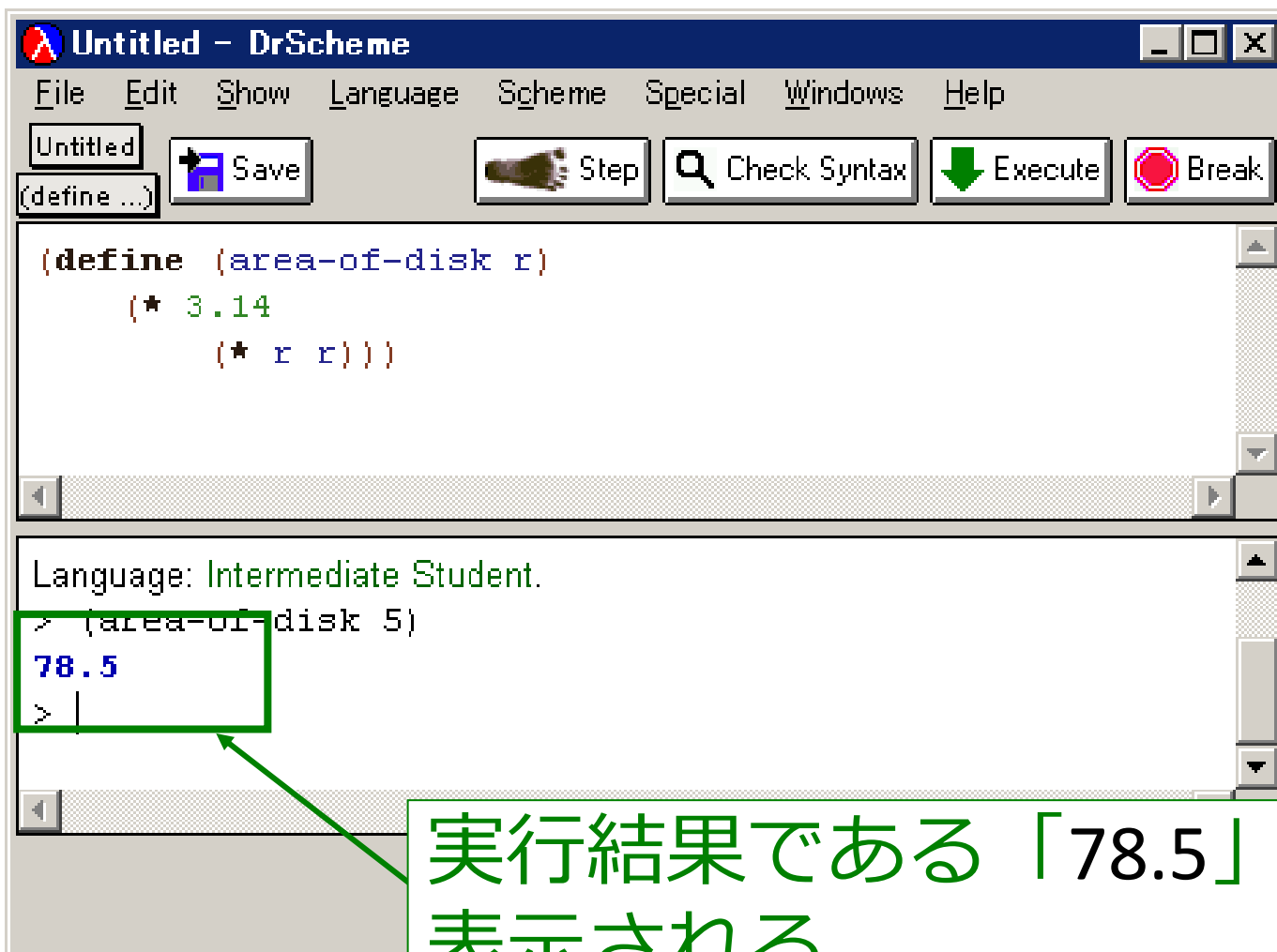


```
(define (area-of-disk r)
  (* 3.14
     (* r r)))
```

Welcome to [DrScheme](#), version 203.
language: Intermediate Student.
-> (area-of-disk 5)

