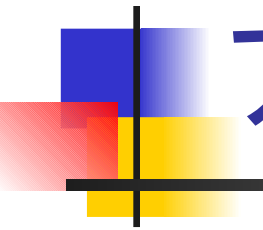
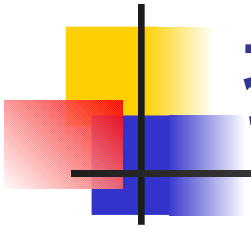


アルゴリズムとデータ構造III

木曜日2時限

鈴木良弥

授業資料　http://ir.cs.yamanashi.ac.jp/~ysuzuki/algorithm3/index.html



授業のねらい

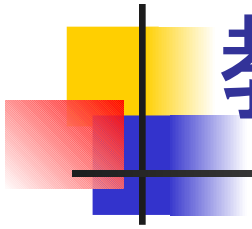
- アルゴリズムとデータ構造I,IIで学んだ事柄の発展的な内容を扱う.
- 事例を通じて，今まで学んだアルゴリズムとデータ構造を組み合わせたアプリケーションのアルゴリズムとデータ構造を学ぶ

他の授業との関連

科目間関係	科目名	キーワード	関連度
先行科目	アルゴリズムとデータ構造Ⅰ	スタック，探索木，グラフ	○
〃	アルゴリズムとデータ構造Ⅰ演習	スタック，探索木，グラフ	
〃	アルゴリズムとデータ構造Ⅱ	グラフ，文字列探索，データ圧縮	○
〃	アルゴリズムとデータ構造Ⅱ演習	グラフ，文字列探索，データ圧縮	
〃	オートマトンと言語	オートマトン，文脈自由文法	○
〃	情報数学	暗号	
同時進行科目	プログラミング言語論	文脈自由文法	
後続科目	ソフトウェア工学	状態遷移図	
〃	ヒューマン・マシンインターフェース	文脈自由文法，DPマッチング，時系列データの圧縮	○
〃	ビジュアルコンピューティング	画像の圧縮	

教科書，参考書(1/2)

- (1)教科書
  - 特に無し.
- (2)参考書
  - 「形式言語と有限オートマトン入門　例題を中心とした情報の離散数学」
    - 小倉久和著, コロナ社, 1996年，ISBN:4-339-02339-6
    - オートマトンと言語の教科書
  - 「アルゴリズムとデータ構造」
    - 湯田幸八，伊原充博共著，コロナ社，2002年，ISBN4-339-01198-3
    - アルゴリズムとデータ構造Ⅰ，Ⅱの参考書



教科書，参考書(2/2)

- 参考書
  - 情報検索アルゴリズム
    - 出版社：共立出版
    - 著者：北研二，津田和彦，獅々堀正幹
    - ISBN4-320-12036-1

授業の予定（中間試験まで）

1	10/02	スタック（後置記法で書かれた式の計算）
2	10/09	チューリング機械，文脈自由文法
3	10/16	構文解析　CKY法
4	10/23	構文解析　CKY法
5	10/30	構文解析　CKY法
6	11/06	構文解析　チャート法
7	11/13	グラフ（動的計画法，ダイクストラ法，DPマッチング）
8	11/20	グラフ（DPマッチング，ビームサーチ，A＊アルゴリズム）
9	11/27	中間試験

授業の予定(中間試験以降)

10	12/04	全文検索アルゴリズム(simple search, KMP)
11	12/11	全文検索アルゴリズム(BM, Aho-Corasick)
12	12/18	全文検索アルゴリズム(Aho-Corasick), データ圧縮
13	01/08	暗号(黄金虫, 踊る人形) 符号化(モールス信号, Zipfの法則, ハフマン符号)テキスト圧縮
14	01/15	テキスト圧縮 (zip), 音声圧縮 (ADPCM, MP3, CELP), 画像圧縮(JPEG)
15	01/29	期末試験

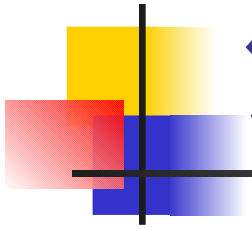
評価

- 演習問題（13点）(A)
- 中間試験（30点）(B)
- 期末試験（57点）(C)

$$評価 = A + B + C$$

- 評価が60点以上なら合格

- 昨年の実績
  - 期末試験まで受験した学生：45人
  - 特別試験を経ずに合格した学生：42人
  - 特別試験を経て合格した学生：2名
  - 期末試験まで受験したが不合格だった学生1名



第1回　10月2日（木）

- スタック（後置記法の計算）
- チューリング機械

数式の記法（オートマトンと言語の復習）

- 前置記法（ポーランド記法）
  - 演算子が先頭
  - *xy
- 中置記法
  - 演算子が真ん中
  - x*y
- 後置記法(逆ポーランド記法）
  - 演算子が最後
  - xy*

