sp-17. フィボナッチ数

（Scheme プログラミング）

URL: https://www.kkaneko.jp/pro/scheme/index.html

金子邦彦

アウトライン

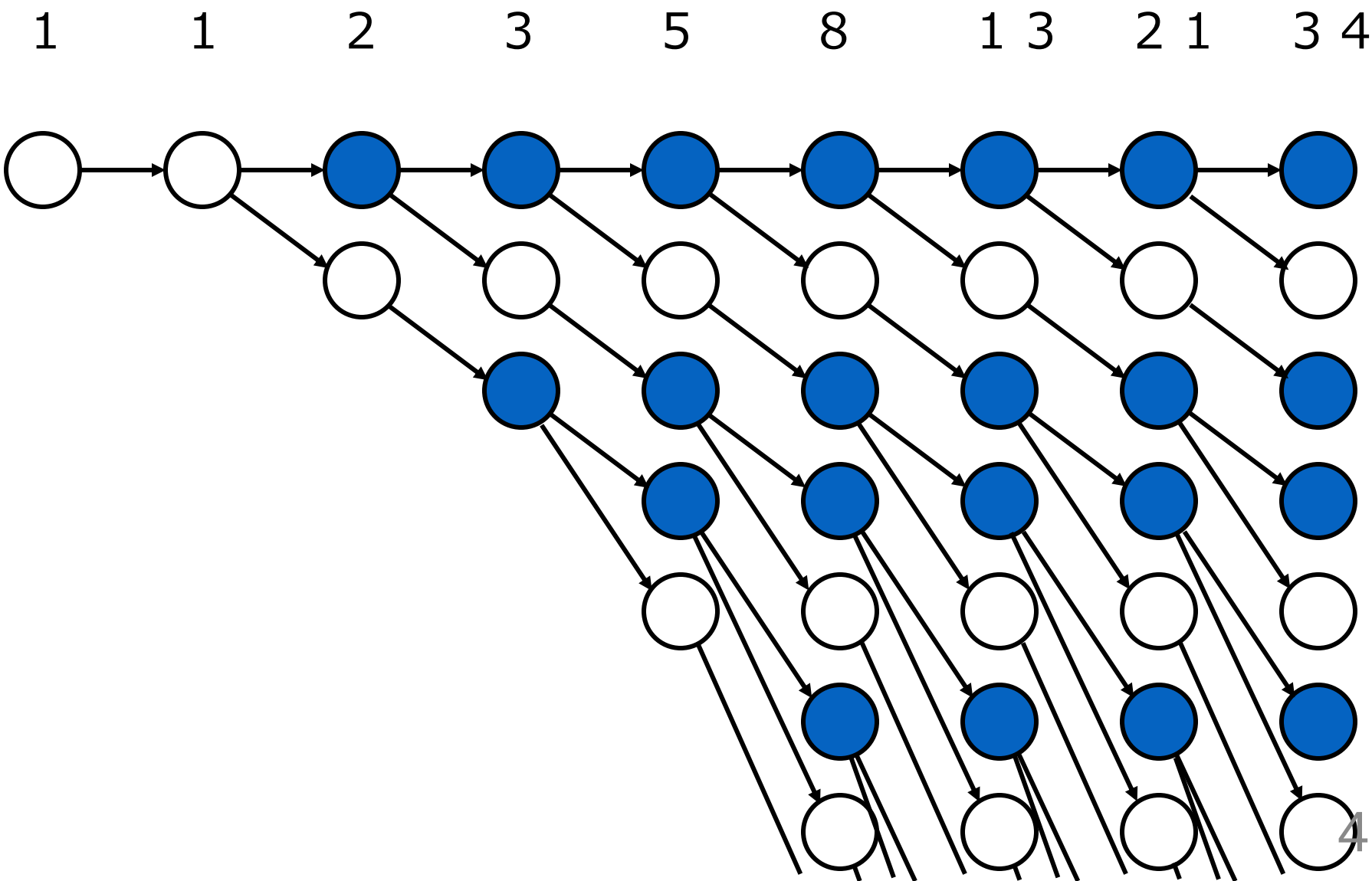
17-1 フィボナッチ数

17-2 パソコン演習

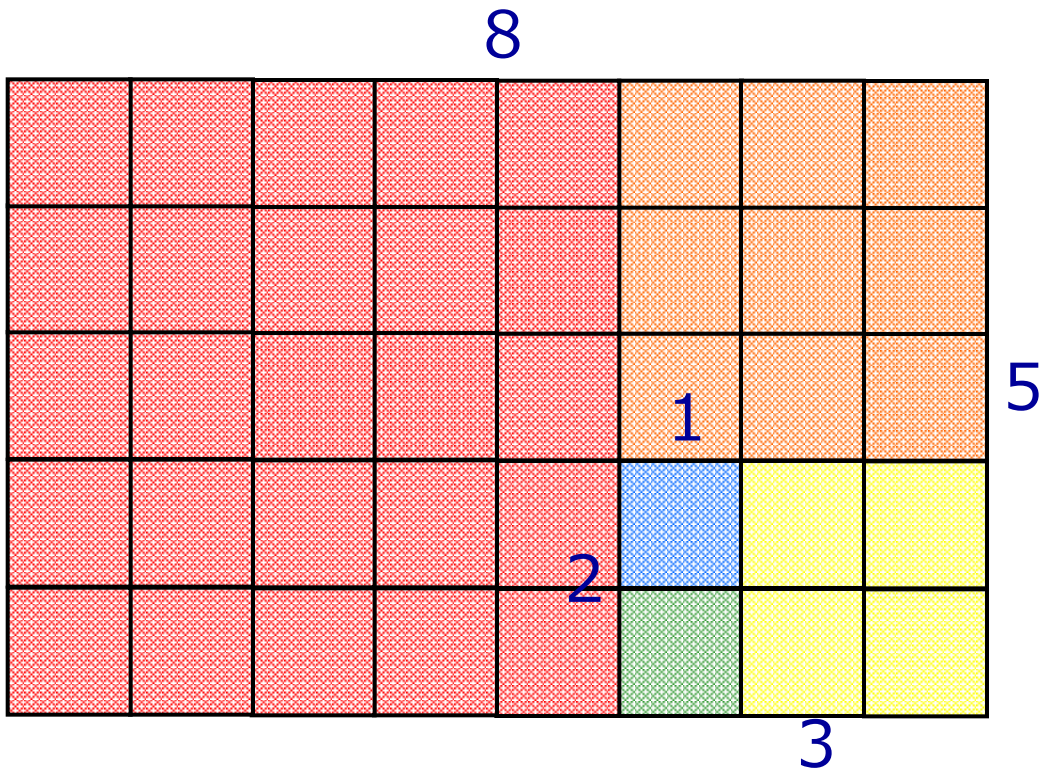
17-3 課題

17-1 フィボナッチ数

フィボナッチ数



フィボナッチ数



フィボナッチ数

- 生成的再帰（木構造再帰プロセス）の形で，フィボナッチ数

$$f_0 = 0,$$
$$f_1 = 1,$$
$$f_n = f_{n-1} + f_{n-2} (n > 1)$$

の i 番めの数 f_i を計算するプログラムを作る

例） 0,1,1,2,3,5,8,13,21,34,55,89,144,....

フィボナッチ数

$$f_0 = 0,$$

$$f_1 = 1,$$

$$f_n = f_{n-1} + f_{n-2} (n > 1)$$

17-2 パソコン演習

パソコン演習の進め方

- 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

DrScheme の使用

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

例題１．フィボナッチ数

- 生成的再帰（木構造再帰プロセス）の形で，フィボナッチ数

$$f_0 = 0,$$
$$f_1 = 1,$$
$$f_n = f_{n-1} + f_{n-2} \quad (n > 1)$$

の i 番めの数 f_i を計算するプログラムを作る

例） 0,1,1,2,3,5,8,13,21,34,55,89,144,....

「例題１．フィボナッチ数」の手順

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

```
(define (fibo n)
   (cond
      [(= n 0) 0]
      [(= n 1) 1]
      [else (+ (fibo (- n 1)) (fibo (- n 2)))]))
```

2. その後，次を「実行用ウインドウ」で実行しなさい

```
(fibo 5)
(fibo 6)
(fibo 7)
```

☆　次は，例題２に進んでください

実行結果の例

Untitled - DrScheme

File Edit Windows Show Language Scheme Help

Untitled (define ...) Save Check Syntax Step Execute Break

```
(define (fibo n)
    (cond
        [(= n 0) 0]
        [(= n 1) 1]
        [else (+ (fibo (- n 1)) (fibo (- n 2)))]))
```

```
> (fibo 5)
5
> (fibo 6)
8
> (fibo 7)
13
```