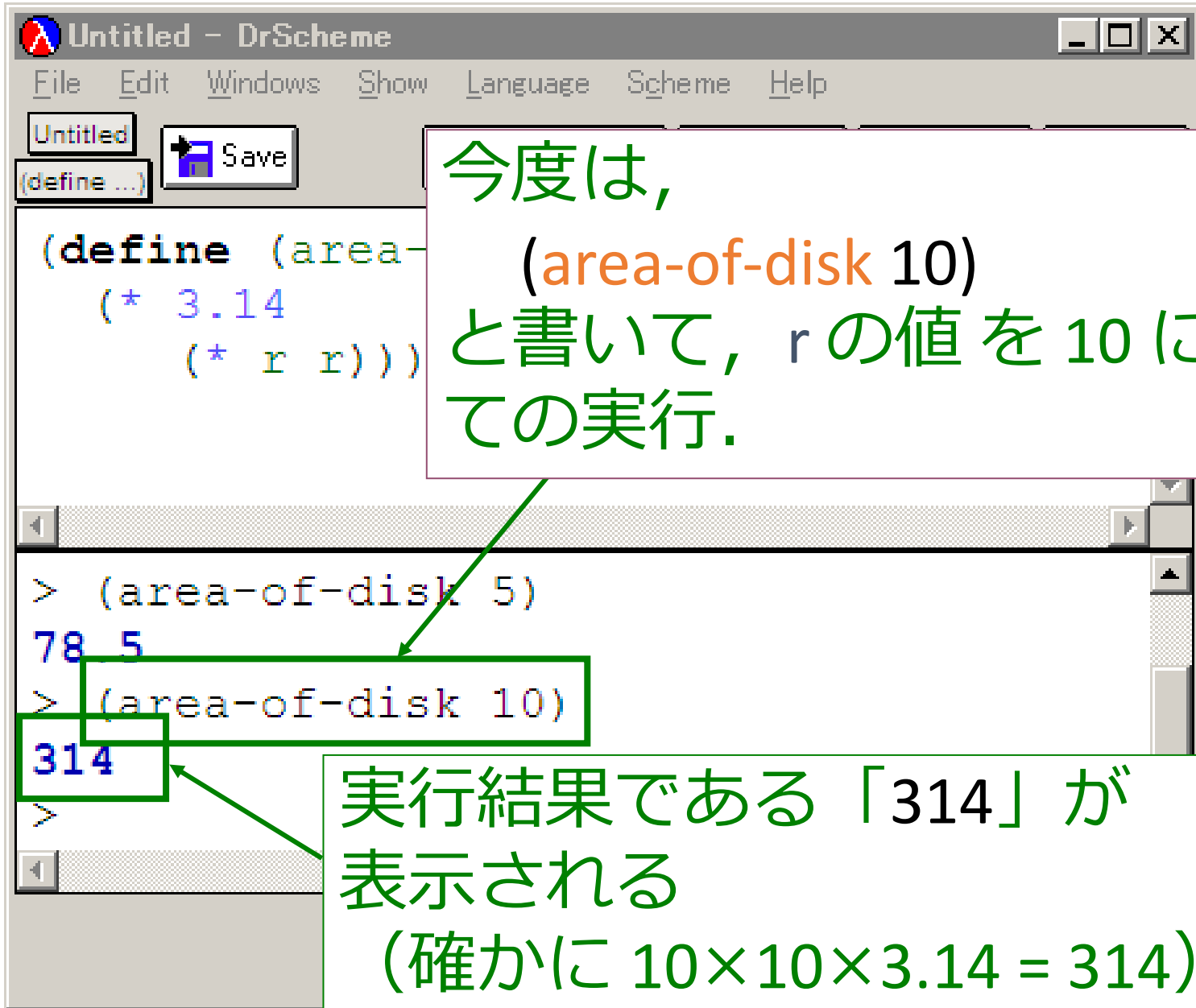
実行用ウィンドウで、
(area-of-disk 5)
と入力して、Enter キーを押す
(これは、r の値を 5 に設定しての実行)

「例題 2. 円の面積」の結果(4/4)



```
Untitled - DrScheme
File Edit Show Language Scheme Special Windows Help
Untitled Save Step Check Syntax Execute Break
(define ...)
(define (area-of-disk r)
  (* 3.14
     (* r r)))
Language: Intermediate Student.
> (area-of-disk 5)
78.5
> |
```

実行結果である「78.5」が表示される
(確かに $5 \times 5 \times 3.14 = 78.5$)



```
Untitled - DrScheme
File Edit Windows Show Language Scheme Help
Untitled Save
(define ...)
(define (area-of-disk
  (* 3.14
    (* r r)))
  )
> (area-of-disk 5)
78.5
> (area-of-disk 10)
314
>
```

今回は,

`(area-of-disk 10)`

と書いて, `r` の値を 10 に設定しての実行.