数式の記法(1)
前置記法(ポーランド記法)

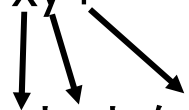
- prefix notation (Polish Notation)
- 例:　*xy
- Lisp言語
  - (car '(A B C))
  - car：リストの第一要素を取り出す
  - (car '(A B C))　→　A

  - 計算方法:左から1文字づつ読み込み，演算子1つと変数2つがそろったら計算し，計算した部分を計算結果に置き換える

数式の記法(2)
中置記法

- infix notation
- 例： x*y
- 数式でよく使われる記法
- 式の意味を一意に確定するために括弧が必要な場合がある.
  - (x+y)*z

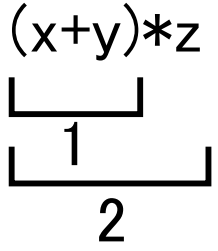
数式の記法(3)
後置記法(逆ポーランド記法)

- postfix notation (Reverse Polish Notation)
- 例： xy*
- Hewlett-Packardの電卓
- 括弧を書かなくても良い.
- 頭の中で計算する順序に近い
- 計算機の中の計算順序と同じ
- 日本語での計算の説明順序と同じ
  - 例： xy+
  - 
    xとyとを足す
- 計算方法：左から1文字づつ読み込み，演算子を読み込んだら直前の2つの変数を使って計算し，計算した部分を計算結果に置き換える

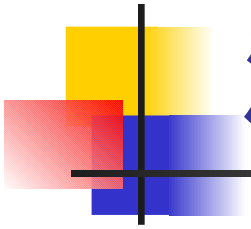
例題

- xy+z*（後置記法）を中置記法に変換
  - xy+z* → (xy+)z*
  - 最初にxy+を計算し，その結果とzをかけ合わせる
  - (x+y)*z （中置記法）

- (x+y)*z (中置記法)を後置記法に変換



  - xy+z* （後置記法）

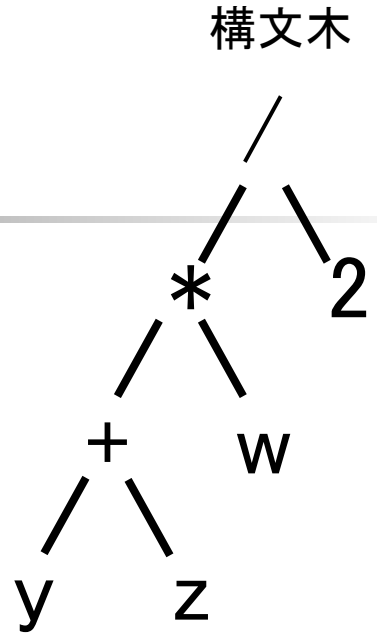


演習問題1

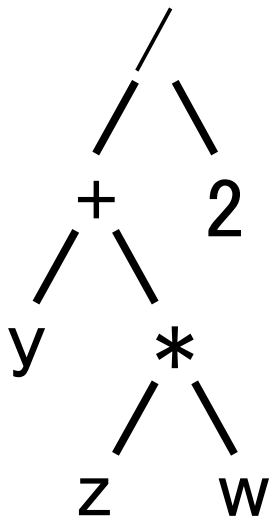
- 中置記法　(y+z)*w/2 を逆ポーランド記法（後置記法）に変換せよ.
- 中置記法　(y+z*w)/2 を逆ポーランド記法（後置記法）に変換せよ.

