BB5 (状態数が5, 文字数1 の Busy Beaver)

- 注意：この notebook はやや時間のかかる (5分程度の) プログラムを含んでいます．notebook 全体でなく，1個ずつ評価することをお勧めします．なお評価は「初期条件を変えなければ不要」です．
  - 遷移関数は Cat on Mat さんからお借りしました．　後で気づきましたが Wiki にもまとまって載っていました．
  - 状態数がnで1文字のBBの書ける最大の文字数をB[n]とします．Cat on Mat さんによると B[3]=6, B[4]=13, B[5]=4098, B[6] = 46×10^{1439} (WikiによるとB[6] = $3.5\times10^{18\,267}$) だそう．B[5]，B[6]については正確な数字はまだ分かっていないそうです．
  - サイトのリンクを張っておきます．

cat on mat （busy beaver）

ビジービーバ Wiki

  - Mathematica組み込みのTuringMachineの仕様は TuringMachine[rule,初期状態,総step数] となります．
    ruleは{q,s}→{q',s',dir} (q:現在の状態,s:ヘッドの下の文字,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.+1,-1,0)
    初期状態は{{q_0,pos},tape} (q_0: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》→(ab右)→《2》」と表されます．
  - tapeの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1個以上「最後まで残る」様にして下さい．

補助 Program

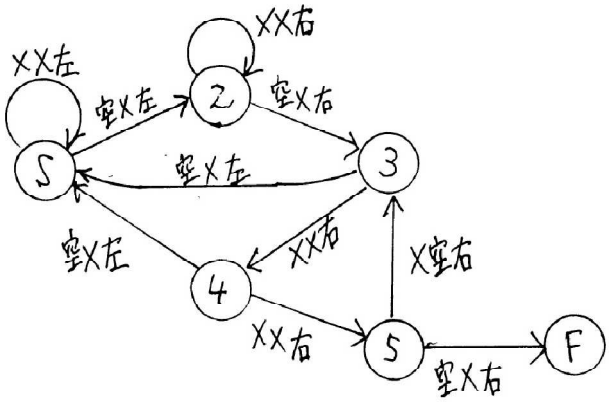
```
ClearAll["Global`*"]

turing[out_, comment_] := (*TuringMachine の結果の表示に使う*)
 Manipulate[
  Block[{now, len, tape, control, pos, state, contents, boxes},
   len = Length[out[[1, 2]]];
   now = out[[step]];
   tape = now[[2]];
   control = now[[1]];
   pos = control[[2]];
   state = control[[1]];
   boxes = {Line[{{1, 0}, {1, 2}}], Table[Line[{{i, 0}, {i + 1, 0}, {i + 1, 2}, {i, 2}}], {i, 1, len}]};
   contents = {Table[Text[Style[tape[[i]], Tiny, Bold], {i + .5, 1}], {i, 1, len}], (*Tinyに変更*)
     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}]};
   Graphics[{boxes, contents}, PlotRange → {{0, len + 2}, {-1.2, 2}}]], Style[comment, 12, Italic],
  {step, 1, Length[out], 1}](*PlotRange→Allでは何故か駄目*)

turing[out_] := turing[out, ""]
```

Main Program

```
Import["https://mixedmoss.com/mathematica/turing/jpg/BB5.jpg"]
```




```
rule = {{s, " "} → {2, X, -1}, {s, X} → {s, X, -1}, {2, " "} → {3, X, +1}, {2, X} → {2, X, +1}, {3, " "} → {s, X, -1},
    {3, X} → {4, X, +1}, {4, " "} → {s, X, -1}, {4, X} -> {5, X, +1}, {5, " "} → {F, X, +1}, {5, X} → {3, " ", +1}};
```

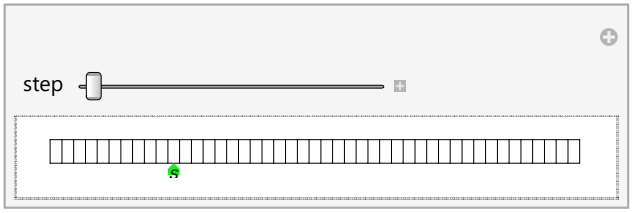
- これはCat on Mat さんから借りてきた遷移関数ですが，(Wikiにも同様のものがあります) Cat on Mat さん自身は始めはPythonで組んで5分ぐらい掛かり，次にCでプログラムして14秒で4098個を数えて終了したそうです．しかしB[5] はまだ正確には分かっていないそうなので，B[5]はもっと大きい可能性があります．Cat on Mat さんの所には Python と C のプログラムも置いてあります．
- B[5]をMathematicaで計算しようとしたのですが，4098個ということは，テープの長さも少なくとも4098区画は必要です．すると重くなり，MathematicaのTuringMachine コマンドでは1日以上掛けても計算できませんでした．経験上，1日で駄目なものは一週間でも駄目なので，自分で少し改良することにしました．