実行結果である「314」が表示される

(確かに $10 \times 10 \times 3.14 = 314$)

プログラム実行までの手順



- ① プログラムを書き, コンピュータに読み込ませる

例

```
(define (area-of-disk r)
  (* 3.14
    (* r r)))
```

Scheme のプログラム

円の面積を求める
プログラム

- ② 読み込ませたプログラムを実行させる

例

```
(area-of-disk 5)
```

```
(area-of-disk 10)
```

Scheme のプログラム
を実行させるための
Scheme の式

実際に, 半径 5, 半径 10
の円の面積を求める

プログラムの実行



- 式の中に「関数名」を書く

例：
式

(area-of-disk 5)

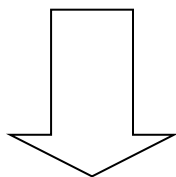
←これも、Scheme の

実際に、半径 5
の円の面積を求める

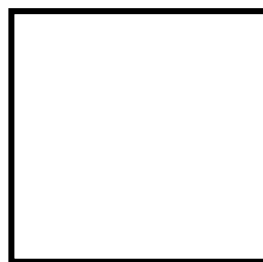
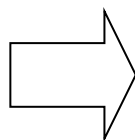
コンピュータが行っていること



Scheme の
プログラム (関数)



コンピュータ
(Scheme 搭載)



例えば

```
(define (area-of-disk r)
  (* 3.14
    (* r r)))
```

を読み込ませると

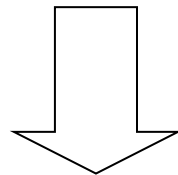
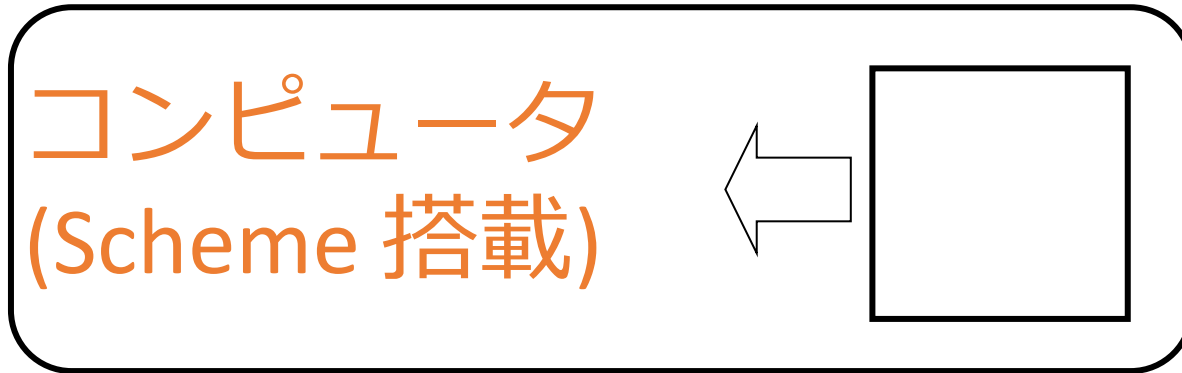
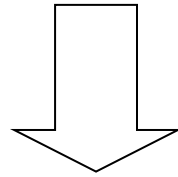


いったん,
コンピュータ内に
記憶される

コンピュータが行っていること



Scheme の式



式の実行結果

例えば :

(area-of-disk 5)

を入力する
と...