演習問題1の解答

- 中置記法　(y+z)*w/2
- →後置記法　yz+w*2/

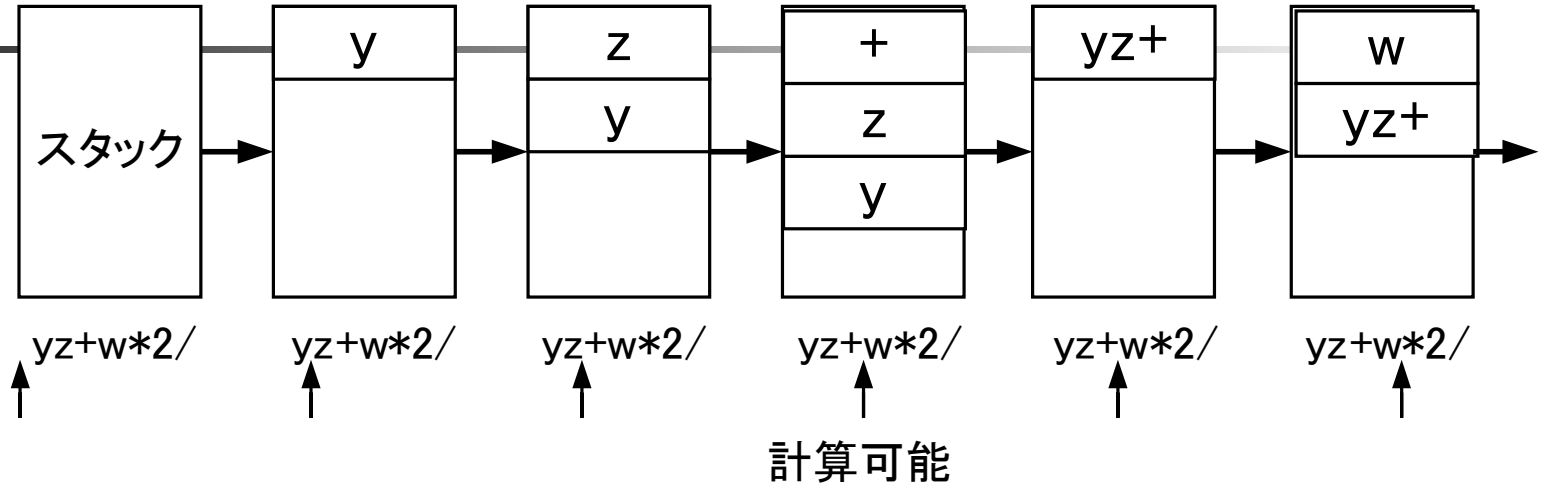


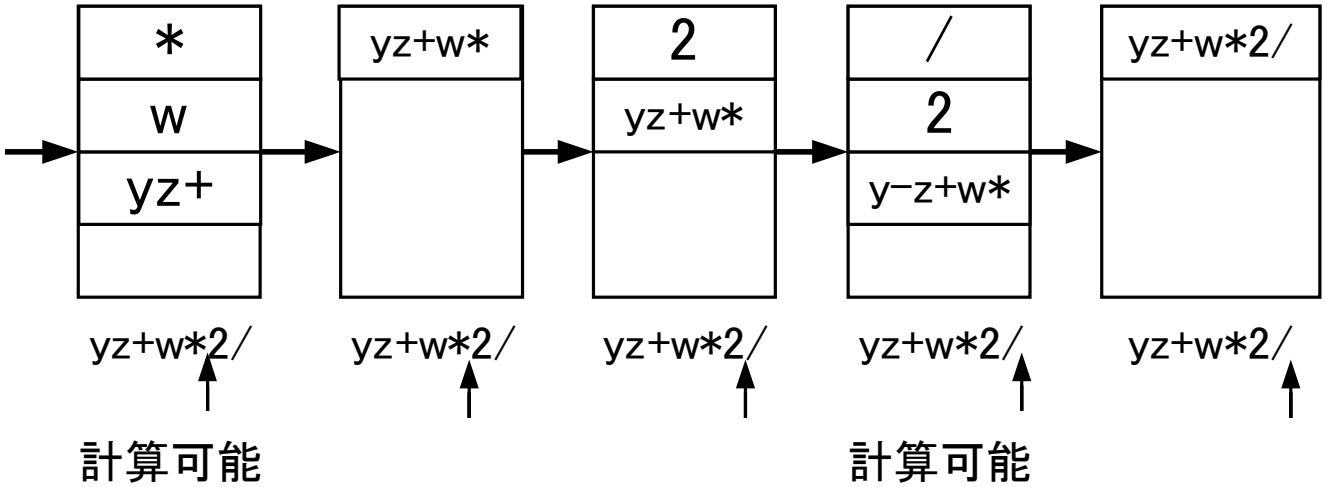
- 中置記法　(y+z*w)/2
- →後置記法　yzw*+2/



yz+w*2/の計算方法(後置記法)