15:3 Unlocked not running

入力と出力

入力は数値

出力は数値

フィボナッチ数


```
(define (fibo n)
  (cond
    [(= n 0) 0]
    [(= n 1) 1]
    [else (+ (fibo (- n 1)) (fibo (- n 2)))]))
```

フィボナッチ数

1. n = 0 ならば：　　→　終了条件

   0　　→　自明な解

2. n = 1 ならば：　　→　終了条件

   1　　→　自明な解

3. そうで無ければ：

   – (fibo (- n 2)) と (fibo (- n 1)) を足す

     ⇒ 結局，fibo の実行を繰り返す

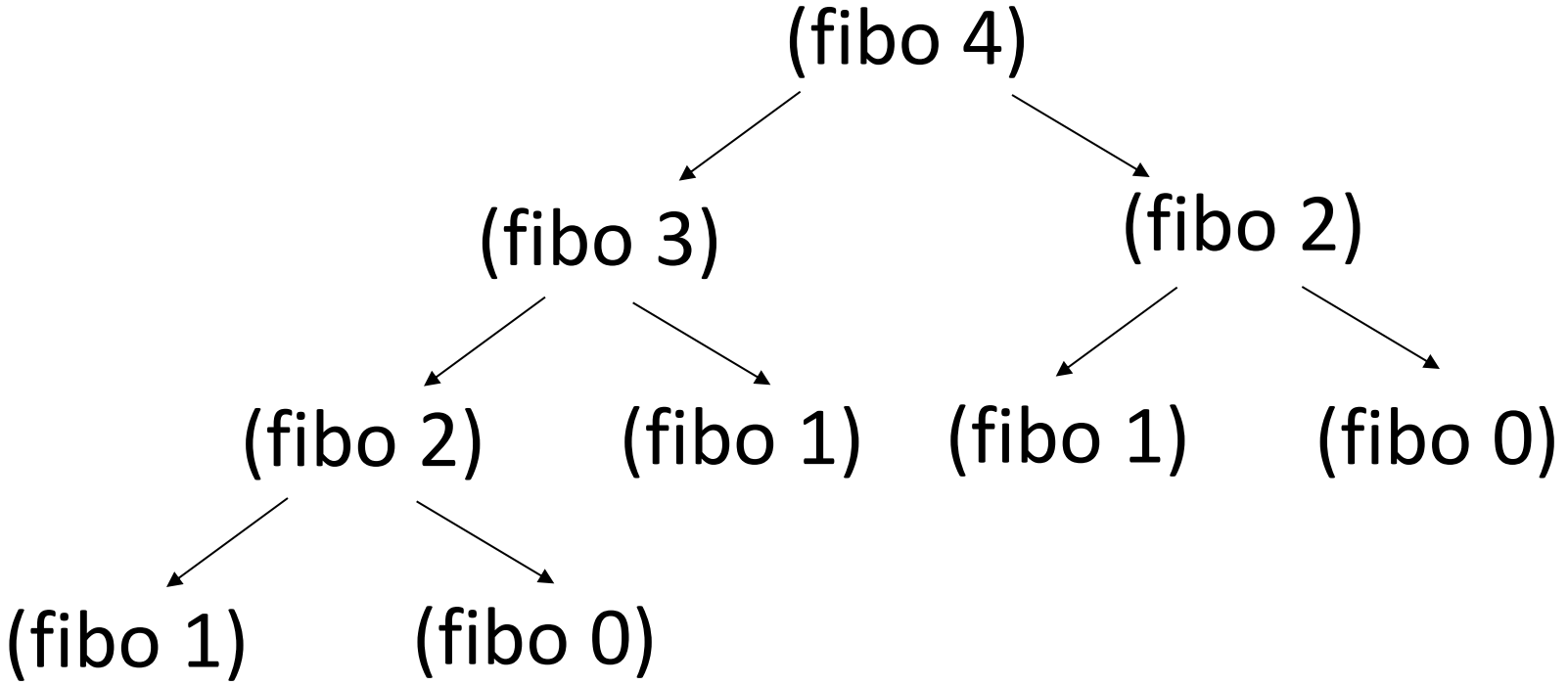
(fibo 4)から 3 が得られる過程の概略 (1/2)

```
(fibo 4)
= …
= (+ (fibo (- 4 1))
     (fibo (- 4 2)))
= …
= (+ (+
       (fibo (- 3 1))
       (fibo (- 3 2)))
     (fibo (- 4 2)))
= …
= (+ (+ (+
           (fibo (- 2 1))
           (fibo (- 2 2)))
         (fibo (- 3 2)))
       (fibo (- 4 2)))
= …
= (+ (+ (+
           1
           0)
         (fibo (- 3 2)))
       (fibo (- 4 2)))
```