78.5

が表示される

例題 3. 簡単なプログラム



- 次の関数を書き, 実行する
 - f1: x と N から 「 x^N / N 」 を求める
 - f2: x と y から 「 x, y のうち大きいほう」 を求める
 - f3: x から 「 x を 100 で割った余り」 を求める
 - f4: x から 「 x を 100 で割った商」 を求める

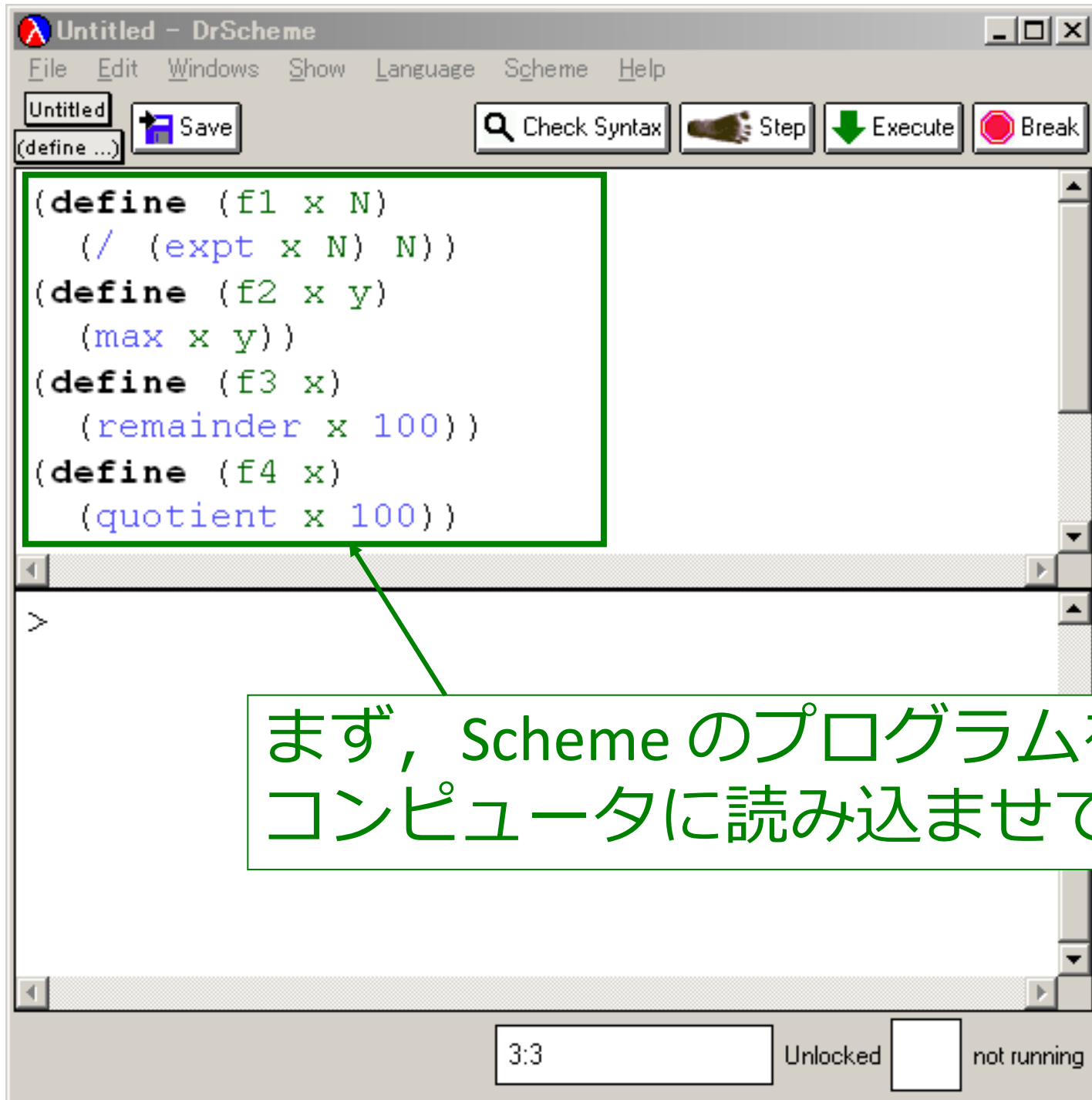
「例題 3. 簡単なプログラム」の手順

1. 次を「定義用ウィンドウ」で、実行しなさい
 - 入力した後に、Execute ボタンを押す

```
(define (f1 x N)
  (/ (expt x N) N))
(define (f2 x y)
  (max x y))
(define (f3 x)
  (remainder x 100))
(define (f4 x)
  (quotient x 100))
```