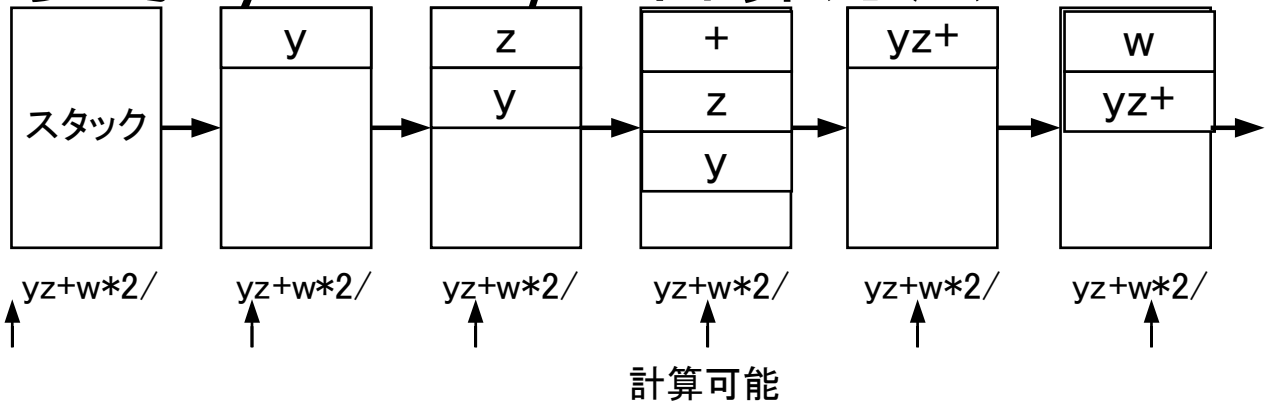
- スタック(Last In First Out)を利用する

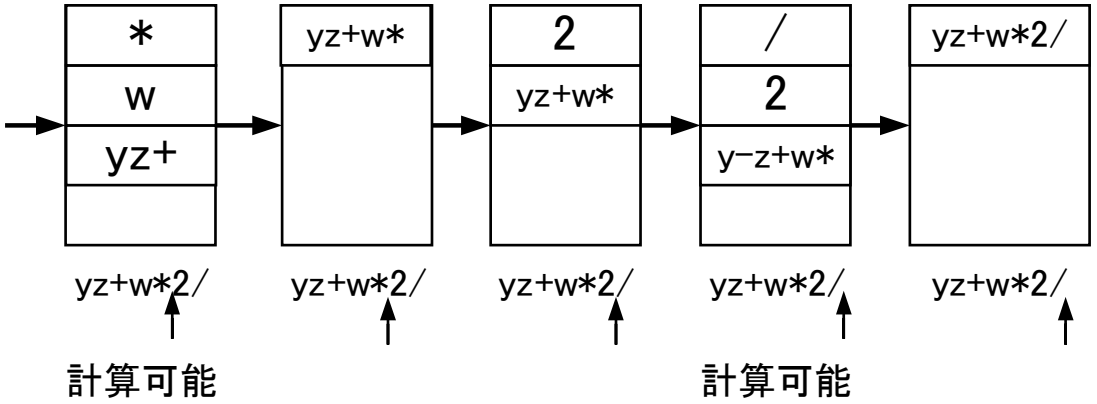




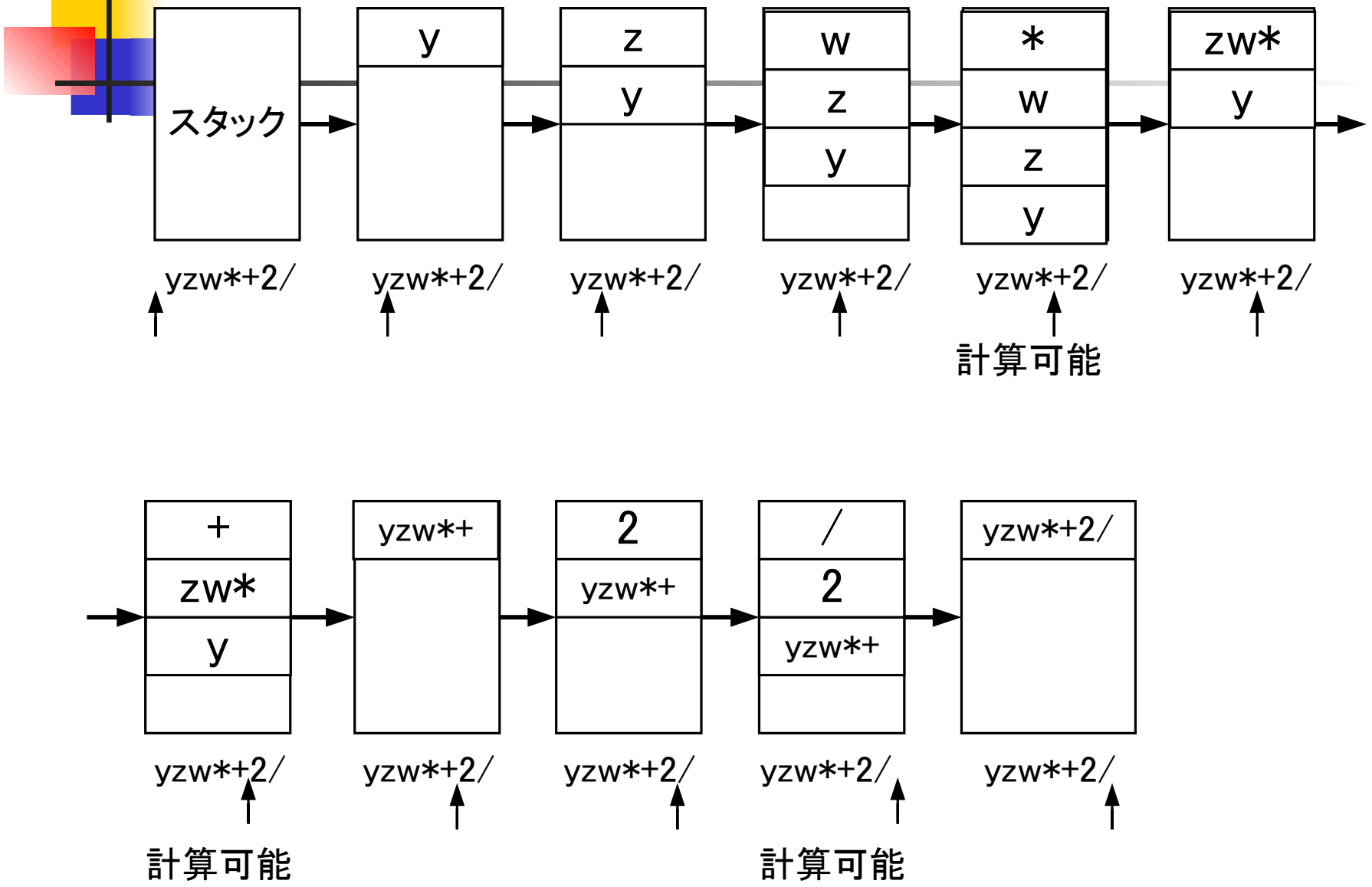
練習問題2
yzw*+2/の計算方法を書け

参考：yz+w*2/の計算方法





練習問題2の解答
yzw*+2/の計算方法（スタックの変化）



７２３＋ －を計算してみよう（アセンブリ言語でプログラミング）

数式(７２３＋ －)がメモリ(データ領域)に書き込まれているとする

3. データ領域から1文字読み込む
    1. 数字(アスキーコード:30H～39H)なら
        - 数値に変換し，数値をスタックにプッシュ
    2. 演算子なら
        1. 一旦スタックにプッシュし，ポップする.
        2. スタックからポップし，数値をBレジスタに取り込む
        3. スタックからポップし，数値をAレジスタ(アキュムレータ)に取り込む
        4. 演算子が＋なら
            - A ＋ B を計算し，Aレジスタに計算結果を格納
        5. 演算子が－なら
            - A －B を計算し，Aレジスタに計算結果を格納
        6. Aレジスタの内容をスタックにプッシュ
4. データ領域すべてを読み終えるまで続ける.

簡単な計算の例　７２３＋ －

```
; 後置記法　７２３＋－の計算
        ORG 8000H ;
        LD HL, DATA ; 数式の先頭番地を指定
LOOP:   LD A, (HL)
        CP 00H
        JP Z, OWARI ; 数式を全部読み込んだら終わり
        LD E, (HL)
        LD D, 0H
        LD A, (HL)
        INC HL
        CP 2BH
        JP Z, LOOPA ; ＋なら加算処理へ
