ABC024解説
A:動物園

A-問題概要

- 動物園の入場料の情報が与えられる
  - 子供1人 A 円
  - 大人1人 B 円
  - K人以上の団体は1人当たりC円引
- 子供S人、大人T人の団体が支払わなければならない入場料はいくらか求めよ

A-問題概要

- 団体の人数がK以上かK未満かで場合分けして計算する

- 人数がK以上の時
  - 入場料＝A*S + B*T – C*(S+T)
- 人数がK未満の時
  - 入場料＝A*S + B*T

A-注意

- この手の問題は「以上」か「超える」か注意しましょう

- 実装時に ＜ と ≦ を間違えてWrongAnswerというのはもったいないです

ABC024解説
B:自動ドア

B-問題概要

- 前を人が通りかかるとT秒開き続ける自動ドアがある
- 開いている途中に新たに人が通りかかったら開く時間がそこからT秒後まで延長される
- N人がドアの前を通りかかった時刻が与えられる
- ドアが開いていた合計秒数を求めよ

B-部分点解法

- N ≦ 10^5
- T ≦ 100
- 時刻 ≦ 10^6

B-部分点解法

- N ≦ 10^5
- T ≦ 100
- 時刻 ≦ 10^6

- 各時刻についてその時刻にドアが開いていたか閉まっていたか覚えておく

B-部分点解法

- N ≦ 10^5
- T ≦ 100
- 時刻 ≦ 10^6

- 各時刻についてその時刻にドアが開いていたか閉まっていたか覚えておく
- 人が通りかかると、その時刻からT秒間のドアは開き続ける

B-部分点実装

- open[i] = 時刻iにドアが開いているかどうか

B-部分点実装

- open[i] ＝ 時刻iにドアが開いているかどうか

- 時刻aに人が通りかかった
  - open[a], open[a+1], … , open[a + t – 1]をTrueにする

B-部分点実装

- open[i] = 時刻iにドアが開いているかどうか

- 時刻aに人が通りかかった
  - open[a], open[a+1], … , open[a + t – 1]をTrueにする

- 最後にopen[i] = Trueであるiを数える

B-部分点実装

- open[i] = 時刻iにドアが開いているかどうか

- 時刻aに人が通りかかった
  - open[a], open[a+1], … , open[a + t − 1]をTrueにする

- 最後にopen[i] = Trueであるiを数える
- O(NT) 50点獲得

B-満点解法

- close[i] = 時刻iの時点でドアが何秒後に閉まる予定か？

B-満点解法

- close[i] = 時刻iの時点でドアが何秒後に閉まる予定か？

- 時刻aに人が通りかかった
  - close[a] = T

B-満点解法

- close[i] ＝ 時刻iの時点でドアが何秒後に閉まる予定か？

- 時刻aに人が通りかかった
  - close[a] ＝ T
- 時刻aに人が通らなかった
  - close[a] ＝ max(close[a-1] – 1, 0)

B-満点解法

- close[i] = 時刻iの時点でドアが何秒後に閉まる予定か？

- 時刻aに人が通りかかった
  - close[a] = T
- 時刻aに人が通らなかった
  - close[a] = max(close[a-1] – 1, 0)

- 最後にclose[i] > 0であるiを数える

B-満点解法

- close[i] = 時刻iの時点でドアが何秒後に閉まる予定か？

- 時刻aに人が通りかかった
  - close[a] = T
- 時刻aに人が通らなかった
  - close[a] = max(close[a-1] – 1, 0)

- 最後にclose[i] > 0であるiを数える
- O(時刻)

ABC024解説
C:民族大移動

C-問題概要

- 1～Nの番号が付けられた街がある
- i(1 ≦ i ≦ D)日目にはL[i] ≦ 街番号 ≦ R[i]となる街の間でのみ行き来が可能である。
- 民族i(1 ≦ i ≦ K)は街S[i]から街T[i]に移動したい
- 各民族はできるだけ早く目的地に到着するように移動する
- 各民族が目的地に着く日を求めよ