2. その後、次を「実行用ウィンドウ」で実行しなさい

```
(f1 2 5)
(f2 3 4)
(f3 123)
(f4 123)
```



The screenshot shows the DrScheme IDE window titled "Untitled - DrScheme". The menu bar includes "File", "Edit", "Windows", "Show", "Language", "Scheme", and "Help". The toolbar contains buttons for "Untitled", "Save", "Check Syntax", "Step", "Execute", and "Break". The main text area contains the following Scheme code:

```
(define (f1 x N)
  (/ (expt x N) N))
(define (f2 x y)
  (max x y))
(define (f3 x)
  (remainder x 100))
(define (f4 x)
  (quotient x 100))
```

The code is highlighted with a green box. Below the code area is a prompt character ">". A green arrow points from the text box below to the code area.

3:3 Unlocked not running

まず、Scheme のプログラムを
コンピュータに読み込ませている

Untitled - DrScheme

File Edit Windows Show Language Scheme Help

Untitled Save Check Syntax Step Execute Break

```
(define (f1 x N)
  (/ (expt x N) N))
(define (f2 x y)
  (max x y))
(define (f3 x)
  (remainder x 100))
(define (f4 x)
  (quotient x 100))
```

> (f1 2 5)
6.4

> (f2 3 4)
4

> (f3 123)
23

> (f4 123)
1

11:3 Unlocked not running

確かに $2^5/5 = 6.4$

確かに, 3, 4 の大きい方は 4

確かに 123 を 100 で割った余りは, 23

確かに 123 を 100 で割った商は, 1

Scheme の式の例 変数が登場するもの



関数の本体には「変数を含む式」を書くことになる

- x^N / N
(/ (expt x N) N) . . . 2 変数の式
- x, y のうち大きいほう
(max x y) . . . 2 変数の式
- x を 100 で割った余り
(remainder x 100) . . . 1 変数の式
- x を 100 で割った商
(quotient x 100) . . . 1 変数の式

2-4 課題

課題 1. ドルから円への変換



- ドル d から円を求める関数 $d2y$ を作成し, 実行結果を報告しなさい
 - `define` を使うこと
 - 1ドルは, 120.53円とする

課題のヒント



- x から「 $10x + 30$ 」を求める関数 **foo** を作成し、実行結果を報告しなさい

解答の例：

```
(define (foo x)
  (+ (* 10 x) 30))
```

実行結果は次の通り。期待通りの結果を得た

```
> (foo 10)
130
> (foo 20)
230
```

(あくまでも解等の例です)

課題 1 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. 「かっこ」の間違い

```
define (d2y d)
  (* 120.53 dollar)
```

⇒ 全体をかっこで囲むこと

3. 関数の書き方の間違い

```
(define (d2y)
  (* 120.53 d))
```

⇒ d2y の後に d が必要

2. 変数名の対応の間違い

```
(define (d2y dollar)
  (* 120.53 d))
```

⇒ 変数名 d と dollar は
どちらか 1 つにそろえること

4. 関数名の付け方の間違い

```
(define (d 2 y d)
  (* 120.53 d))
```

⇒ 「d 2 y」では無く,
「d2y」と書くこと

課題 2. 摂氏から華氏への変換



- 摂氏 (Celsius) c から華氏 (Fahrenheit) を求める関数 $c2f$ を作成し, 実行結果を報告しなさい
 - `define` を使うこと
 - 摂氏と華氏の変換式 : $c = 5 \times (f - 32) / 9$

課題 3 . 元利の計算



- 元利を求める関数 **interest** を作成し, 実行結果を報告しなさい
 - define を使うこと
 - 元利の計算式:
$$\text{「元利} = \text{元金} \times (1 + \text{年利})^{\text{年数}}\text{」}$$
 - 作成した関数を実行し, 元金 1 0 0 0 円, 年利 2 % での, 5 0 年後の元利を報告しなさい

課題 4 . Scheme 式



- 次の計算を行う Scheme の式を書き、「DrScheme の実行用ウィンドウ」で実行して、実行結果を報告しなさい

$5 + 5$

$-5 + 5$

$3 * 4$

$8 / 12$

$(2 + 2) * (((3 + 5) * (30 / 10)) / 2)$

$3 + 4.5$

2の平方根

3の5乗

356を4で割った余り

7の対数 (但し, e を底とする)

$\sin (0.7865)$ (0.7865 はラジアン)