        CP 2DH
        JP Z, LOOPS ; －なら減算処理へ
        LD A, E
        SUB 30H ; 数字なら数値に変換
; Aレジスタの内容をスタックへプッシュ
STPUSH: LD E, A
        LD D, 0H
        PUSH DE ; 読み込んだ数値をスタックへプッシュ
        JP LOOP ; つぎの文字読み込みへ
```

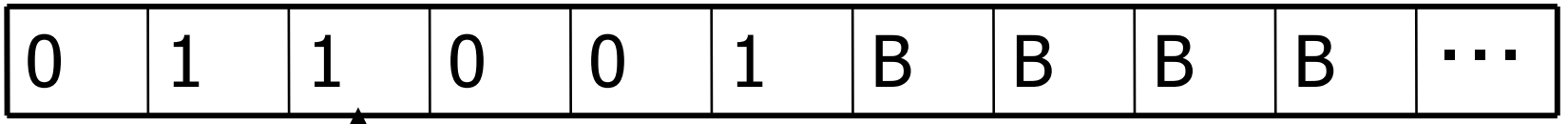
Z80シミュレータで動作を確認できます.

```
;加算
LOOPA:  PUSH DE ; 演算子をスタックへプッシュ
        POP DE ; 演算子をスタックからポップ
        POP DE ; 数値をスタックからポップ
        LD B, E ; スタックトップの値をBレジスタへ
        POP DE ; 数値をスタックからポップ
        LD A, E ; スタックトップの値をAレジスタへ
        ADD A, B ; 加算( A <= A + B )
        JP STPUSH
; 減算
LOOPS:  PUSH DE ; 演算子をスタックへプッシュ
        POP DE ; 演算子をスタックからポップ
        POP DE ; 数値をスタックからポップ
        LD B, E ; スタックトップの値をBレジスタへ
        POP DE ; 数値をスタックからポップ
        LD A, E ; スタックトップの値をAレジスタへ
        SUB B ; 減算( A <= A - B )
        JP STPUSH
;
OWARI: HALT
;入力データ
DATA:   DEFB 37H ;7
        DEFB 32H ;2
        DEFB 33H ;3
        DEFB 2BH ;+
        DEFB 2DH ;-
        DEFB 00H ;END
END
```

