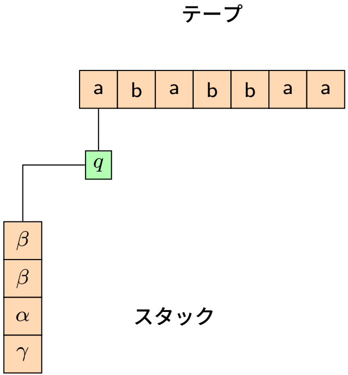
プッシュダウンオートマトン

離散数学・オートマトン
2024 年後期
佐賀大学理工学部　只木進一

プッシュダウンオートマトン: Pushdown automata

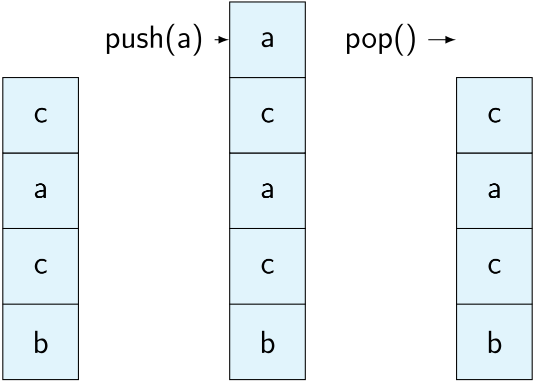
- テープとともに、スタックの文字を読む: Reading stack symbols along with tape symbols
- 状態遷移するとともに、スタックへ文字列を書き込む: Writing strings to the stack along with state transitions
- スタックという特殊な無限に大きなメモリを持つ機械: A machine with a special infinite memory called a stack

プッシュダウンオートマトンのイメージ
Image of a pushdown automaton



スタック: stacks

- リストのような 1 次元のデータ列: A one-dimensional data sequence like a list
- 先頭に書く (push) ことと、先頭から読む (pop) ことだけが許される: Only writing (push) and reading (pop) from the top are allowed
  - 先頭以外のデータは触れない: Other data except at the top cannot be touched
  - FILO (First-In Last-Out)
  - pop: 先頭を取り出して読む、つまり、先頭の要素はスタックから無くなることに注意: Pick up and read the top element, i.e., the top element is removed from the stack


c
a
c
b
push(a)
a
c
a
c
b
pop()
c
a
c
b

Stack: Python での実装例

- `deque` を利用: Use `deque`
- `append()`: 最後に要素を追加: Add an element at the end
- `pop()`: 最後の要素を取り出し、削除: Pick up and remove the last element

Stackクラス定義

```python
from collections import deque
from typing import TypeVar, Generic
T = TypeVar('T')
class Stack(Generic[T]):
    """
    スタックのクラス
    """
    def __init__(self):#コンストラクタ
        self.elements:deque[T] = deque()
    def is_empty(self) -> bool:#要素が無いとき True
        return len(self.elements) == 0
    def push(self,e:T) -> None:#要素を追加
        self.elements.append(e)
    def pop(self) -> T:#末尾の要素を取り出す。要素は削除される
        return self.elements.pop()
    def peek(self) -> T:#末尾の要素を調べる
        return self.elements[-1]
    def size(self) -> int:#要素数を返す
        return len(self.elements)
    def __str__(self) -> str:#文字列化
        s = '['
        for x in self.elements:
            s += str(x)+','
        s = s.removesuffix(',')
        s += ']'
        return s
```

Stack クラス利用例

```
myStack = Stack[str]()
myStack.push('a')
myStack.push('b')
myStack.push('b')
print(myStack)

myStack.pop()
myStack.pop()
print(myStack)
```

https://github.com/discrete-math-saga/PDA

決定性プッシュダウンオートマトン Deterministic Pushdown Automata

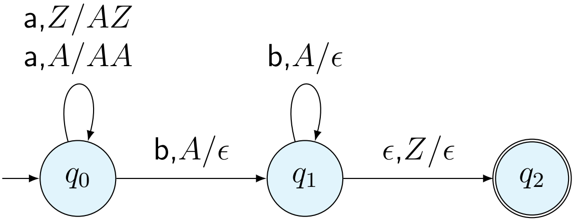
$$M = \langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F \rangle \tag{2.1}$$