[1] default の TuringMachine コマンドをそのまま使用．しかし終わらない…

- まずはテープ全体を全ての区画が表示できる様に500回ぐらいでやってみます．

```
short = Table[" ", 45];
bb5 = TuringMachine[rule, {{s, 11}, short}, 500];
```

```
turing[bb5](*テープの部分をクリックしてdragして下さい．大きくなります．*)
```



- 次は終了状態まで達するように実行したい所ですが，危険なので，コメントアウトしてあります．ちなみに下の回数mは改良型のマシーンを動かして得た回数です．速いマシーンをお持ちの方は自己責任で評価してみて下さい．私は10万回が一晩かけても終わりませんでした．

```
(*longtape=Table[" ",8000];
m=11798826;
bb5long=TuringMachine[rule,{{s,1000},longtape},m]*)
```

[2] TuringMachine コマンドを改良して マシンが終了状態になるまで動かす．

- 上記のように一晩かけても終わらなかったので自分で少し改良することにしました．基本的なアイデアは現在のヘッドの前後でテープを短く切ります．(下では名前はcutとしてます．幅は 2w+1 です．) このcutに対してTuringMachineコマンドでn(n<=w)ステップ進めます．このときの出力の第2要素(newcut) を長いテープの元の場所にはめ込んで newtape を作ります．(ヘッドは一度に1つしか進めないのでヘッドはcut からはみ出しません．ですから長いテープでTuringMachineを働かせたのと同じ結果になります．) これをNestWhileListで繰り返します．するとnステップ毎(以下1ブロックと言う)の結果が得られます．これで終了状態Fまで到達できました．(^o^) 個数はやはり4098個でした．回数は11798826回 (約1200万回), 時間は最短で4分30秒掛かりました．(Pythonと同じですね.) しかし私は7年ぐらい前に買った CPUがCore i5-4670 という骨董品を使っているので，今のPC/Mac ならず～と速いと思います．
- 改良Program

```
(*{状態q,ヘッド位置h,全tape}={{q,h},{" ",...,X,...,X,...," "}}から,nステップ後のセット(newset)を作る*)
n = 100; w = 100;(*n≤wでないと駄目.[3]参照*)
turingMachine[{{q_, h_Integer}, tape_List}] := Block[{cut, newset, newcut, newh, newq, newtape},
  cut = Take[tape, {h - w, h + w}];
  newset = TuringMachine[rule, {{q, w + 1}, cut}, n] // Last;
  newq = newset[[1, 1]];
  newh = h + newset[[1, 2]] - (w + 1);
  newcut = newset[[2]];
  newtape = Join[Take[tape, h - w - 1], newcut, Take[tape, {h + w + 1, Length[tape]}]];
  {{newq, newh}, newtape}
 ]
```

■ これを終了状態になるまで NestWhileList で繰り返します. 私の骨董品のPCでは270秒掛かりました. (ちなみにn=m=50なら306秒,n=m=200なら280秒でした.)

!historyの評価は時間とメモリーがかかります!

```
longtape = Table[" ", 8000];
notyetF[x_] := UnsameQ[x[[1, 1]], F];
history = NestWhileList[turingMachine, {{s, 1000}, longtape}, notyetF];
```

■ 終了状態まで行きました. !(^_^)! そして X は4098個. 総ステップ数 m=11798826 でした.

```
Count[Last[history][[2]], X]
```

```
4098
```

```
Length[history]
```

```
117 990
```

```
FirstPosition[TuringMachine[rule, history[[% - 1]], n][[All, 1, 1]], F] // First
```

```
27
```

```
m = (%% - 2) * n + (% - 1)
```

```
11 798 826
```