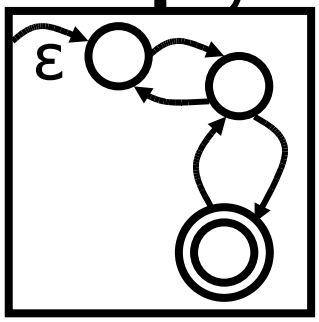
形式言語と有限オートマトン入門
4.5.2 チューリング機械　重要！

- 言語受理能力が最も高いオートマトン
- 半無限長の読み書きが自由にできるテープを用いた有限状態機械

読み書きテープ（初期状態では入力語が記述されている）



読み書きヘッド（初期状態：左端　語の先頭文字位置
テープ上を左右に移動，read,rewrite）



有限状態制御部

最終状態に遷移すると停止して入力語を受理する

チューリング機械(TM)の定義

TM　M=(Q,Σ,Γ,δ,S,B,F)

Q：内部状態の集合

Σ: 入力アルファベット　Bを含まない

Γ: テープ記号の集合　(Γ⊃Σ)

B：空白記号　Γの要素であるがΣの要素ではない

δ: 状態遷移関数　δ: Q×Γ→Q×Γ×{R,S,L}

R:ヘッドを右に移動，S：ヘッドを移動させない，

L：ヘッドを左に移動

S：初期状態　S∈Q

F：最終状態（受理状態）の集合　F⊂Q

ここまで

例題4.71 w1=0101

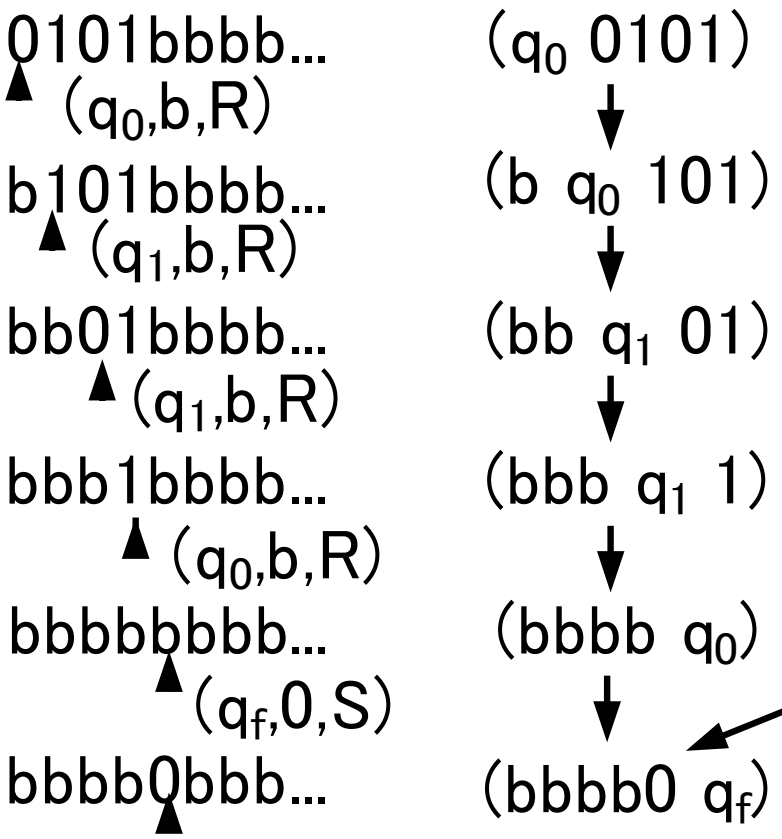
Q={q0,q1,qf}, Σ={0,1}, Γ={0,1,b}, S=q0, B=b, F={qf}

δ	0	1	b
q0	(q0,b,R)	(q1,b,R)	(qf,0,S)
q1	(q1,b,R)	(q0,b,R)	(qf,1,S)
qf	---	---	---

計算状況を示せ.
Σ*上の任意の語と，その最終計算状況におけるテープ上の記号との対応を答えよ

例題4.71　答え
w1=0101

時点表示(計算状況)