- Q: **内部状態の集合**: Set of internal states
- Σ: **テープのアルファベット**: Alphabet of the tape
- Γ: **スタックのアルファベット**: Alphabet of the stack
- $\delta : Q \times \Sigma \times \Gamma \to Q \times \Gamma^*$: **遷移関数**: Transition function
  - **注意: スタックから 1 文字読み、文字列を書き込む**
- $q_0 \in Q$: **開始状態**: Initial state
- $Z_0 \in \Gamma$: **スタックの底の記号**: Symbol at the bottom of the stack
- $F \subseteq Q$: **終状態の集合**: Set of final states

例 2.1:

$$Q = \{q_0, q_1, q_2\}, \qquad F = \{q_2\},$$
$$\Sigma = \{\mathsf{a}, \mathsf{b}\}, \qquad \Gamma = \{A, Z\}.$$

$$\delta(q_0, \mathsf{a}, Z) = (q_0, AZ), \quad \delta(q_0, \mathsf{a}, A) = (q_0, A), \quad \delta(q_0, \mathsf{b}, A) = (q_0, \epsilon),$$
$$\delta(q_1, \mathsf{b}, A) = (q_1, \epsilon), \qquad \delta(q_1, \epsilon, Z) = (q_2, \epsilon).$$

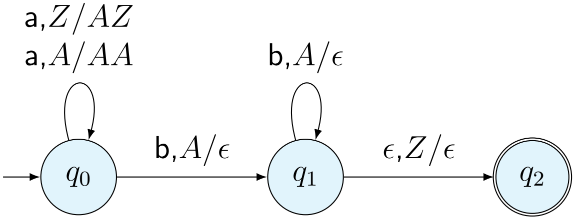


注意:$\delta(q_1, \epsilon, Z)$ は決定的動作であることに注意。スタック文字が Z のときのみ。

動作: (Q, Σ^*, Γ^*) の変化

$$
\begin{aligned}
(q_0, \text{aaabbb}, Z) &\vdash (q_0, \text{aabbb}, AZ) && \text{aを読んでいる間は}q_0\text{に留まる} \\
&\vdash (q_0, \text{abbb}, AAZ) \\
&\vdash (q_0, \text{bbb}, AAAZ) && \text{bを読むと}q_1\text{へ遷移} \\
&\vdash (q_1, \text{bb}, AAZ) \\
&\vdash (q_1, \text{b}, AZ) \\
&\vdash (q_1, \epsilon, Z) && \text{空スタックで}q_2\text{へ遷移} \\
&\vdash (q_2, \epsilon, \epsilon)
\end{aligned}
$$

aabb に対する動作



動作失敗: a と b の数が異なる

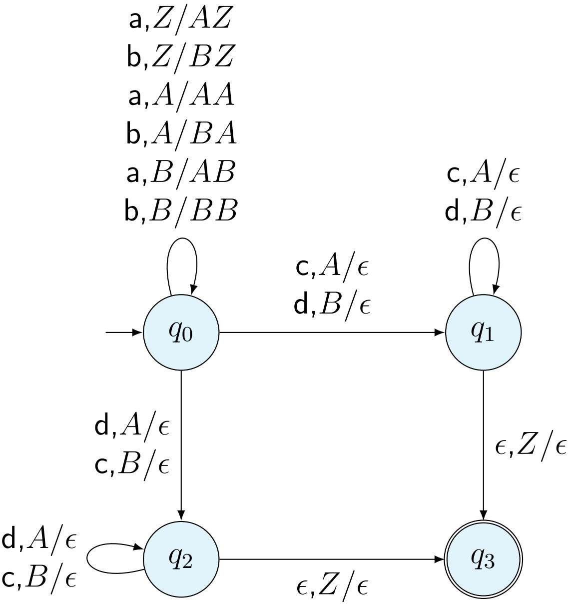
$$
\begin{aligned}
(q_0, \mathsf{aaabb}, X) &\vdash (q_0, \mathsf{aabb}, AZ) \\
&\vdash (q_0, \mathsf{abb}, AAZ) \\
&\vdash (q_0, \mathsf{bb}, AAAZ) \\
&\vdash (q_1, \mathsf{b}, AAZ) \\
&\vdash (q_1, \epsilon, AZ)
\end{aligned}
$$

