アルゴリズムとデータ構造

第2回　基本的な抽象データ型

http://www-ui.is.s.u-tokyo.ac.jp/~takeo/course/algorithm/

五十嵐 健夫

takeo@is.s.u-tokyo.ac.jp

前回の内容

イントロダクション
モデル化の例（交差点）
抽象データ型
計算量（Oと**Ω**）

今回の内容

リスト
スタック
待ち行列
写像、連想記憶
再帰呼び出し

抽象データ型としてのリスト(列)

$a_1, a_2, a_3, \ldots, a_{1n}$　**線形順序**

insert, indexOf, get, remove, next, prev,
clear, first, print, end

Purge **の例**

リスト（列）の実現

配列による実現

○ ランダムアクセス　× 挿入と削除

ポインタによる実現 （通常のリスト）

× ランダムアクセス　○ 挿入と削除

配列

○ ランダムアクセス　× 挿入と削除

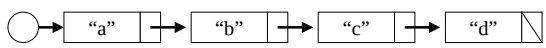
String[] labels = {“a”,”b”,”c” ,”d”}

“a”
“b”
“c”
”d”

リスト

× ランダムアクセス　○ 挿入と削除

```
LinkedList labels = new LinkedList();
labels.add(“a”);
labels.add(“b”);
labels.add(“c”);
labels.add(“d”);
```



スタック

clear, pop, push, empty

例 (消去文字の処理、再帰呼び出し)

実装 (ポインタと配列)

待ち行列 (Queue)

clear, front, enqueue,
dequeue, empty

実装 (ポインタと循環配列)

連想記憶 (Map)

clear, put, get

例 (社員)

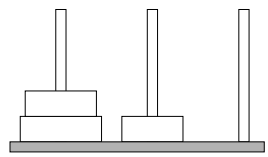
実装 (配列とポインタ)

再帰呼び出し

ハノイの塔

ピンiからn枚円盤をピンjへ移す。

```
void hanoi(int n, int i, int j){
   if (n == 1) move(i,j);
   else {
        int k = 6-i-j; //空ピン
        hanoi(n-1,i,k);
        move(i,j);
        hanoi(n-1,k,j);
   }
}
```

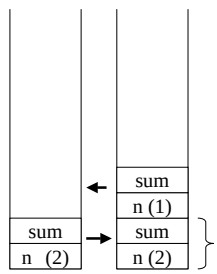


計算量 $O(2^n)$

スタックと再帰呼び出し

$$\sum_{i=1}^{n} i^2$$

```
int foo(int n){
   if (n == 1) return 1
   int sum = foo(n-1)+n*n;
   return sum;
}
```



活動レコード

再帰呼び出しの
繰り返しへの変換

(早くなるが読みにくくなる)

ナップザック問題 (単純版)

重さ $w_1 \ldots w_n$ 重の荷物を
容量tのバッグに積めるか？