δ	0	1	b
q_0	(q_0,b,R)	(q_1,b,R)	$(q_f,0,S)$
q_1	(q_1,b,R)	(q_0,b,R)	$(q_f,1,S)$
q_f	---	---	---

w: 1が奇数個　→　1を出力
w: 1が偶数個　→　0を出力

最終状態q_fに遷移　→　w1を受理

例題4.71 w2′=011010

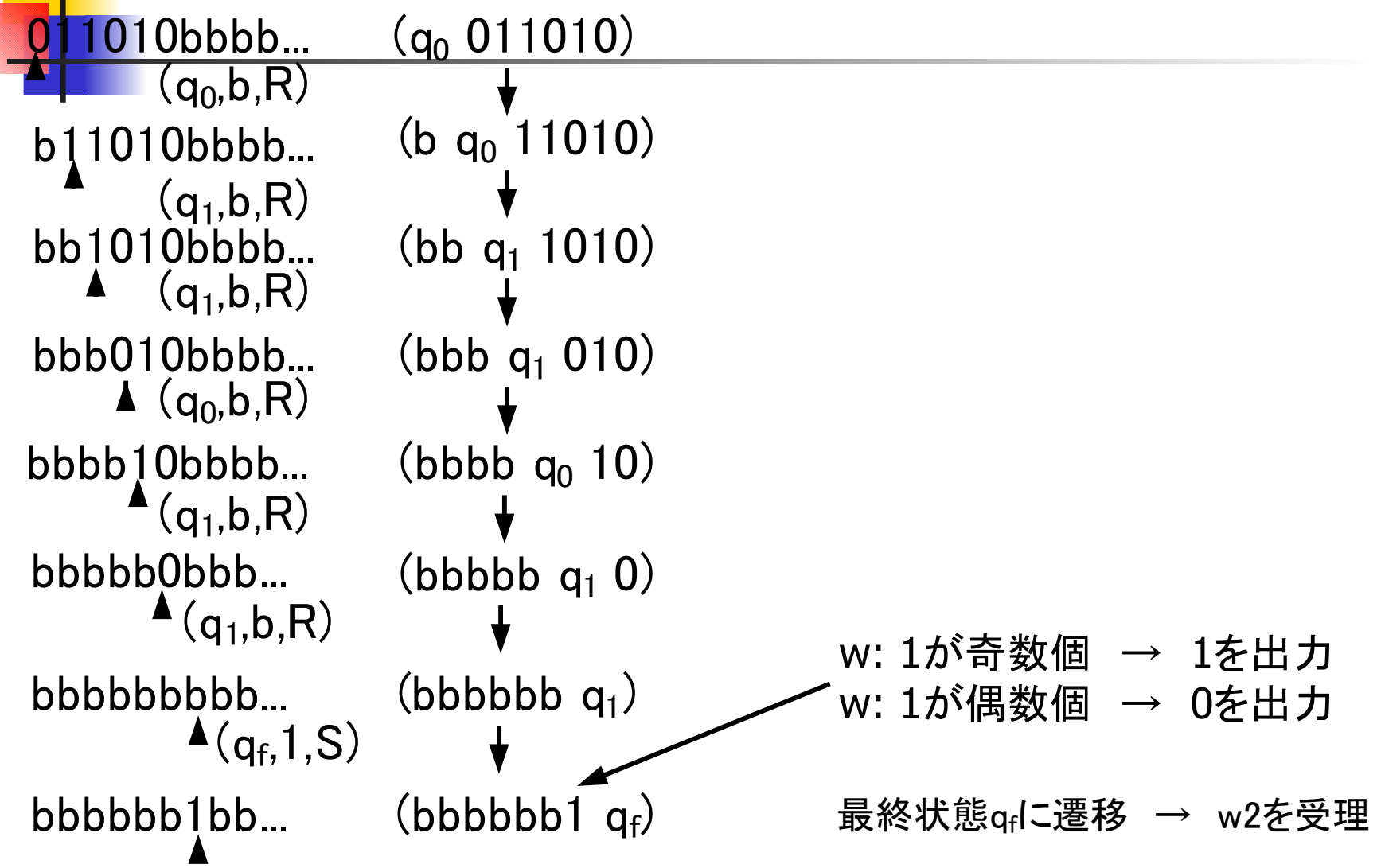
Q={q0,q1,qf}, Σ={0,1}, Γ={0,1,b}, S=q0, B=b, F={qf}

δ	0	1	b
q0	(q0,b,R)	(q1,b,R)	(qf,0,S)
q1	(q1,b,R)	(q0,b,R)	(qf,1,S)
qf	---	---	---

計算状況を示せ.
Σ*上の任意の語と，その最終計算状況におけるテープ上の記号との対応を答えよ

例題4.71　答え W2′=011010

時点表示(計算状況)