[3] 結果の分析

各ステップ (n,2n,3n...) 時に於いて, 左端と右端のXの位置と個数を求めます.

```
posX1 = Map[FirstPosition[#[[2]], X] &, Drop[history, 1]] // Flatten;
posX2 = Map[Length[longtape] + 1 - FirstPosition[Reverse[#[[2]]], X] &, Drop[history, 1]] // Flatten;
kosuu = Map[Count[#[[2]], X] &, Drop[history, 1]]
```

```
{17, 24, 30, 30, 37, 43, 46, 51, 54, 57, 43, 51, 57, 62, 66, 69, 72, 74, 77, 80, 81, 83, 86,
 87, 89, 90, 93, 95, 96, 98, 91, 74, 81, 87, 92, 95, 98, 101, 102, 105, 108, 110, 111, 114, 116,
 ··· 117 899 ···, 5538, 5505, 5472, 5438, 5405, 5372, 5338, 5305, 5272, 5238, 5205, 5172, 5138,
 5105, 5072, 5038, 5005, 4972, 4938, 4905, 4872, 4838, 4805, 4772, 4738, 4705, 4672, 4638, 4605,
 4572, 4538, 4505, 4472, 4438, 4405, 4372, 4338, 4305, 4272, 4238, 4205, 4172, 4138, 4105, 4098}
```

大きい出力 | 表示を少なく | もっと表示 | すべて表示 | 大きさ制限の設定...

```
Max[kosuu]
Position[kosuu, %] // Flatten
```

```
6142
```

```
{117 866, 117 867, 117 868, 117 869, 117 870, 117 871, 117 872, 117 873, 117 874, 117 875, 117 876, 117 877, 117 878,
 117 879, 117 880, 117 881, 117 882, 117 883, 117 884, 117 885, 117 886, 117 887, 117 888, 117 889, 117 890,
 117 891, 117 892, 117 893, 117 894, 117 895, 117 896, 117 897, 117 898, 117 899, 117 900, 117 901, 117 902,
 117 903, 117 904, 117 905, 117 906, 117 907, 117 908, 117 909, 117 910, 117 911, 117 912, 117 913, 117 914,
 117 915, 117 916, 117 917, 117 918, 117 919, 117 920, 117 921, 117 922, 117 923, 117 924, 117 925, 117 926}
```

■ 面白いことに個数は単調増加してません. 最大個数は6142個もあります. しかも何回も達しています. 全て終了の直前です. でもここでは終了できなかったようです.

さらにXの密度 (密度が1の時は隙間なく埋まっている状態) を求めます. 後のグラフを見ると分かりやすいです.

```
mitsudo = kosuu / (posX2 - posX1 + 1)
```

$$\left\{1, \frac{12}{13}, 1, \frac{10}{13}, \frac{37}{43}, \frac{43}{47}, \frac{46}{49}, \frac{51}{53}, \frac{54}{55}, 1, \frac{43}{59}, \frac{17}{22}, \frac{57}{70}, \frac{31}{37}, \frac{33}{38}, \frac{23}{26}, \frac{9}{10}, \frac{37}{41}, \frac{11}{12}, \frac{40}{43}, \frac{81}{86}, \frac{83}{88}, \frac{43}{45}, \frac{29}{30}, \frac{89}{92}, \frac{45}{46}, \frac{93}{94}, \frac{95}{96}, \cdots 117\,933 \cdots, \frac{1243}{1536}, \frac{823}{1024}, \frac{1635}{2048}, \frac{203}{256}, \frac{2419}{3072}, \frac{4805}{6144}, \frac{1193}{1536}, \frac{2369}{3072}, \frac{4705}{6144}, \frac{73}{96}, \frac{773}{1024}, \frac{1535}{2048}, \frac{381}{512}, \frac{2269}{3072}, \frac{4505}{6144}, \frac{559}{768}, \frac{2219}{3072}, \frac{4405}{6144}, \frac{1093}{1536}, \frac{723}{1024}, \frac{1435}{2048}, \frac{89}{128}, \frac{2119}{3072}, \frac{4205}{6144}, \frac{1043}{1536}, \frac{2069}{3072}, \frac{4105}{6144}, \frac{4098}{6145}\right\}$$

