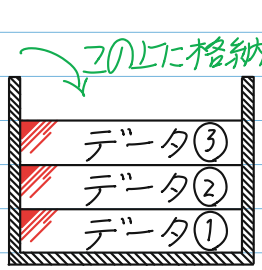
アルゴリズムとプログラミング

▷データの構造

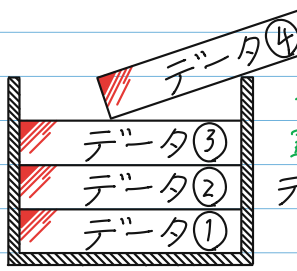
・コンピュータが扱う情報をデータといい、コンピュータの主記憶装置(RAM)に保存される。
・コンピュータの処理能力率はデータの並べ方によって変わり、このデータの並べ方のことをデータの構造という。

・データ構造の種類
・スタック　・キュー　・配列　・連結リスト　・木構造

・スタック　<<アルゴリズム:後入れ先出し(LIFO:Last In First Out)
スタックは、データを1列に並べて、最後に格納したデータを最初に取り出すデータ構造



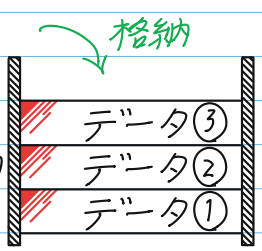
データ④
スタックでの格納を
pushという。
このとき、「データ④」を
格納するのでpush④



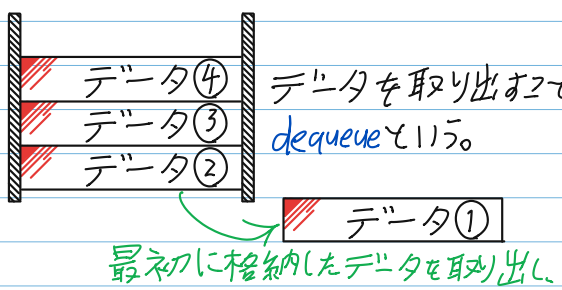
最後に格納したデータから
取り出し
データを取り出すことをpopという。

スタックのデータ構造を利用して、処理の履歴を保存しておけば、元の処理に戻ることができる。

・キュー　<<アルゴリズム:先入れ先出し(FIFO:First In First Out)
キューはデータを1列に並べて最初に格納したデータを最初に取り出すデータ構造