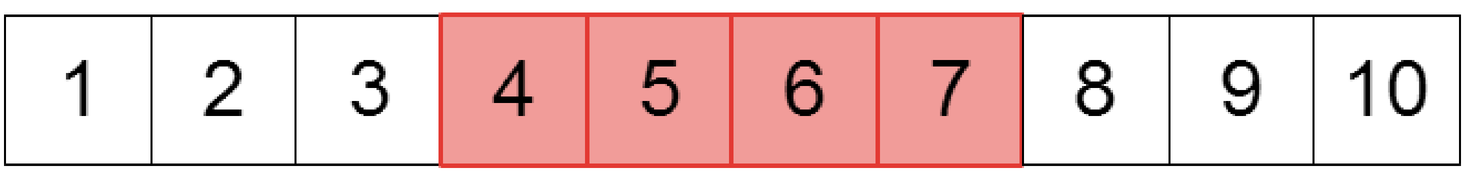
C-考察

- i(1 ≦ i ≦ D)日目にはL[i] ≦ 街番号 ≦ R[i]となる街の間でのみ行き来が可能である。
- この移動制限の性質から、街は以下のように一直線に並んでいると考えても良い

1	2	3	4	5	6	7	8	9	10

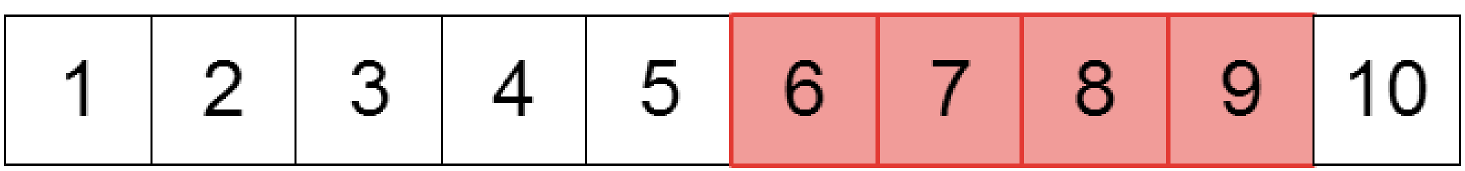
C-考察

- i(1 ≦ i ≦ D)日目にはL[i] ≦ 街番号 ≦ R[i]となる街の間でのみ行き来が可能である。
- この移動制限の性質から、街は以下のように一直線に並んでいると考えても良い
- このうち一区間が移動可能と捉えることが出来る (L[i] = 4, R[i] = 7の例)