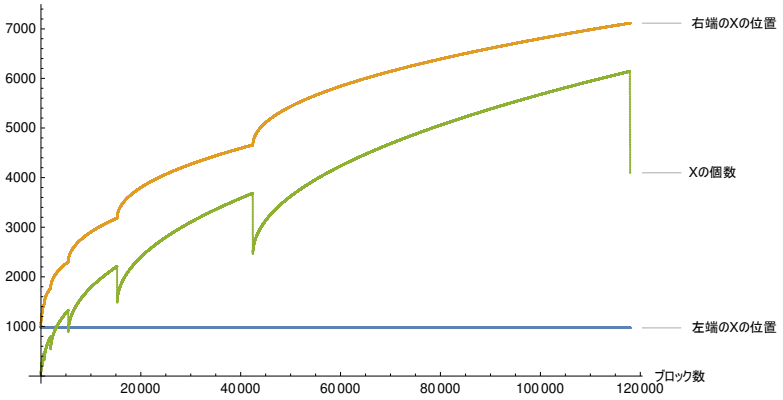
大きい出力 | 表示を少なく | もっと表示 | すべて表示 | 大きさ制限の設定...

```
Position[mitsudo, 1] // Flatten
```

```
{1, 3, 10, 29, 30, 85, 86, 87, 88, 245, 246, 247, 248, 249, 250, 693, 694, 695, 696, 697, 698, 699, 700, 701, 702,
 1949, 1950, 1951, 1952, 1953, 1954, 1955, 1956, 1957, 1958, 1959, 1960, 1961, 1962, 1963, 1964, 5453, 5454,
 5455, 5456, 5457, 5458, 5459, 5460, 5461, 5462, 5463, 5464, 5465, 5466, 5467, 5468, 5469, 5470, 5471, 5472,
 5473, 5474, 5475, 5476, 5477, 5478, 15 208, 15 209, 15 210, 15 211, 15 212, 15 213, 15 214, 15 215, 15 216, 15 217,
 15 218, 15 219, 15 220, 15 221, 15 222, 15 223, 15 224, 15 225, 15 226, 15 227, 15 228, 15 229, 15 230, 15 231, 15 232,
 15 233, 15 234, 15 235, 15 236, 15 237, 15 238, 15 239, 15 240, 15 241, 15 242, 15 243, 15 244, 15 245, 15 246, 15 247,
 15 248, 15 249, 15 250, 15 251, 42 344, 42 345, 42 346, 42 347, 42 348, 42 349, 42 350, 42 351, 42 352, 42 353,
 42 354, 42 355, 42 356, 42 357, 42 358, 42 359, 42 360, 42 361, 42 362, 42 363, 42 364, 42 365, 42 366, 42 367,
 42 368, 42 369, 42 370, 42 371, 42 372, 42 373, 42 374, 42 375, 42 376, 42 377, 42 378, 42 379, 42 380, 42 381,
 42 382, 42 383, 42 384, 42 385, 42 386, 42 387, 42 388, 42 389, 42 390, 42 391, 42 392, 42 393, 42 394, 42 395,
 42 396, 42 397, 42 398, 42 399, 42 400, 42 401, 42 402, 42 403, 42 404, 42 405, 42 406, 42 407, 42 408, 42 409,
 42 410, 42 411, 42 412, 42 413, 42 414, 42 415, 42 416, 42 417, 117 804, 117 805, 117 806, 117 807, 117 808, 117 809,
 117 810, 117 811, 117 812, 117 813, 117 814, 117 815, 117 816, 117 817, 117 818, 117 819, 117 820, 117 821, 117 822,
 117 823, 117 824, 117 825, 117 826, 117 827, 117 828, 117 829, 117 830, 117 831, 117 832, 117 833, 117 834, 117 835,
 117 836, 117 837, 117 838, 117 839, 117 840, 117 841, 117 842, 117 843, 117 844, 117 845, 117 846, 117 847, 117 848,
 117 849, 117 850, 117 851, 117 852, 117 853, 117 854, 117 855, 117 856, 117 857, 117 858, 117 859, 117 860, 117 861,
 117 862, 117 863, 117 864, 117 865, 117 866, 117 867, 117 868, 117 869, 117 870, 117 871, 117 872, 117 873, 117 874,
 117 875, 117 876, 117 877, 117 878, 117 879, 117 880, 117 881, 117 882, 117 883, 117 884, 117 885, 117 886, 117 887,
 117 888, 117 889, 117 890, 117 891, 117 892, 117 893, 117 894, 117 895, 117 896, 117 897, 117 898, 117 899, 117 900,
 117 901, 117 902, 117 903, 117 904, 117 905, 117 906, 117 907, 117 908, 117 909, 117 910, 117 911, 117 912, 117 913,
 117 914, 117 915, 117 916, 117 917, 117 918, 117 919, 117 920, 117 921, 117 922, 117 923, 117 924, 117 925, 117 926}
```

Xの範囲, 個数のグラフです.

