コンピュータサイエンス第２

南出 靖彦

第１回

講義スケジュールと評価

スケジュール

- 12 月 5 日
- 12 月 12 日
- 12 月 19 日
- 12 月 26 日
- 1 月 9 日
- 1 月 16 日
- 1 月 23 日
- 1 月 30 日： 期末試験

評価

- 課題：3〜4 回, 40〜50 %
- 期末試験：50〜60 %

今日の内容

- サブルーチン
- ローカル変数
- グローバル変数
- 再帰呼び出し

準備：講義のホームページから `cs2-day1.zip` をダウンロードする

復習：サブルーチン

- 「関数」としてのサブルーチン

```
# 関数 kaijou(n)（n の階乗）
def kaijou(n)
  ans = 1
  for i in 1..n
    ans = ans * i
  end
  return(ans)
end
# kaijou の定義ここまで

# ここからプログラム本体
x = gets().to_i
puts(kaijou(x))
```

値を返さないサブルーチン

```
# サブルーチン writeX(a)
# 働き：文字 X を a 個横に並べて書いて改行する
def writeX(a)
  for i in 1..a
    print("X")
  end
  print("\n")
end
# writeX の定義ここまで

# ここからプログラム本体
n = gets().to_i
writeX(n)
```

- return を持たない

writeX(式) という文を実行すると，

1. 式 の値が計算される．
2. その値がサブルーチン writeX の変数 a に代入される．
3. writeX の定義部分が実行開始される．
   （サブルーチンが呼び出される，と言う）
4. 実行が終了すると本体に戻ってくる

用語：式 や a のことを引数と呼ぶ

ローカル変数とグローバル変数

- ローカル変数：英小文字から始まる変数
  - サブルーチン（又はプログラム本体）の中でのみ有効
  - プログラム本体や他のサブルーチン内に同じ名前の変数があっても実体は別
- グローバル変数： $ から始まる変数
  - プログラム本体からも各サブルーチン内部からも触ることのできる
  - プログラム実行中ずっと存在し続ける

ローカル変数

- プログラム本体や他のサブルーチン内に同じ名前の変数があっても実体は別なので混同は起こらない。
- 同じサブルーチンが何度も呼び出される場合にもそのサブルーチン定義内のローカル変数は毎回名前は同じだが実体は別になる。

```
def test(a)
  a = 100
  b = 200
  puts(a, b)
end

a = 1
b = 2
test(a)
puts(a, b)
```

実行結果

```
100
200
1
2
```

- 関数 test 中の変数 a とプログラム本体中の変数 a は，別の変数として扱われる

グローバル変数

グローバル変数： $ から始まる変数

- プログラム本体からも各サブルーチン内部からも触ることのできる
- プログラム実行中ずっと存在し続ける

```
def test(a)
  a = 100
  $b = 200
  puts(a, $b)
end

a = 1
$b = 2
test(a)
puts(a, $b)
```

実行結果

```
100
200
1
200
```

- 関数 test の中の$ b への代入が，プログラム本体へも影響する．

再帰呼び出し

サブルーチンの定義の中で，今定義している自分自身を呼び出す．

```
# 階乗の再帰的な定義
def kaijou(n)
  if n <= 1
    return 1
  else
    return (n*kaijou(n-1))
  end
end

# ここからプログラム本体
x = gets().to_i
puts(kaijou(x))
```

kaijou(3) を実行すると

```
   kaijou(3)
=> 3 * kaijou(2)
=> 3 * 2 * kaijou(1)
=> 3 * 2 * 1
   ...
=> 6
```

練習: フィボナッチ数

F_n: n 番目のフィボナッチ数

$$\begin{aligned} F_0 &= 0 \\ F_1 &= 1 \\ F_{n+2} &= F_n + F_{n+1} \quad (n \geq 2) \end{aligned}$$

F_0	F_1	F_2	F_3	F_4	F_5	$\cdots$	F_{10}	$\cdots$
0	1	1	2	3	5	$\cdots$	55	$\cdots$

練習: ファイル `fib.rb` の中の関数 `fib` を完成せよ.

練習: 指数関数

次の考え方で，n^k を計算する関数 `exp(n,k)` を書け

$$\begin{aligned} n^{2k} &= (n^2)^k \\ n^{2k+1} &= n \times (n^2)^k \end{aligned}$$

練習: ファイル `exp.rb` の中の関数 `exp` を完成せよ.

グローバル変数の応用

関数が何回呼び出されるか？

```
def gcd(m,n)
  $calls = $calls + 1
  if n == 0
    m
  else
    return gcd(n, m % n)
  end
end

$calls = 0
puts(gcd(m,n))
puts($calls)
```

gcd(100,60) を実行すると

```
   gcd(100,60)
=> gcd(60,40)
=> gcd(40,20)
=> gcd(20,0)
=> 20
```

再帰呼び出し 6.rb

```
# サブルーチン figure(n)
# 働き：大きさ n の三角形を文字 X で描く

def figure(n)
  if n <= 1
    writeX(n)
  else
    figure(n-1)
    writeX(n)
  end
end
```

figure(5) を呼び出せば次が画面に出力される。

```
X
XX
XXX
XXXX
XXXXX
```

等価なプログラム 7.rb

```
# サブルーチン figure(n)
# 働き：大きさ n の三角形を文字 X で描く

def figure(n)
  if n >= 2
    figure(n-1)
  end
  writeX(n)
end
```

等価なプログラム：なぜ？

```
def figure(n)
  if n >= 2
    figure(n-1)
  end
  writeX(n)
end
```

条件の真と偽を入れ替える

```
def figure(n)
  if n <= 1
     # 何もしない.
  else
    figure(n-1)
  end
  writeX(n)
end
```

writeX(n) を if 文の中に入れる

```
def figure(n)
  if n <= 1
    writeX(n)
  else
    figure(n-1)
    writeX(n)
  end
end
```

【練習問題】8.rb

次のプログラムを実行するとどのような出力がされるか？

```
def test(n)
  if n == 1
    writeX(1)
  end
  if n == 2
    writeX(2)
  end
  if n > 2
    test(n - 1)
    test(n - 2)
    writeX(n)
  end
end

test(5)
```

【課題 1 】: 再帰プログラム

授業ページ（OCWi ではない）から課題用プログラムをダウンロードして，それを以下の問題 (A)(B) の指示に従って改造せよ。そして改造したプログラムを OCWi の課題提出機能で提出せよ。

- 締め切り： 12 月 12 日 午前 10:40
  - 12 月 12 日　午後 9:00 までの提出は採点しますが，大きく減点します.
- 提出するファイル: `kadai.rb`
  - 問題 A, 問題 B の両方の変更を行ったもの.

(問題 A)

サブルーチン figure の定義の中身だけを変更して，次のような出力をするプログラムにせよ。figure の定義以外を変更してはいけない。

１を入力した場合

```
X
```

２を入力した場合

```
X
XX
X
```

３を入力した場合

```
X
XX
X
XXX
X
XX
X
```

４を入力した場合

```
X
XX
X
XXX
X
XX
X
XXXX
X
XX
X
XXX
X
XX
X
```

５以上の入力も同様（類推せよ）

(問題 B)

サブルーチン `writeX` を，働きは変えずに再帰呼び出しを使った定義に書き換えよ（ただし `writeX` に 0 や負数を与えたときの働きは変わってもよい）。while や for などの繰り返し構文を使用してはいけない。