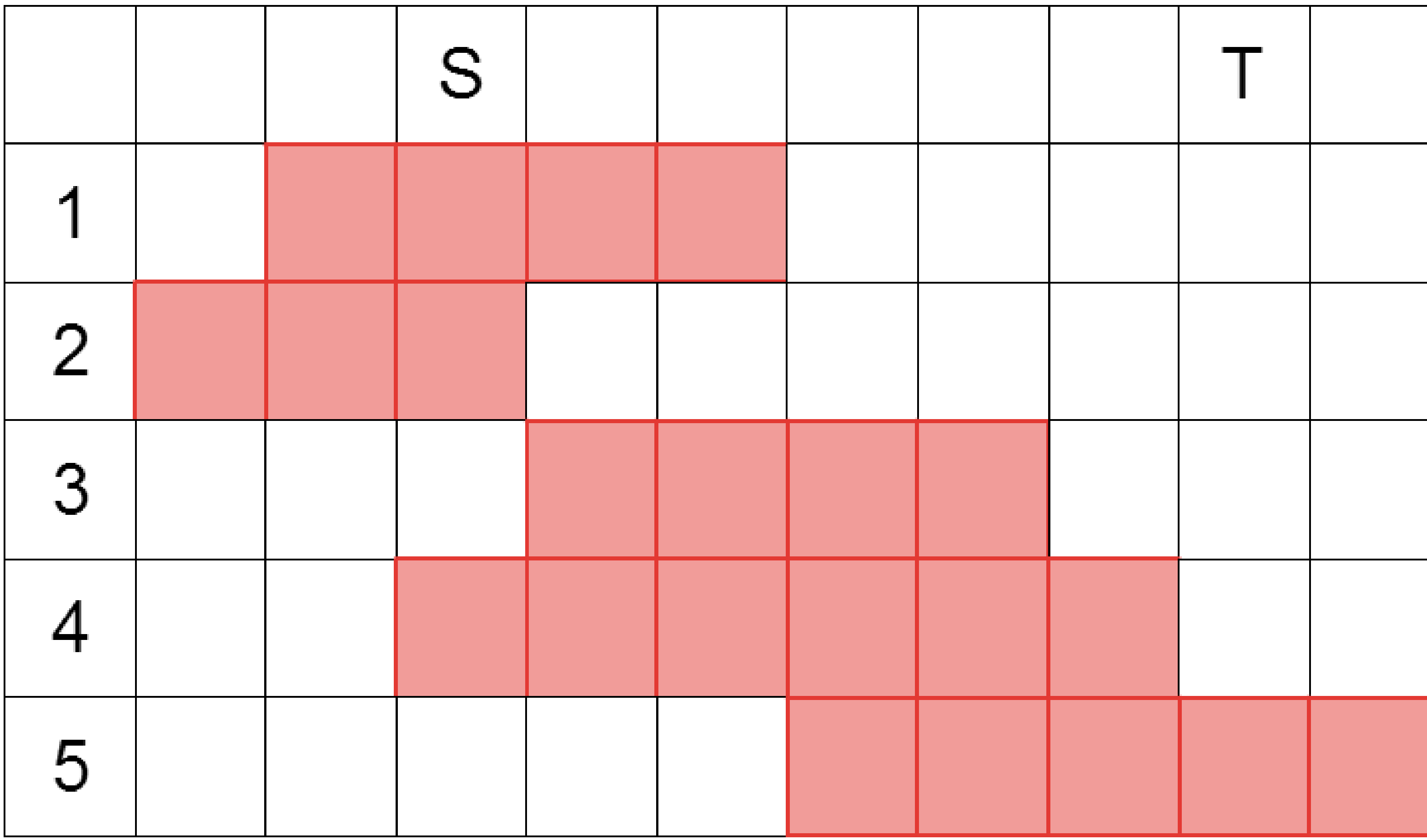
C-考察

- i(1 ≦ i ≦ D)日目にはL[i] ≦ 街番号 ≦ R[i]となる街の間でのみ行き来が可能である。
- この移動制限の性質から、街は以下のように一直線に並んでいると考えても良い
- このうち一区間が移動可能と捉えることが出来る (L[i] = 6, R[i] = 9の例)