練習問題1
例題4.71　w2=01101

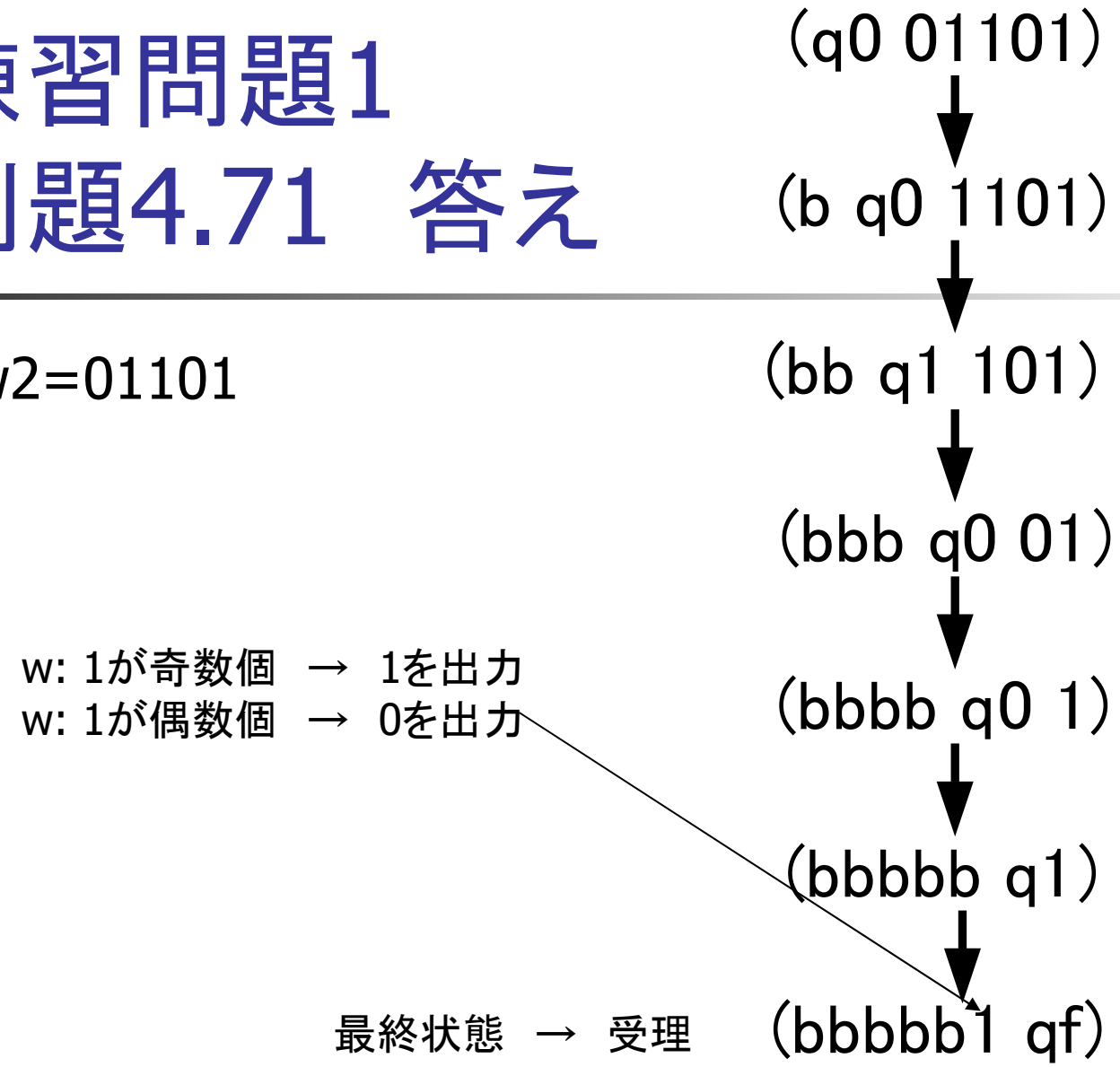
Q={q0,q1,qf}, Σ={0,1}, Γ={0,1,b}, S=q0, B=b, F={qf}

δ	0	1	b
q0	(q0,b,R)	(q1,b,R)	(qf,0,S)
q1	(q1,b,R)	(q0,b,R)	(qf,1,S)
qf	---	---	---

計算状況を示せ.
Σ*上の任意の語と，その最終計算状況におけるテープ上の記号との対応を答えよ

練習問題1
例題4.71　答え

w2=01101



w: 1が奇数個　→　1を出力
w: 1が偶数個　→　0を出力

最終状態　→　受理

4.5.3 オートマトンと計算理論

オートマトンの受理する言語クラス

オートマトン	受理言語型	言語クラス
チューリング機械	第0型言語	句構造言語(PSL)
線形拘束チューリング機械	第1型言語	文脈依存言語(CSL)
プッシュダウンオートマトン	第2型言語	文脈自由言語(CFL)
有限オートマトン	第3型言語	正規言語(RL)

RL ⊂ CFL ⊂ CSL ⊂ PSL （チョムスキーの言語階層）

万能チューリングマシン

- 任意のTMについて，その動作表を与えられるとあたかもそのTMのように振る舞うTM

- コンピュータ
  - プログラム＝動作表（状態遷移関数表）
  - 入力＝入力語
  - コンピュータは万能TM

- チューリングテスト
  - TM M が人間
  - コンピュータ（TM）がTM M を完全に模倣できるか