(fibo 4)から 3 が得られる過程の概略 (2/2)

```
= …
= (+ (+
        1
        1)
     (fibo (- 4 2)))
= …
= (+ 2
     (+
        (fibo (- 2 1))
        (fibo (- 2
 2))))
```

```
= …
= (+ 2
     (+ 1 0))
= …
= 3
```



fibo の計算パターンは、木構造再帰である

フィボナッチ数

- 関数 fibo は，n > 1 のとき，fibo を2回呼び出す

  例）(fibo 10) を計算するために (fibo 9) と (fibo 8) を計算している

- 計算パターンは，木構造再帰(tree recursion)をなす
  - 樹木状をなす　（前ページ）

例題２．「反復的プロセス」でのフィボナッチ数

- フィボナッチ数

$$f_0 = 0,$$
$$f_1 = 1,$$
$$f_n = f_{n-1} + f_{n-2} (n > 1)$$

の i 番めの数 f_i を計算するプログラムを作る

- 例題１よりも繰り返し回数が少なくなるように工夫する

「例題２．反復的プロセスでのフィボナッチ数」の手順

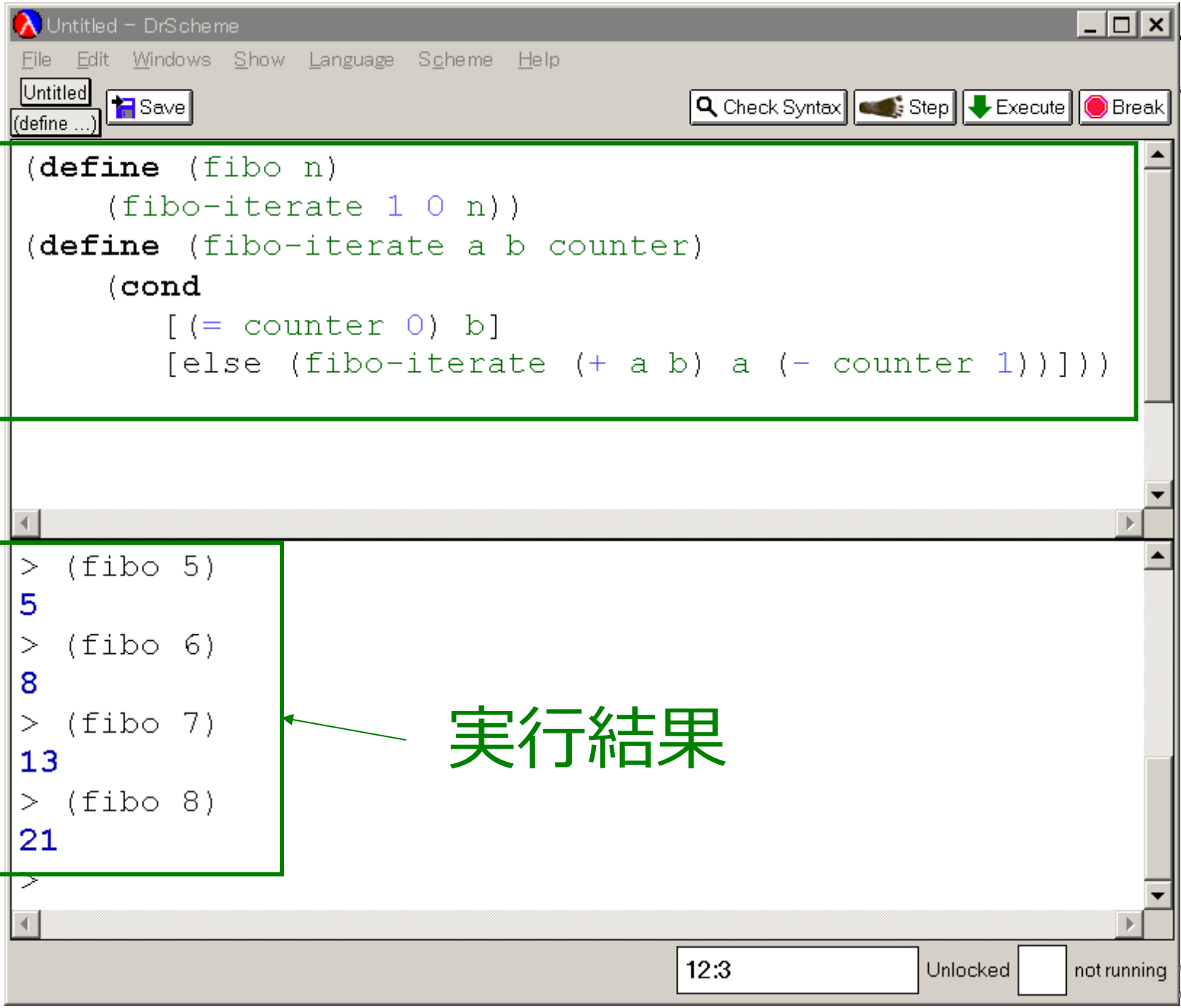
1. 次を「定義用ウインドウ」で，実行しなさい
   - 入力した後に，Execute ボタンを押す

```
(define (fibo n)
   (fibo-iterate 1 0 n))
(define (fibo-iterate a b counter)
   (cond
      [(= counter 0) b]
      [else (fibo-iterate (+ a b) a (- counter 1))]))
```