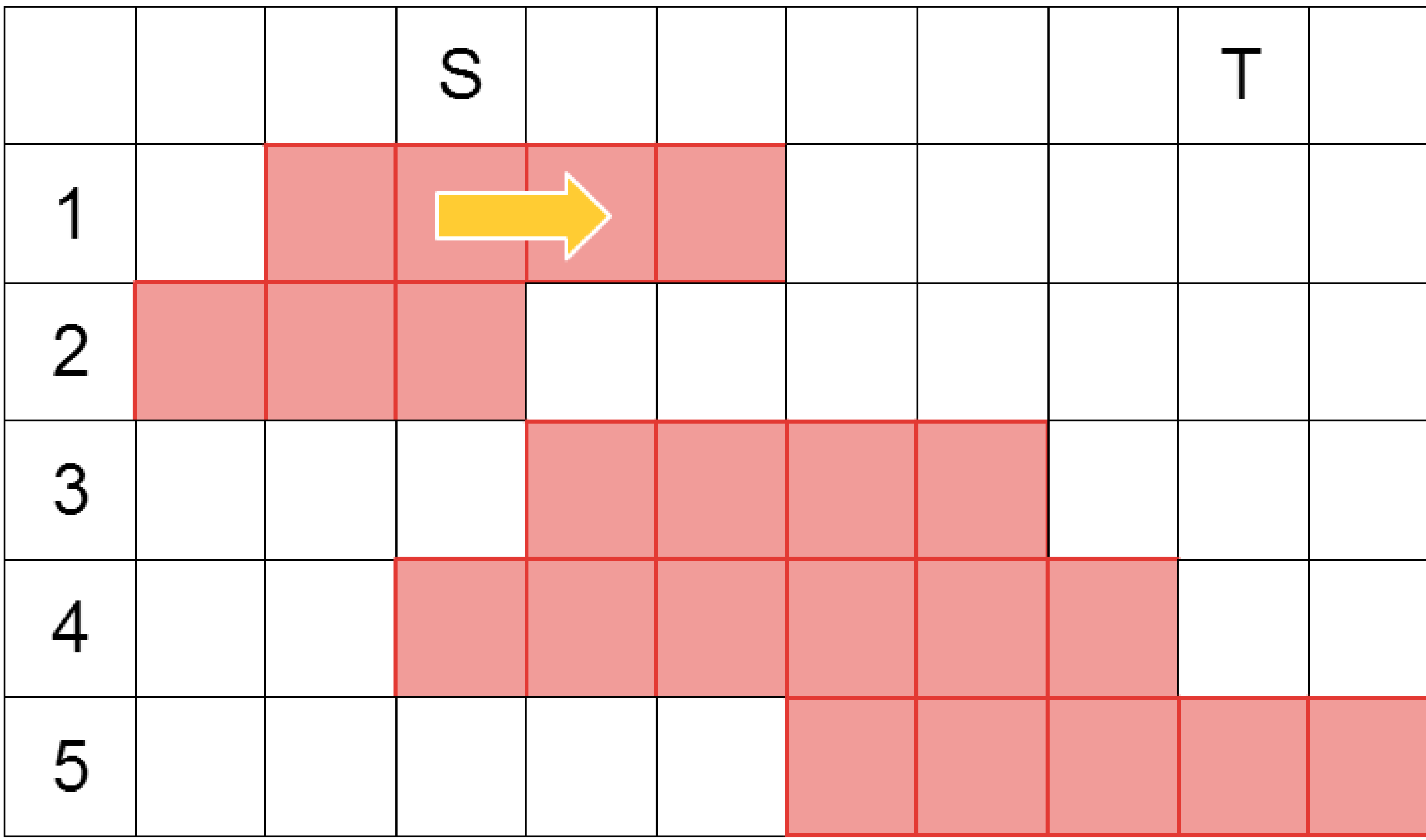
C-例

1つの民族について考える



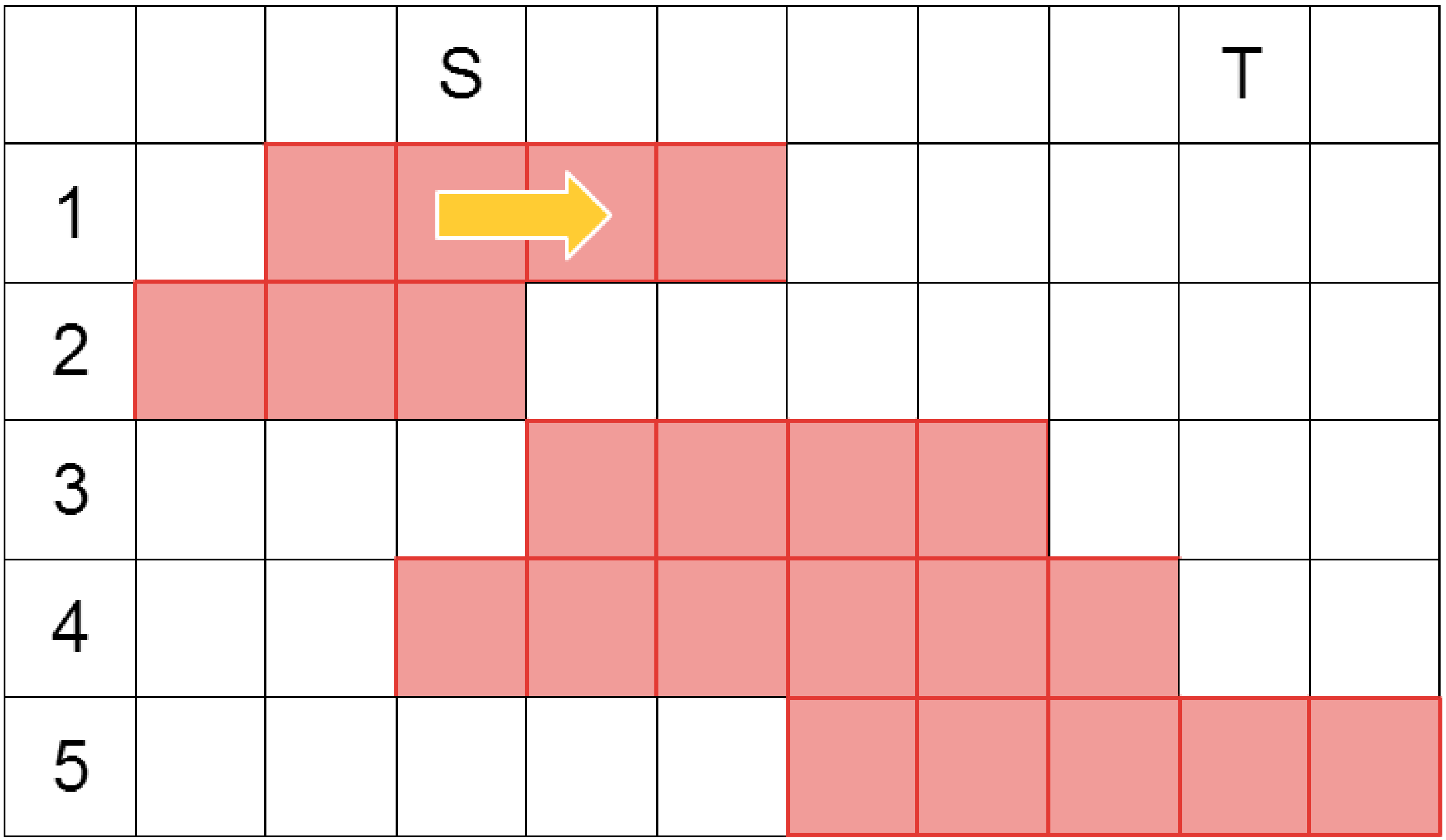
C-例

1日目