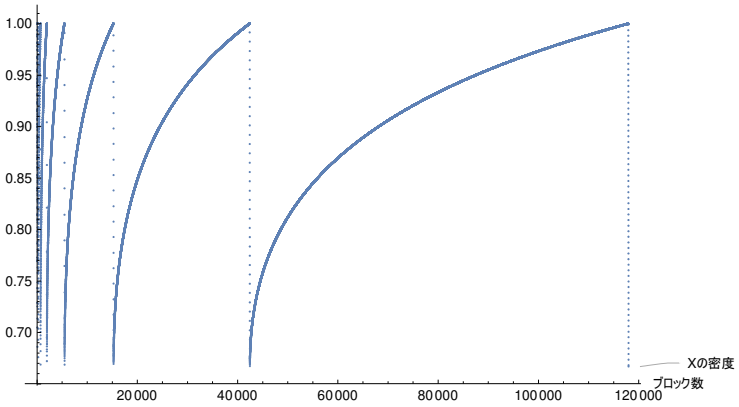
```
ListPlot[{Labeled[posX1, "左端のXの位置"], Labeled[posX2, "右端のXの位置"], Labeled[kosuu, "Xの個数"]},
 AxesLabel → {"ブロック数"}]
```



■ Xは右の方にばかり広がっているようです. 左には殆ど動いていません.

今度はXの密度のグラフです．大きな波があります．単調減少ではありません．何度も急速に落下してます．

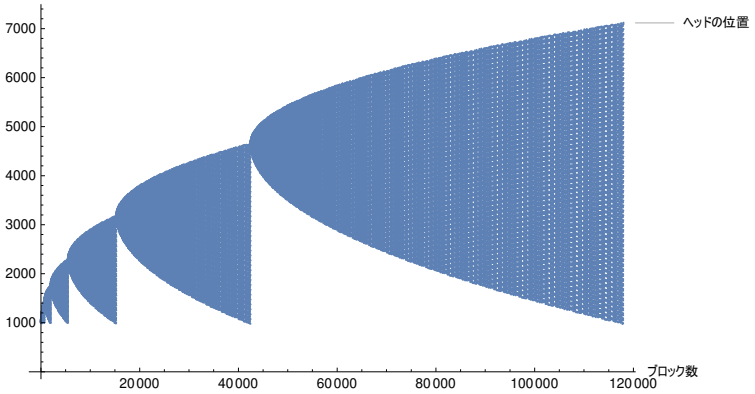
```
ListPlot[Labeled[mitsudo, "Xの密度"], AxesLabel → {"ブロック数"}]
```



次にヘッドの位置の変化を見てみます．ブロック単位の平均速度になります．下のグラフは面白いです．結節点のstep数はほぼ公比3の等比数列になっています．

```
head = Map[#[[1, 2]] &, history];
```

```
ListPlot[Labeled[head, "ヘッドの位置"], AxesLabel → {"ブロック数"}]
```

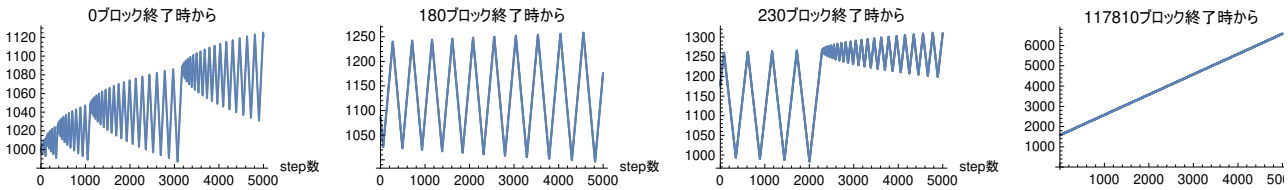


- 上の曲線と下の曲線の乖離が大きい点は速く動いていて，この2曲線の結節点 (目視では約5 箇所) では殆どヘッドは動いてないはずです．(平均速度の話)

さらにヘッドが「1step毎に」どれだけ動くか調べます．

- 先のグラフは「ブロックごと(1ブロック=n steps)」のヘッドの位置の変化でしたが，今度は n×k (kは自然数) 経った時点から，「1回ごと」に 5000回までの変化を調べます．
  全部は調べられないので開始直後と，密度=1 の点（おそらくグラフの結節点と同じ) の前後，さらに 終了点の直前を調べます，
  !評価にやや時間がかかります!

```
checklist = {0, 180, 230, 117810};
ListPlot[TuringMachine[rule, history[[# + 1]], 5000][[All, 1, 2]],
    AxesLabel → {"step数"}, PlotLabel → ToString[#] <> "ブロック終了時から"] & /@ checklist // GraphicsRow
```