例 2.2:

$$Q = \{q_0, q_1, q_2, q_3\}, \qquad F = \{q_3\},$$
$$\Sigma = \{\mathsf{a}, \mathsf{b}, \mathsf{c}, \mathsf{d}\}, \qquad \Gamma = \{A, B, Z\}.$$

$$\delta(q_0, \mathsf{a}, Z) = (q_0, AZ), \qquad \delta(q_0, \mathsf{b}, Z) = (q_0, BZ),$$
$$\delta(q_0, \mathsf{a}, A) = (q_0, AA), \qquad \delta(q_0, \mathsf{b}, A) = (q_0, BA),$$
$$\delta(q_0, \mathsf{a}, B) = (q_0, AB), \qquad \delta(q_0, \mathsf{b}, B) = (q_0, BB),$$
$$\delta(q_0, \mathsf{c}, A) = (q_1, \epsilon), \qquad \delta(q_0, \mathsf{d}, B) = (q_1, \epsilon),$$
$$\delta(q_0, \mathsf{d}, A) = (q_2, \epsilon), \qquad \delta(q_0, \mathsf{c}, B) = (q_2, \epsilon),$$
$$\delta(q_1, \mathsf{c}, A) = (q_1, \epsilon), \qquad \delta(q_1, \mathsf{d}, B) = (q_1, \epsilon),$$
$$\delta(q_2, \mathsf{d}, A) = (q_2, \epsilon), \qquad \delta(q_2, \mathsf{c}, B) = (q_2, \epsilon),$$
$$\delta(q_1, \epsilon, Z) = (q_3, \epsilon), \qquad \delta(q_2, \epsilon, Z) = (q_3, \epsilon)$$

注意:$\delta(q_1, \epsilon, Z)$ は決定的動作であることに注意。スタック文字が Z のときのみ。


a,Z/AZ
b,Z/BZ
a,A/AA
b,A/BA
a,B/AB
b,B/BB
c,A/ϵ
d,B/ϵ
c,A/ϵ
d,B/ϵ
q_0
q_1
d,A/ϵ
c,B/ϵ
ϵ,Z/ϵ
d,A/ϵ
c,B/ϵ
q_2
q_3
ϵ,Z/ϵ

動作例:abaaccdc

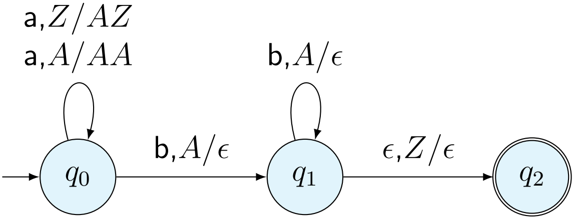
$$
\begin{aligned}
(q_0, \text{abaaddcd}, Z) &\vdash (q_0, \text{baaddcd}, AZ) \\
&\vdash (q_0, \text{aaddcd}, BAZ) \\
&\vdash (q_0, \text{addcd}, ABAZ) \\
&\vdash (q_0, \text{ddcd}, AABAZ) \\
&\vdash (q_2, \text{dcd}, ABAZ) \\
&\vdash (q_2, \text{cd}, BAZ) \\
&\vdash (q_2, \text{d}, AZ) \\
&\vdash (q_2, \epsilon, Z) \\
&\vdash (q_3, \epsilon, \epsilon)
\end{aligned}
$$

動作例:ababdcdc

$$
\begin{aligned}
(q_0, \text{ababdcdc}, Z) &\vdash (q_0, \text{babdcdc}, AZ) \\
&\vdash (q_0, \text{abdcdc}, BAZ) \\
&\vdash (q_0, \text{bdcdc}, ABAZ) \\
&\vdash (q_0, \text{dcdc}, BABAZ) \\
&\vdash (q_1, \text{cdc}, ABAZ) \\
&\vdash (q_1, \text{dc}, BAZ) \\
&\vdash (q_1, \text{c}, AZ) \\
&\vdash (q_1, \epsilon, Z) \\
&\vdash (q_3, \epsilon, \epsilon)
\end{aligned}
$$