データ④
キューでのデータの格納
をenqueueという。
このとき、「データ④」を格納
するのでenqueue④



データを取り出すことを
dequeueという。
最初に格納したデータを取り出し。

キューのメリットは順番通りに処理されること。

データの格納をするenqueueもデータを取り出すdequeueは
↳ENQ　↳DEQ　と記されることもある。

・配列　データを表にしたもの。
基本情報では配列のひとつひとつを要素といい、要素の位置を要素番号といいます。

・1次元配列…要素が1列に並ぶ配列

例：

	1	2	3	4
配列A	佐藤	鈴木	高橋	今井

↑要素番号
↑要素

プログラミングでは要素番号は0から数える。

例えば配列Aから今井を取り出すときは「A[4]」と書く

1次元配列より大きい配列を多次元配列という。
・2次元配列…要素が縦横に並ぶ配列

例：2次元配列A

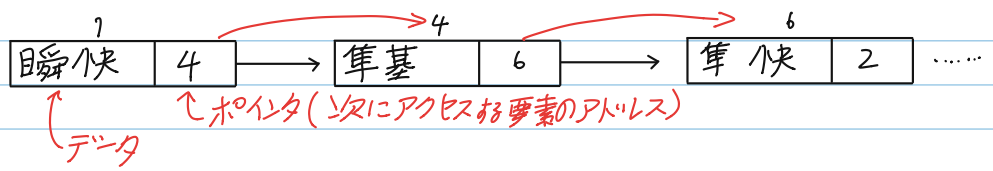
	1	2	3	4
1	隼基	男性	15歳	港区
2	隼快	男性	16歳	南区

この2次元配列から「16歳」を取り出すときは、2行目の3列目を指定する必要がある。なので「A[2,3]」と書く。
配列「行番号、列番号」

・連結リスト…データを数珠つなぎに並べたもの。
形式は配列に似ている。

・ポインタ…連結リスト各要素にポインタと呼ばれるアドレスのようなものがあり、このポインタは次にアクセスする要素のポインタを指す。

例：



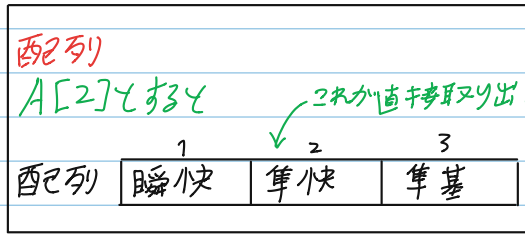
▷連結リストと配列の違い
連結リストは配列と同じようにデータが順に並んでいる。
違う点はアクセスする方法

▷連結リストと配列の違い

・アクセスするおう去：配列では配列名と要素番号を指定することでデータを取り出す。
連結リストでは データの先頭から、順番にアクセスするため、要素が
多いほど、アクセスに時間がかかる。