- ブロック数 k=0 の時は，振動が徐々に大きくなり，突然すぼみまた徐々に振動が増えています．これほど分かり易くは無いですが，他の「密度=1 の点」の時も同様です．また終了前はず～と右に動いています．（これが[2]のプログラムで n≤w とした理由です)

各状態の出現頻度を見てみます．最初はブロック単位，次はステップ単位です．

```
history[[All, 1, 1]]
```

{s, s, 2, 2, s, 2, 2, 2, s, s, s, 3, 2, 2, s, 2, 2, 2, s, s, s, 2, s, s, 2, s, 2, 2, s, 2, s, 4, s, 2, 2, s, s, s, s, 2,
 2, 2, s, 2, 2, s, 2, s, 2, s, s, 2, s, 2, s, s, 2, s, 2, s, 2, s, 2, 2, s, 2, s, s, 2, s, 2, 2, s, 2, 2, s, 2, 2,
 s, 2, 2, s, 2, 2, s, 2, 2, s, s, 4, 2, 2, s, ··· 117804 ···, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s,
 s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, 3, 4, 5, 3, 4, 5, 3, 4, 5, 3, 4, 5, 3, 4, 5, 3, 4, 5, 3, 4, 5, 3,
 4, 5, 3, 4, 5, 3, 4, 5, 3, 4, 5, 3, 4, 5, 3, 4, 5, 3, 4, 5, 3, 4, 5, 3, 4, 5, 3, 4, 5, 3, 4, 5, 3, 4, F}

大きい出力 | 表示を少なく | もっと表示 | すべて表示 | 大きさ制限の設定...

```
Tally[%]
```

{{s, 58916}, {2, 58884}, {3, 87}, {4, 52}, {5, 50}, {F, 1}}

■ 圧倒的に状態sと状態2が占めます．しかもほぼ同数です．step単位で見ると次の様になります．開始から 500 steps(=5 blocks) の結果です．

```
bb5[[All, 1, 1]]
```

{s, 2, 3, 4, s, s, s, 2, 3, 4, 5, 3, 4, s, s, 2, 2, 2, 2, 2, 3, s, s, s, s, s, s, s, s, 2, 3, 4, 5, 3, 4, 5, 3, 4, 5, 3,
 s, 2, 2, 2, 2, 3, s, s, s, s, s, s, s, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, s, s, s, s, s, s, s, s, s, s, s, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2, 2, 3, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, 2, 3, 4, 5, 3, 4, 5, 3, 4, 5, 3, 4, 5,
 3, 4, 5, 3, 4, 5, 3, s, 2, 2, 2, 2, 3, s, s, s, s, s, s, s, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, s, s, s, s, s, s, s, s, s, s, s,
 s, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, 2, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, 2, 2, 2, 2, 2, 2,
 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s,
 s, s, s, s, s, s, s, s, s, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3,
 s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, 2, 3, 4, 5, 3, 4,
 5, 3, 4, 5, 3, 4, 5, 3, 4, 5, 3, 4, 5, 3, 4, 5, 3, 4, 5, 3, 4, 5, 3, 4, 5, 3, 4, 5, 3, s, 2, 2, 2, 2, 3, s, s, s, s,
 s, s, s, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, s, s, s, s, s, s, s, s, s, s, s, s, s, s, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
 3, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 3, s,
 s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, s, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2}

```
Tally[%]
```

{{s, 217}, {2, 200}, {3, 40}, {4, 23}, {5, 21}}

終了直前の50ステップの様子です．"kiritori" を使って，ステップ単位の変化を見る事ができます．

```
(*kiritori[{開始ステップ, 終了ステップ}]. *)

kiritori[{start_Integer, end_Integer}] := Block[{grp, motodata, heads, hmin, hmax, modified},
  grp = Quotient[start, n];
  motodata = Drop[TuringMachine[rule, history[[grp + 1]], end - grp * n], start - n * grp];
  heads = motodata[[All, 1, 2]];
  hmin = Min[heads];
  hmax = Max[heads];
  modified = {#[[1]] - {0, hmin - 2, 0}, Take[#[[2]], {hmin - 1, hmax + 1}]} & /@ motodata;
  turing[modified, "「step」は," <> ToString[start] <>
    "steps からの増加量．テープは" <> ToString[hmin - 1] <> "~" <> ToString[hmax + 1] <> "区画."]
 ]

kiritori[{m - 49, m}](*mは総ステップ数*)
```