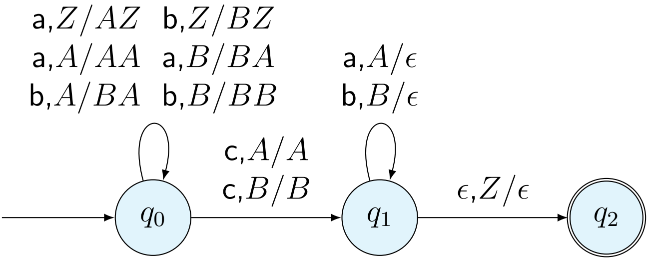
受理言語: Accepted Languages

例 2.1



- **入力とスタックが空になった時に、終状態に居るか？**: Stop automata the final state when the input and stack are empty?
- **例** 2.1: $L = \left\{ \mathsf{a}^i \mathsf{b}^i | i \in N \right\}$
  - a **の数をスタック文字** A **で記録**: Record the number of a's with stack symbols A
  - **テープ上の** b **とスタック上の** A **を照合**: Match b's on the tape with A's on the stack
  - FA **では受理できない言語**: **任意の数の** a **の「数」を記録**: FA cannot accept languages that record the number of a's
- **例** 2.2 **の受理言語は**?: The accepted language of Example 2.2?
- **例** 3.1: $L = \left\{ w\mathsf{c}w^R | w \in (\mathsf{a} + \mathsf{b})^* \right\}$

例 3.1:


a,Z/AZ b,Z/BZ
a,A/AA a,B/BA
b,A/BA b,B/BB
a,A/ϵ
b,B/ϵ
c,A/A
c,B/B
ϵ,Z/ϵ
q_0
q_1
q_2

$$Q = \{q_0, q_1, q_2\}, \qquad F = \{q_2\},$$
$$\Sigma = \{\mathsf{a}, \mathsf{b}, \mathsf{c}\}, \qquad \Gamma = \{A, B, Z\}.$$

$$\begin{aligned}
&\delta(q_0, \mathsf{a}, Z) = (q_0, AZ), \quad &&\delta(q_0, \mathsf{a}, A) = (q_0, AA), \quad &&\delta(q_0, \mathsf{a}, B) = (q_0, AB),\\
&\delta(q_0, \mathsf{b}, Z) = (q_0, BZ), \quad &&\delta(q_0, \mathsf{b}, A) = (q_0, BA), \quad &&\delta(q_0, \mathsf{b}, B) = (q_0, BB),\\
&\delta(q_0, \mathsf{c}, A) = (q_1, A), \quad &&\delta(q_0, \mathsf{c}, B) = (q_1, B),\\
&\delta(q_1, \mathsf{a}, A) = (q_1, \epsilon), \quad &&\delta(q_1, \mathsf{b}, B) = (q_1, \epsilon), \quad &&\delta(q_1, \epsilon, Z) = (q_2, \epsilon).
\end{aligned}$$

動作例

$$
\begin{aligned}
(q_0, \mathsf{abaacaaba}, Z) &\vdash (q_0, \mathsf{baacaaba}, AZ) \\
&\vdash (q_0, \mathsf{aacaaba}, BAZ) \\
&\vdash (q_0, \mathsf{acaaba}, ABAZ) \\
&\vdash (q_0, \mathsf{caaba}, AABAZ) \\
&\vdash (q_1, \mathsf{aaba}, AABAZ) \\
&\vdash (q_1, \mathsf{aba}, ABAZ) \\
&\vdash (q_1, \mathsf{ba}, BAZ) \\
&\vdash (q_1, \mathsf{a}, AZ) \\
&\vdash (q_1, \epsilon, Z) \\
&\vdash (q_2, \epsilon, \epsilon)
\end{aligned}
$$

動作失敗例

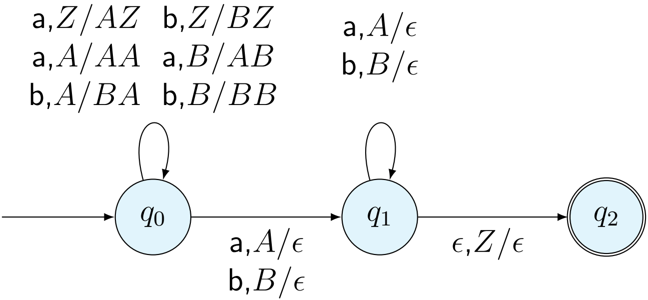
$$
\begin{aligned}
(q_0, \mathsf{abaaaaba}, Z) &\vdash (q_0, \mathsf{baaaaba}, AZ) \\
&\vdash (q_0, \mathsf{aaaaba}, BAZ) \\
&\vdash (q_0, \mathsf{aaaba}, ABAZ) \\
&\vdash (q_0, \mathsf{aaba}, AABAZ) \\
&\vdash (q_0, \mathsf{aba}, AAABAZ) \\
&\vdash (q_0, \mathsf{ba}, AAAABAZ) \\
&\vdash (q_0, \mathsf{a}, BAAAABAZ) \\
&\vdash (q_0, \epsilon, ABAAAABAZ)
\end{aligned}
$$

