

# Pred(「1以上の2進数」に対し、1引いた2進数を出力)

- ここに上げたのは私が2022年の春に「C言語による計算の理論(鹿島亮 著)」を読んだときに興味を持って作ったプログラムです。第9章の章末問題でしたが、解答がないので自作です。
- 「C言語による計算の理論」の Turing machine は一度に「文字を1つ書くこと」と「左右に1つ進むこと(または動かないこと)」の2つの事ができます。
- 「2進法」も「aやbの個数」も使います。終了する時は数字列の左端に移動する必要はありません。
- 状態の数が少ないので遷移図も書きました。遷移図とプログラムの rules が異なる場合はプログラムのほうが正解です(^\_^;)
- TuringMachineの仕様は TuringMachine[rules,初期状態,step数] となります。  
ruleは{q,s}→{q',s',dir} (q:現在の状態,s:ヘッドの下文字,q':次の状態,s':ヘッドが書き込む文字,dir:ヘッドの進む方向.右,左,留の3通りで,+1,-1,0で指定)  
初期状態は{{q0,pos},tape} (q0:最初の状態,pos:最初のヘッドの位置,tape:最初のテープの状態)となります。
- 例えば{1,a}→{2,b,+1}は「状態1でheadの文字がaなら「ヘッドの下に文字bを書いて、ヘッドは右に1つ進み、内部状態は状態2へ移る」という事です。
- 同じ例は遷移図の方では「(1の書いてある丸)→(ab右)→(2の書いてある丸)」と表されます。
- tapeの初期状態は変数[tape]に入っています。初期状態は変えることができますが、両端に空白がそれぞれ2個以上「最後まで残る」様にして下さい。
- 最後のManipulateではテープをクリックしてdragすると、画面の大きさが変わります。

## ■ 補助program

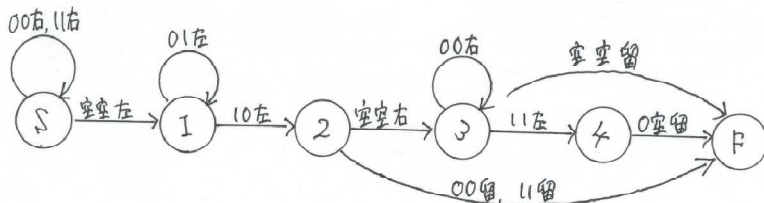
```
ClearAll["Global`*"]

(*ヘッドの位置を[]で、状態は添字で表示する*)
qbracket[x_List] := (*状態+ヘッドの位置. プログラムの検証にGood*)
  ReplacePart[x[[2]], x[[1]][[2]] → Subscript["[" <> ToString[x[[2]][[x[[1]][[2]]]] <> "]", x[[1], 1]]]

(*out には TuringMachine[]の出力が入る*)
turing[out_] :=
  Manipulate[
    Block[{now, tape, control, pos, state, contents, boxes},
      now = out[[step]];
      tape = now[[2]];
      control = now[[1]];
      pos = control[[2]];
      state = control[[1]];
      boxes = Graphics[Table[Line[{{i, 0}, {i + 1, 0}, {i + 1, 2}, {i, 2}, {i, 0}}], {i, 1, Length[tape]}]];
      contents = Graphics[Table[Text[Style[tape[[i]], Large], {i + .5, 1}], {i, 1, Length[tape]}],
        Green, Polygon[{{pos, -1}, {pos + 1, -1}, {pos + 1, -.5}, {pos + .5, 0}, {pos, -.5}}],
        Black, Text[Style[state, Medium], {pos + 0.5, -0.7}]];
      Show[{boxes, contents}], {step, 1, Length[out], 1}]
```

## ■ main program

```
Import["https://mixedmoss.com/mathematica/pred.jpg"]
```



- sucとはほぼ同様だが、 $1000_2$ などは状態[2]でやめると $0111_2$ になるので、先頭の0を消去するための[3]が必要。さらに $0_2$ は消去してはいけないので[4]も必要。

```
In[2237]= rule = {{s, 0} → {s, 0, +1}, {s, 1} → {s, 1, +1}, {s, " "} → {1, " ", -1}, {1, 0} → {1, 1, -1}, {1, 1} → {2, 0, -1},
  {2, 0} → {F, 0, 0}, {2, 1} → {F, 1, 0}, {2, " "} → {3, " ", 1}, {3, 0} → {3, 0, +1}, {3, 1} → {4, 1, -1}, {4, 0} → {F, " ", 0}};

In[2255]= tape = {" ", " ", " ", " ", " ", 1, 0, 1, 0, 0, " ", " "};

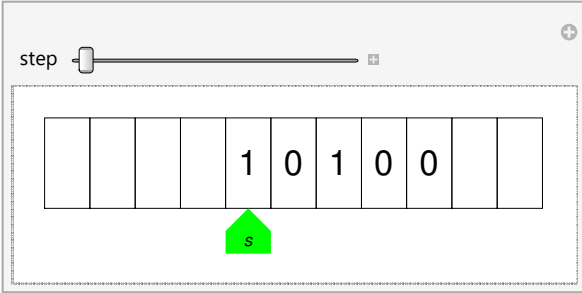
Out[2255]= { , , , 1, 0, 1, 0, 0, , }

In[2260]= pred = TuringMachine[rule, {{s, 5}, tape}, 10]
(*FirstPosition[%[[All,1,1],F][[1]]-1*)

Out[2260]= {{{s, 5, 0}, { , , , 1, 0, 1, 0, 0, , }},
  {{s, 6, 1}, { , , , 1, 0, 1, 0, 0, , }}, {{s, 7, 2}, { , , , 1, 0, 1, 0, 0, , }},
  {{s, 8, 3}, { , , , 1, 0, 1, 0, 0, , }}, {{s, 9, 4}, { , , , 1, 0, 1, 0, 0, , }},
  {{s, 10, 5}, { , , , 1, 0, 1, 0, 0, , }}, {{1, 9, 4}, { , , , 1, 0, 1, 0, 0, , }},
  {{1, 8, 3}, { , , , 1, 0, 1, 0, 1, , }}, {{1, 7, 2}, { , , , 1, 0, 1, 1, 1, , }},
  {{2, 6, 1}, { , , , 1, 0, 0, 1, 1, , }}, {{F, 6, 1}, { , , , 1, 0, 0, 1, 1, , }}}
```

In[2261]:= `turing[pred]`

Out[2261]=



The output cell displays a slider labeled "step" at the top. Below the slider is a grid of 10 cells containing the binary sequence "1 0 1 0 0". A green arrow labeled "s" points to the first "1" in the sequence.

|  |  |  |  |   |   |   |   |   |  |
|--|--|--|--|---|---|---|---|---|--|
|  |  |  |  | 1 | 0 | 1 | 0 | 0 |  |
|--|--|--|--|---|---|---|---|---|--|