プログラミング概論
http://bit.ly/kosen01

Week9@後期
2016/11/24

本日の内容

- 講義
  - 関数の再帰呼び出し
- 演習
  - フィボナッチ数を返す関数(再帰関数版)

「再帰的」とは？

- 再帰的
  - ある事象が自分自身を含んでいたり、自分自身を用いて定義されている
  - 例) たまねぎ
    玉ねぎの内部では
    同じ構造の
    小さなたまねぎが
    繰り返し現れる

例)階乗(n!)の計算

- 5! は5 x 4 x 3 x 2 x 1
- n!の定義

1. 0! = 1
2. n > 0 とき $n! = n \times (n - 1)!$

- 例） $2! = 2 \times 1!$

$= 2 \times 1 \times 0!$

$= 2 \times 1 \times 1$

kaijou()関数による表現

- C言語のプログラムを書くために
  n!を求める手続きをkaijou(n)と関数の形で
  置くと

  2! = kaijou(2) = 2 × kaijou(1)
  1! = kaijou(1) = 1 × kaijou(0)
  n! = kaijou(n) = n × kaijou(n-1)

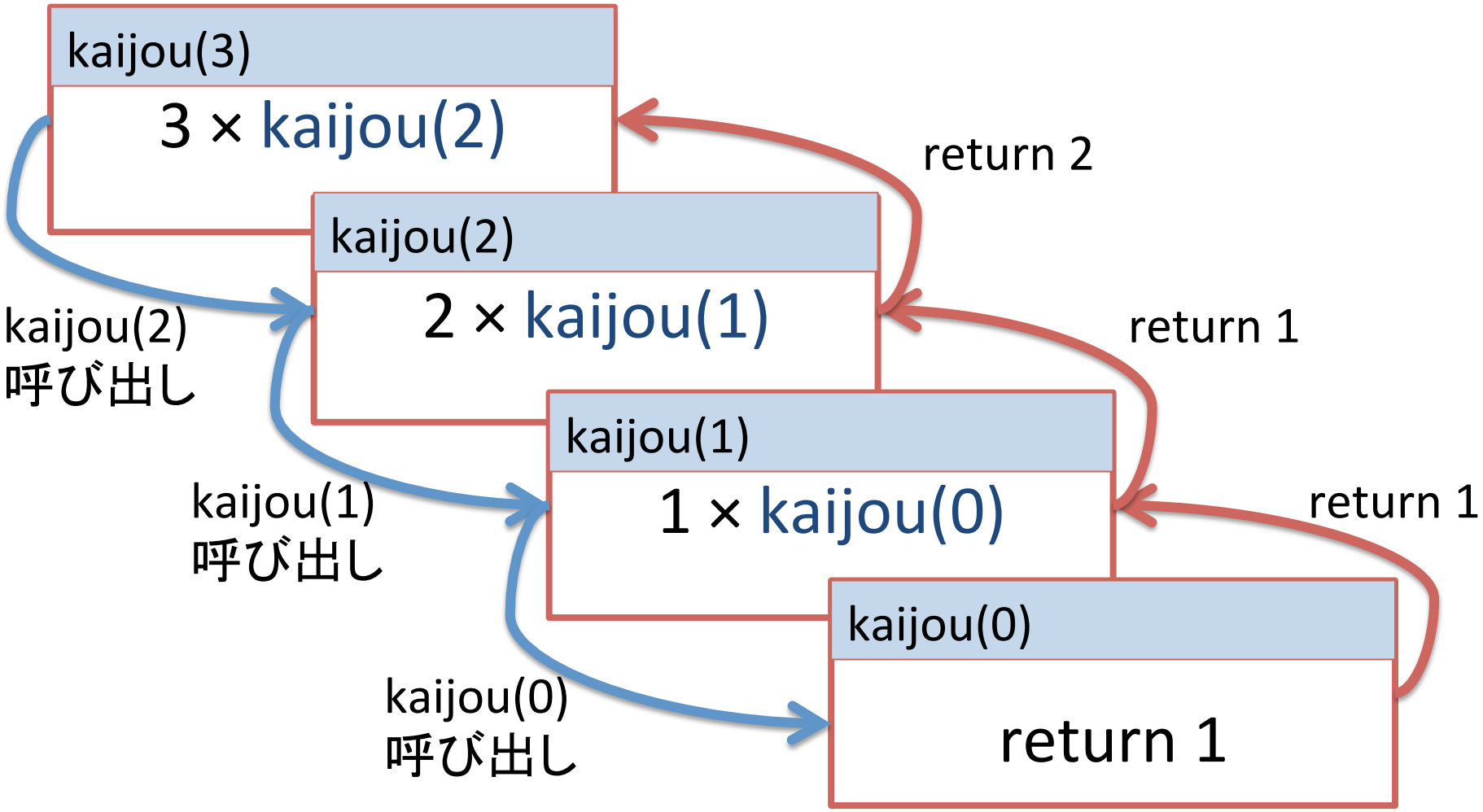
kaijou(n)は
n × kaijou(n-1)で求めることが出来る。
(ただしkaijou(0)は1を返す)