非決定性プッシュダウンオートマトン Non-deterministic PDA

$$M = \langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F \rangle \tag{4.1}$$

- Q: **内部状態の集合**: Set of internal states
- Σ: **テープのアルファベット**: Alphabet of the tape
- Γ: **スタックのアルファベット**: Alphabet of the stack
- $\delta : Q \times \Sigma \times \Gamma \to 2^{Q \times \Gamma^*}$: **遷移関数**: Transition function
- $q_0 \in Q$: **開始状態**: Initial state
- $Z_0 \in \Gamma$: **スタックの底の記号**: Symbol at the bottom of the stack
- $F \subseteq Q$: **終状態の集合**: Set of final states

例 4.1:


a,Z/AZ b,Z/BZ
a,A/AA a,B/AB
b,A/BA b,B/BB
a,A/ϵ
b,B/ϵ
q_0
q_1
q_2
a,A/ϵ
b,B/ϵ
ϵ,Z/ϵ

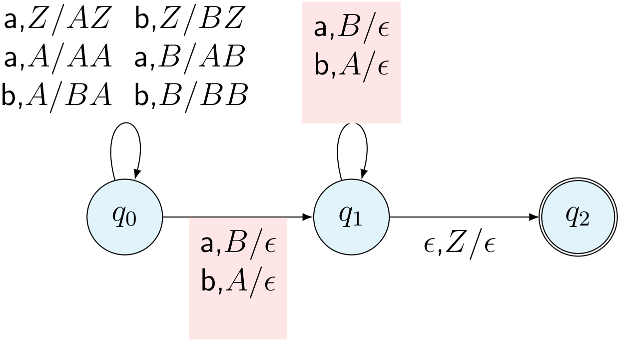
$$Q = \{q_0, q_1, q_2\}\,, \qquad F = \{q_2\}\,,$$
$$\Sigma = \{\mathsf{a}, \mathsf{b}, \mathsf{c}\}\,, \qquad \Gamma = \{A, B, Z\}\,.$$

$$\delta\left(q_0, \mathsf{a}, Z\right) = \{(q_0, AZ)\} \qquad \delta\left(q_0, \mathsf{a}, A\right) = \{(q_0, AA)\,, (q_1, \epsilon)\}$$
$$\delta\left(q_0, \mathsf{a}, B\right) = \{(q_0, AB)\} \qquad \delta\left(q_0, \mathsf{b}, Z\right) = \{(q_0, BZ)\}$$
$$\delta\left(q_0, \mathsf{b}, A\right) = \{(q_0, BA)\} \qquad \delta\left(q_0, \mathsf{b}, B\right) = \{(q_0, BB)\,, (q_1, \epsilon)\}$$
$$\delta\left(q_1, \mathsf{a}, A\right) = \{(q_1, \epsilon)\} \qquad \delta\left(q_1, \mathsf{b}, B\right) = \{(q_1, \epsilon)\}$$
$$\delta\left(q_1, \epsilon, Z\right) = \{(q_2, \epsilon)\}$$

動作 (受理した例)

$$
\begin{aligned}
(q_0, \mathsf{abaaaaba}, Z) &\vdash (q_0, \mathsf{baaaaba}, AZ) \\
&\vdash (q_0, \mathsf{aaaaba}, BAZ) \\
&\vdash (q_0, \mathsf{aaaba}, ABAZ) \\
&\vdash (q_0, \mathsf{aaba}, AABAZ) \\
&\vdash (q_1, \mathsf{aba}, ABAZ) \\
&\vdash (q_1, \mathsf{ba}, BAZ) \\
&\vdash (q_1, \mathsf{a}, AZ) \\
&\vdash (q_1, \epsilon, Z) \\
&\vdash (q_1, \epsilon, \epsilon)
\end{aligned}
$$