2. その後，次を「実行用ウインドウ」で実行しなさい

```
(fibo 5)
(fibo 6)
(fibo 7)
```

☆　次は，課題に進んでください


Untitled – DrScheme
File Edit Windows Show Language Scheme Help
Untitled
(define ...)
Save
Check Syntax
Step
Execute
Break
(define (fibo n)
(fibo-iterate 1 0 n))
(define (fibo-iterate a b counter)
(cond
[(= counter 0) b]
[else (fibo-iterate (+ a b) a (- counter 1))]))
> (fibo 5)
5
> (fibo 6)
8
> (fibo 7)
13
> (fibo 8)
21
>
12:3
Unlocked
not running

実行結果

「反復的プロセス」でのフィボナッチ数


```
(define (fibo n)
    (fibo-iterate 1 0 n))
(define (fibo-iterate a b counter)
    (cond
       [(= counter 0) b]
       [else (fibo-iterate (+ a b) a (- counter 1))]))
```

「反復的プロセス」でのフィボナッチ数

1. まず，f_1 (=1), f_0 (=0) から始める
2. 次に，f_0, f_1 を使って，f_2 を求める
3. ・・・
4. n に達するまで続ける

「反復的プロセス」でのフィボナッチ数

a=1, b=0 から開始して,

$a \leftarrow a + b$

$b \leftarrow a$

を繰り返す.

終了条件

```
(define (fibo-iterate a b counter)
  (cond
    [(= counter 0) b]
    [else (fibo-iterate (+ a b) a (- counter
1))]))
```

a ← a + b

b← a

(fibo 4) から 3 が得られる過程の概略

(fibo 4)
= (fibo-iterate 1 0 4)
= ...
= (fibo-iterate 1 1 3)
= ...
= (fibo-iterate 2 1 2)
= ...
= (fibo-iterate 3 2 1)
= ...
= (fibo-iterate 5 3 0)
= ...
= 3

a, b, counter の値が変化する

まとめ

- 例題 1
  - 木構造的な再帰
  - 計算が冗長

    例）　(fibo 4) の計算では，(fibo 2), (fibo 1), (fibo 0) が繰り返し現れる
- 例題 2
  - 反復的プロセス
  - 例題 1 ではあった「冗長な計算」が、例題 2 で無い

17-3 課題

課題１

- 例題１と例題２のフィボナッチ数のプログラムを，性能の面から比較せよ
  - 例題１と例題２のプログラムについて，(fibo 6) を計算するために，例題１の fibo, 例題２の fibo-iterate が何回繰り返し実行されるか数えよ
  - 例題１と例題２のプログラムを，DrScheme のstepper で実行し，最終的な結果が得られるまでに「next」ボタンを押した回数（置き換えが起こった回数）を数えてみよ
    - これは，(fibo 3), (fibo 4), (fibo 5), (fibo 6) について行え