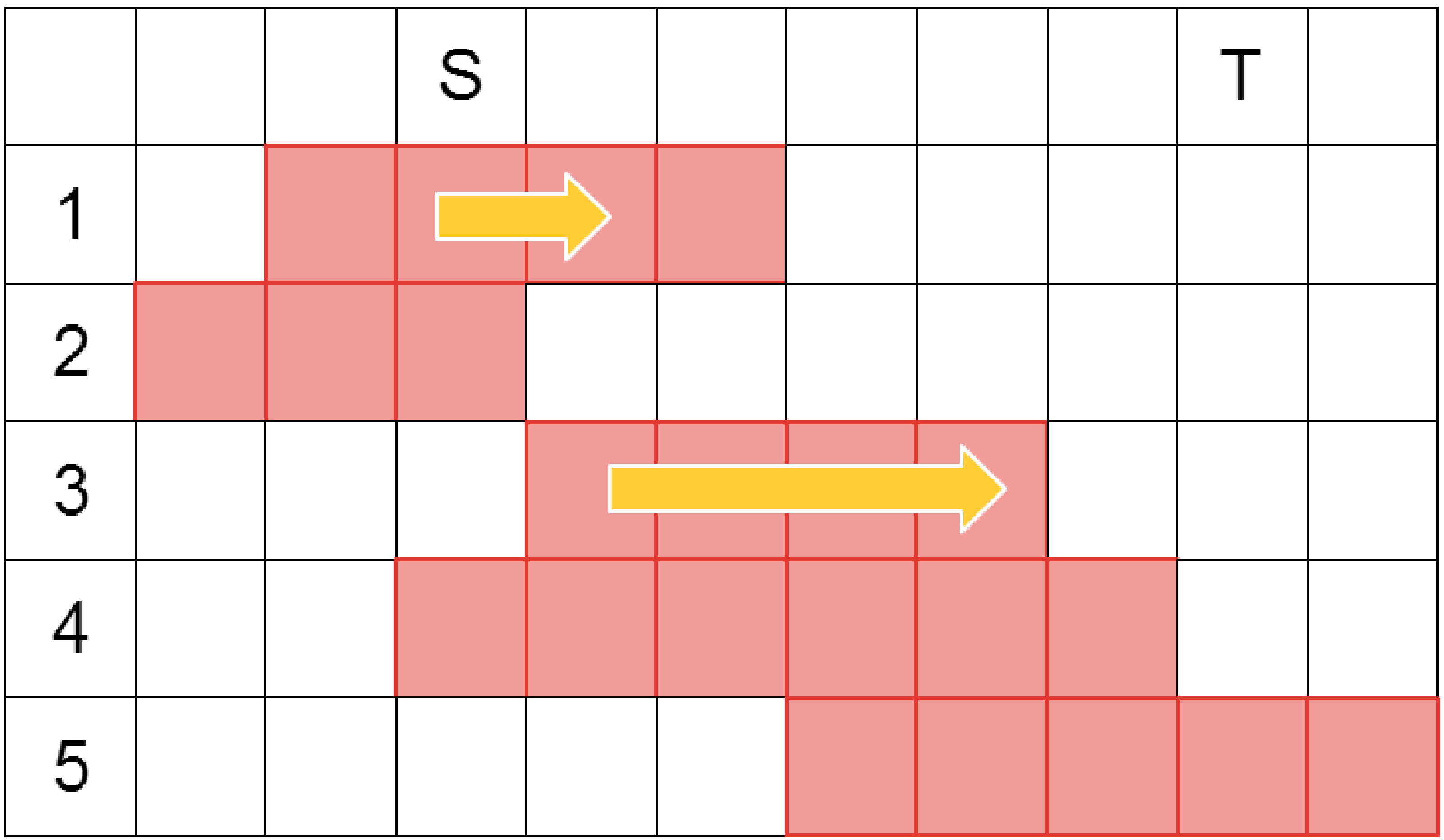
C-例

2日目:移動せず



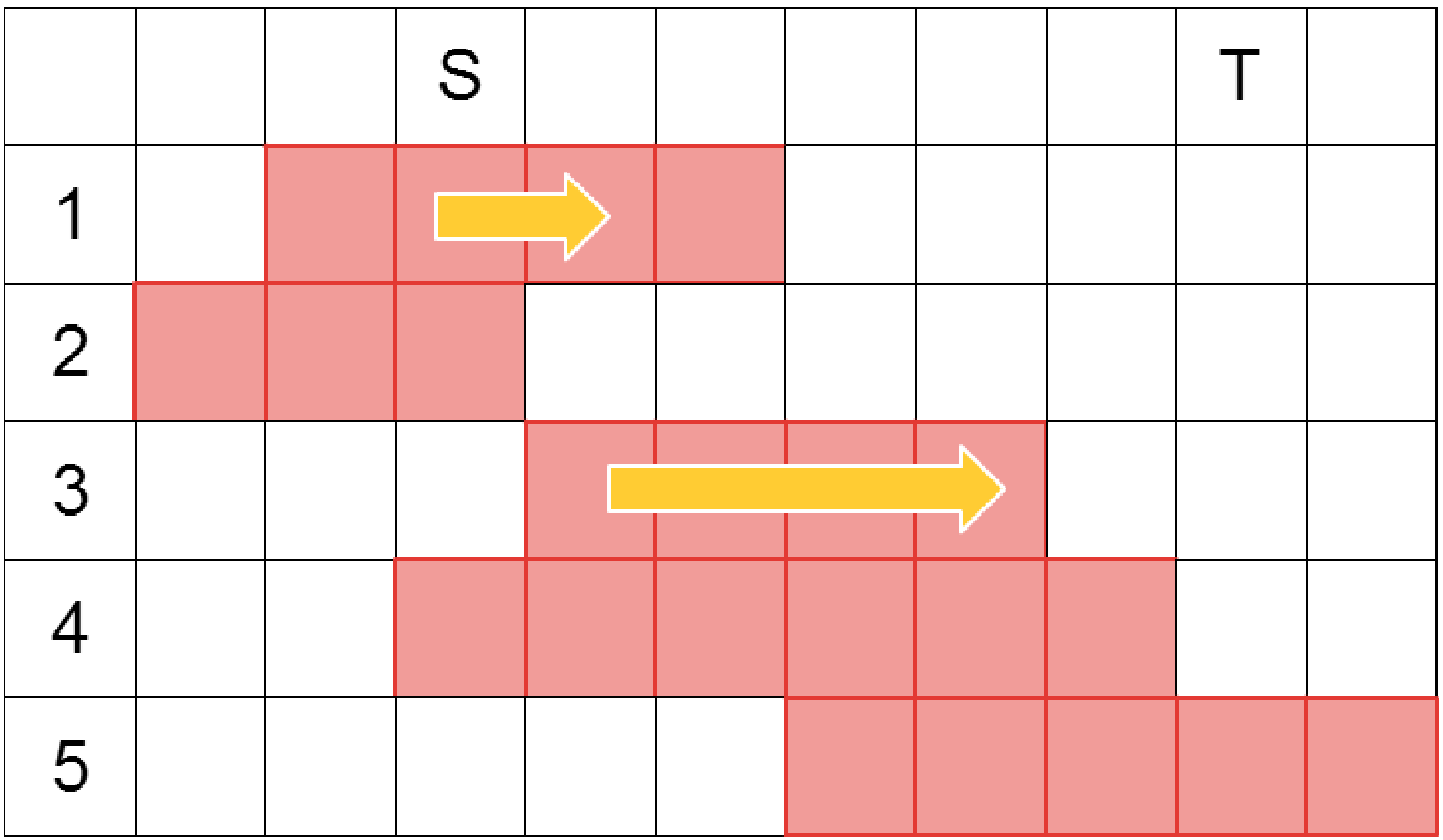
C-例

3日目