n!を計算するサンプルプログラム (kaijou_recursive.c)

```c
#include <stdio.h>

int kaijou(int n); // 形式宣言
// 関数の本体
int kaijou(int n) {
  if (n > 0) { // nが0でない場合
    return n * kaijou(n-1);
  } else { // nが0の場合
    return 1;
  }
}
// main()関数
int main() {
  printf("3の階乗: %d\n", kaijou(3));
  return 0;
}
```

kaijou()関数の動作



関数の再帰的呼び出し

- 関数の再帰的呼び出し
  - 関数から自分自身と同じ関数を呼び出すこと
    (❌ 自分自身を呼び出す
    ⭕ 自分自身と同じ関数を別途呼び出す)
  - 再帰的に定義される式の計算では再帰的呼び出しを使うと良い
    - 例）フィボナッチ数列(→ 演習)
      - 1 1 2 3 5 8 13・・・

演習0

- 講義資料中のサンプルプログラム「kaijou_recursive.c」を入力し動作を確認しなさい

演習1

- kaijou_recursive.cの5行目と6行目の間に以下の記述を追加し、関数呼び出しの途中経過を出力するように変更しなさい。

  追加する内容：
  printf(“kaijou(%d)が呼び出された。\n”, n);

演習1の正しい出力

kaijou(3)が呼び出された。
kaijou(2)が呼び出された。
kaijou(1)が呼び出された。
kaijou(0)が呼び出された。
3の階乗: 6

関数が再帰的に呼び出されていることを確認する。

演習2(フィボナッチ数)

- フィボナッチ数F_nは以下のように再帰的に定義される
  1. n==0のとき$F_0 = 0$
  2. n==1のとき$F_1 = 1$
  3. n > 1のとき$Fn = F_{n-1} + F_{n-2}$
- フィボナッチ数を整数で返す以下の形式の関数をfibo_recursive.cというファイルに記述しなさい。

  int fibo(int n);

次のページへ続く

演習2(続き)

- main()関数から以下のようにfibo()関数を呼び出し、正しくフィボナッチ数が計算されていることを確認しなさい。

```
int main() {
  int n;
  printf("n=");
  scanf("%d", &n); // nに値を読み込む
  printf("%dのフィボナッチ数は%dです。\n", n, fibo(n));
  return 0;
}
```

演習3

- fibo_recursive.cのmain()関数を書き換え、0番目から10番目までのフィボナッチ数を一覧表示しなさい。

出力例:

```
0番目：0
1番目：1
2番目：1
3番目：2
4番目：3
5番目：5
6番目：8
7番目：13
8番目：21
9番目：34
10番目：55
```

次回

- 複数の変数をまとめて宣言する「配列」という機能を学習します。
- 放課後も演習室を開放しています。
  活用してください。