例 4.2:


a, Z/AZ b, Z/BZ
a, A/AA a, B/AB
b, A/BA b, B/BB
a, B/ϵ
b, A/ϵ
q_0
q_1
q_2
a, B/ϵ
b, A/ϵ
ϵ, Z/ϵ

$$Q = \{q_0, q_1\}, \qquad F = \{q_1\},$$
$$\Sigma = \{\mathsf{a}, \mathsf{b}, \mathsf{c}\}, \qquad \Gamma = \{A, B, Z\}.$$

$$\delta(q_0, \mathsf{a}, Z) = \{(q_0, AZ)\} \qquad \delta(q_0, \mathsf{a}, A) = \{(q_0, AA)\}$$
$$\delta(q_0, \mathsf{a}, B) = \{(q_0, AB), (q_1, \epsilon)\} \qquad \delta(q_0, \mathsf{b}, Z) = \{(q_0, BZ)\}$$
$$\delta(q_0, \mathsf{b}, A) = \{(q_0, BA), (q_1, \epsilon)\} \qquad \delta(q_0, \mathsf{b}, B) = \{(q_0, BB)\}$$
$$\delta(q_1, \mathsf{a}, B) = \{(q_1, \epsilon)\} \qquad \delta(q_1, \mathsf{b}, A) = \{(q_1, \epsilon)\}$$
$$\delta(q_1, \epsilon, Z) = \{(q_2, \epsilon)\}$$

動作 (受理した例)

$$
\begin{aligned}
(q_0, \text{abaabbab}, Z) &\vdash (q_0, \text{baabbab}, AZ) \\
&\vdash (q_0, \text{aabbab}, BAZ) \\
&\vdash (q_0, \text{abbab}, ABAZ) \\
&\vdash (q_0, \text{bbab}, AABAZ) \\
&\vdash (q_1, \text{bab}, ABAZ) \\
&\vdash (q_1, \text{ab}, BAZ) \\
&\vdash (q_1, \text{b}, AZ) \\
&\vdash (q_1, \epsilon, Z) \\
&\vdash (q_1, \epsilon, \epsilon)
\end{aligned}
$$

- **例** 4.1: $L = \left\{ww^R | w \in (\mathsf{a}+\mathsf{b})^*\right\}$
  - **注意**: **折り返しの文字** c **は不要**: The folding character c is not necessary
- **例** 4.2: w^R **において** a **と** b **を入れ替えた文字列を受理**: Accept strings with a and b swapped in w^R

PDA の受理言語: Accepted Languages of PDA

- **PDA の受理言語は、正規表現では表せない**: PDA's accepted languages cannot be expressed by regular expressions
  - **前半と後半の文字数が同じ、前後を反転などは正規表現では表せない**: The number of characters in the first and second halves are the same, the second half is the reversed string of the first, etc. cannot be expressed by regular expressions
  - $L_{01} = \{0^n 1^n | n \in N\}$ **は正規表現で表せない**: **反復補題**: L_{01} cannot be expressed by regular expressions: Pumping lemma
- **スタックを使うことで、前半の文字列を覚えることができる**: Using a stack, you can remember the first half of the string
  - **長さに制限なし**: No limit on length
- **再帰関数の実装にはスタックが必要**: A stack is required for implementing recursive functions

回文の集合は正規言語ではない: A set of palindromes is not a regular language

- 0 と 1 からなる回文全体 L が正規言語であると仮定: Assume that the entire set of palindromes L consisting of 0's and 1's is a regular language
- 反復補題に現れる n に対して、$w = xyz = 0^n1^n1^n0^n$ を考える: Consider $w = xyz = 0^n1^n1^n0^n$ for n appearing in the pumping lemma
- $|xy| \leq n$ より、$y = 0^k(1 \leq k \leq n)$

$$xy^0z = xz = 0^{n-k}1^n1^n0^n \notin L$$

- L は正規言語ではない: L is not a regular language

回文 (palindrome) 受理する Python コード: 再帰

```python
def palindrome(inputStr:str) -> bool:
    if len(inputStr) <= 1:
        return True
    if inputStr[0] != inputStr[-1]:
        return False
    return palindrome(inputStr[1:-1])
```

回文 (palindrome) 受理する Python コード: 非再帰

```
def palindrome2(inputStr:str) -> bool:
    myStack:Stack[str] = Stack()
    n = len(inputStr)
    m = n//2
    for i in range(m):#前半をスタックに積む
        myStack.push(inputStr[i])
    if n % 2 !=0:#入力の長さが奇数の場合
        m += 1
    for j in range(m,n):#後半と照合
        c = myStack.pop()
        if c != inputStr[j]:
            return False
    return True
```