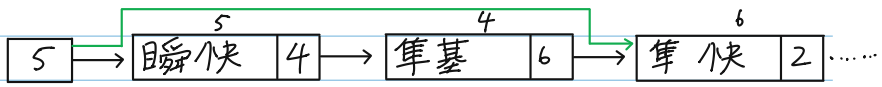
配列

A[2]とすると



連結リスト

このリストから「隼快」を取り出すときは先頭から順にたどってアクセスする。



それぞれのメリット

・配列：データに直接アクセスできる。更新が速い

・連結リスト：ポインタを変更するだけで、データを挿入・削除できる。

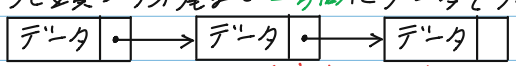
デメリット

・配列：データを挿入・削除するたびに、後のデータをずらす必要がある。

・連結リスト：要素数が多いと、データのアクセスに時間がかかる

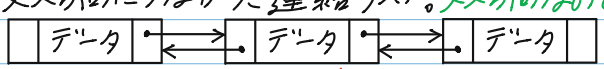
▷連結リストの分類

単方向リスト… 先頭から末尾まで一方向にデータをつなげた連結リスト



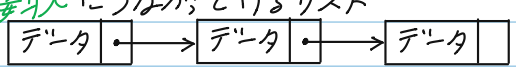
× ← 逆方向には行けない

双方向リスト… 双方向につなげた連結リスト。双方向なので逆にも行ける。

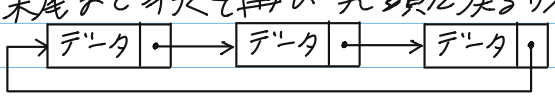


← 両方向に移動できる →

線形リスト… 直接状につながっているリスト

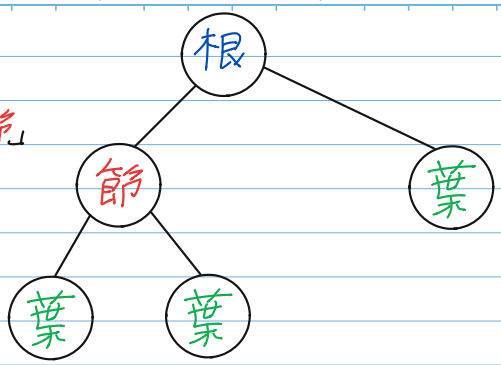


環状リスト… 末尾まで行くと再び先頭に戻るリスト



・木構造…階層構造を持つデータ構造

木構造は始まりの点を「根」、枝分かれする箇所を「節」
末端を「葉」という。

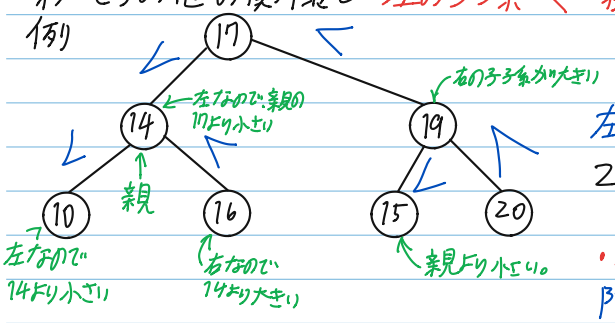


▷木構造の親子関係
木構の分岐元を親、分岐先を子とも言い
親は子を1つでも持つことができる。

木構造で、親が0~2個の子を持つ木構造を2分木、0~3個なら3分木
親が0個~n個の子を持つ木構造をn分木と表す。

▷2分探索木
親と子の値の関係が左の子孫 < 親 < 右の子孫になっているものが2分探索木

例



左の子孫 < 親 < 右の子孫 これ以外は
2分探索木ではない。

・2分探索木のメリット
階層の数だけ値を調べれば目的のデータを見つける
ことができる。

例問題：A、B、C、Dの順に到着するデータに対して、一つのスタックだけを用いて
出力可能なデータはどれか。

ア. A, D, B, C
イ. B, D, A, C
ウ. C, B, D, A
エ. D, C, A, B

スタックでは後に到着したデータを順に取り出すので、最初に
到着したデータは一番最後に出力される。
ここではAが最初に到着しているので、最後に出力される。

答え：ウ

(2) 空の待ち行列に対し、ENQ1、ENQ2、ENQ3、DEQ、ENQ4、ENQ5、DEQ、ENQ6、DEQ、DEQ の操作を行った。次にDEQの操作を行ったとき取り出せるデータを答えよ。

※ENQn：待ち行列にデータを挿入、DEQ：待ち行列からデータ取り出し、

ENQ（エンキュー）、DEQ（デキュー）の操作はキューでのみ行われる。

キューは挿入した順にデータを取り出すので

→

ENQ1、ENQ2、ENQ3、DEQ、ENQ4、ENQ5、DEQ、ENQ6、DEQ、DEQ

（DEQ：1取出、DEQ：2取出、DEQ：3取出、DEQ：4取出）

と出力される。なので、次のDEQでの出力　5

・復習必須の内容

配列、連結リスト、2分探索木の特徴

▷アルゴリズムの基礎

アルゴリズムとは処理を行うための流れを表したもの

・アルゴリズムの表現方法

文章、フローチャート、数式、プログラミング言語の4つのうちどれかを使って表す。

ここでのアルゴリズムの例は数値を絶対値に変換するアルゴリズムを例に解いていきます。

▷文章

文章で「数値を絶対値に変換するアルゴリズム」を表すと、

① $x < 0$ ならば②へ、$x \geqq 0$ ならば③へ　≪条件に応じて②か③ワープ

② $-x \rightarrow x$　≪ xが0未満なら−をかけて、　ワープしたらそこから順次進行

③ xを返す　↙ xを返す

メリット：数学記号と文字が読めれば、その他の知識がなくてもアルゴリズムを理解できる。

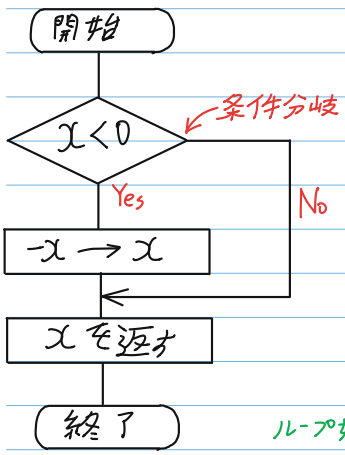
$-x \rightarrow x$　↖ 計算結果

↑

xが0未満なら、−(x)で、例えばxが−1なら $-(-1)=1$

▷フローチャート

フローチャートは命令を記号にしてわかりやすくしたもの。
数値を絶対値に変換するアルゴリズムを表す。

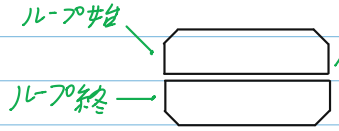


・フローチャートの記号

端子：開始、終了の宣言に使われる。

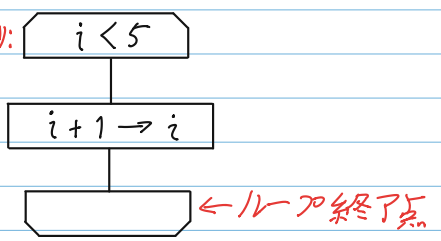
命令：処理を意味する。
A→Bなどのように代入や計算を表現する。この例はAにBを代入するという意味

条件分岐：この中に条件式を入れてtrueなら下の処理を
falseなら横から別の処理まで矢印でつなぐ。



ループ端：条件式に応じてループをする。

例：



解くコツ

それぞれの処理ブロックの処理の結果を表にすると
変数の値などを求めやすくなる。

▷数式

数値を絶対値に変換するアルゴリズムを数式で表す。

$$f(x)\begin{cases} -x & (x<0\text{のとき}) \\ x & (x\geqq 0\text{のとき}) \end{cases}$$

この数式 $f(x)$ は関数を表す記号
括弧内(x)は引数といい、関数内に値を代入するのに使う

関数…処理をひとまとまりにまとめることができるもの。≪プログラミング言語では関数で呼び出して関数内の処理をしている。

この数式の処理の流れ
・引数xの値が0未満なら、負の数なので、正負を反転。
・引数xの値が0以上なら、正の数なので、そのまま値を返す。

・数式の例題

$$F(n)=\begin{cases}1 & (n=0)\\ n\times F(n-1)\end{cases}$$

この数式が表している計算式を書け

仮にnに3を入れると、

$F_{(3)}=3\times F_{(3-1)}\rightarrow 3\times 2$

$F_{(2)}=2\times F_{(2-1)}\rightarrow 2\times 1$

$F_{(1)}=1\times F_{(1-1)}\rightarrow 1\times 0$

$F_{(0)}=1$ となる。

これが表すのは階乗とわかるので計算式 $n!$ //

整数x, y ($x>y\geqq 0$) に対して、次のように定義された関数F(x, y)がある。

$$F(x,y)=\begin{cases}x & (y=0のとき)\\ F(y, x \bmod y) & (y>0のとき)\end{cases}$$

F(231, 15)の値を求めよ。 ※x mod y はxをyで割った余り

$$F(x,y)=\begin{cases}x & (y=0のとき)\\ F(y, x \bmod y) & (y>0のとき)\end{cases}$$

F(231, 15) を F(y, x mod y) に代入すると、

$F_{(15,6)}$ ≪231÷15の余は6なのでy側に6

$F_{(6,3)}$ ≪15÷6の余りは3

$F_{(3,0)}$ ≪6÷3の余りは0 ← yが0になったので x (y=0のとき)の条件を満たす

x y　　xの値を返すということなので

答え：3 //

▷プログラミング言語

プログラミング言語での数値を絶対値に変換するアルゴリズム

```
f(x) :
    if x<0 then          ← then条件が正しいとき
        return -x        ← returnは値を返す
    else                 ← elseはそうでない場合
        return x
```

・プログラミング言語例問題

関数f(x, y)が f(7775, 527) であるとき

f(x, y) : if y = 0 then return x else return f(y, x mod y)の結果を求めよ。

x mod y はxをyで割って出る余り

$f_{(7775,527)}\rightarrow f_{(527,248)}$

$f_{(527,248)}\rightarrow f_{(248,31)}$

$f_{(248,31)}\rightarrow f_{(31,0)}$

yが0になり、xは31、よって結果は 31 //

関数の実行内容に引数を代入する命令があるときは別の条件を満たすまでループする。