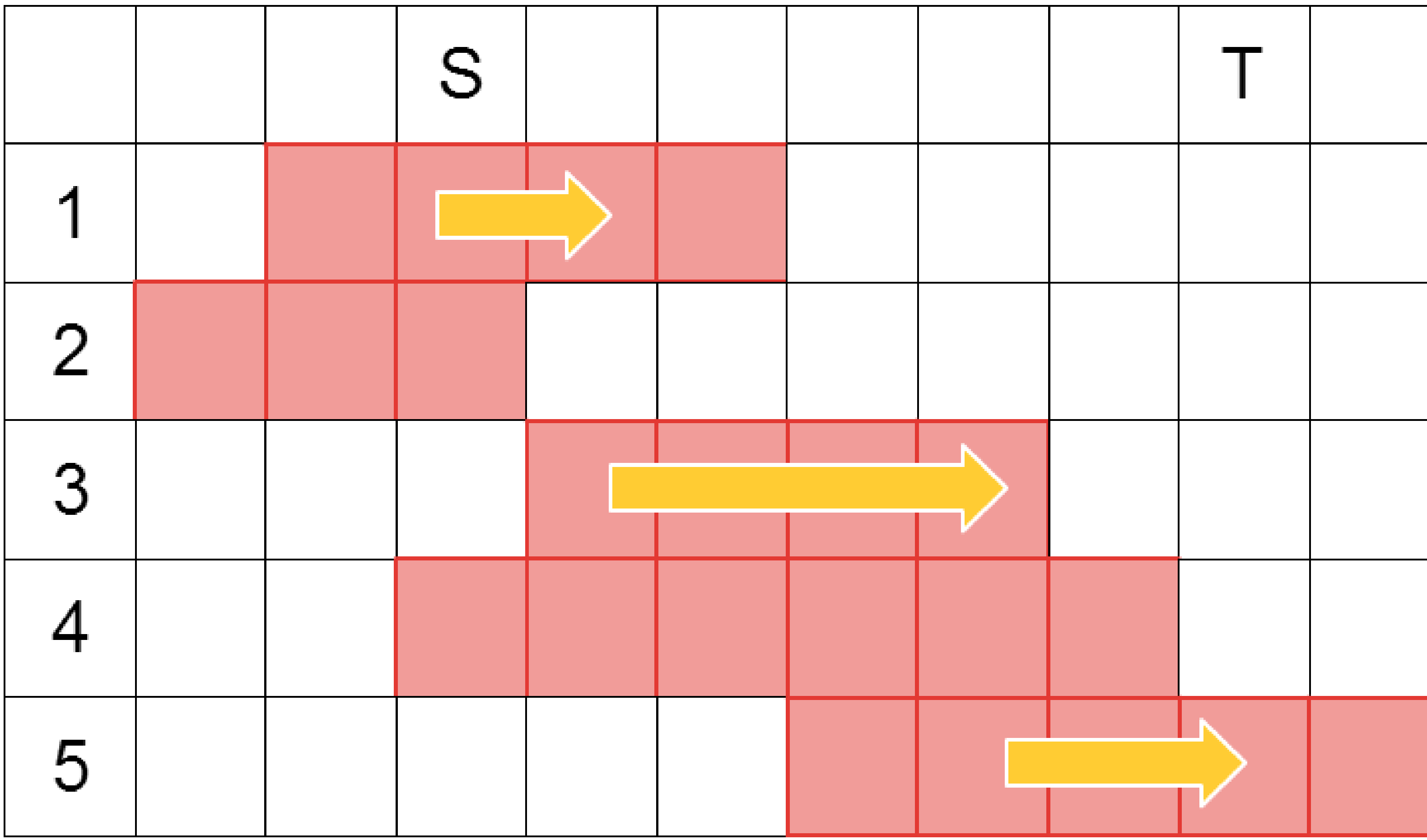
C-例

4日目:移動せず



C-例

5日目:目的地に到着



C-考察

- 街が一直線に並んでいるので、民族はSとTの間のすべての街を通過しなければならない

C-考察

- 街が一直線に並んでいるので、民族はSとTの間のすべての街を通過しなければならない
- 民族は近づく方向にのみ進めば良い
  - 再び同じ所に帰ってくるならば、その間移動しなくてもよい

C-考察

- 街が一直線に並んでいるので、民族はSとTの間のすべての街を通過しなければならない
- 民族は近づく方向にのみ進めば良い
  - 再び同じ所に帰ってくるならば、その間移動しなくてもよい
- 近づけるだけ近づいたほうが良い
  - 進みすぎたせいで、他の移動方法より近づけないということはない

    →

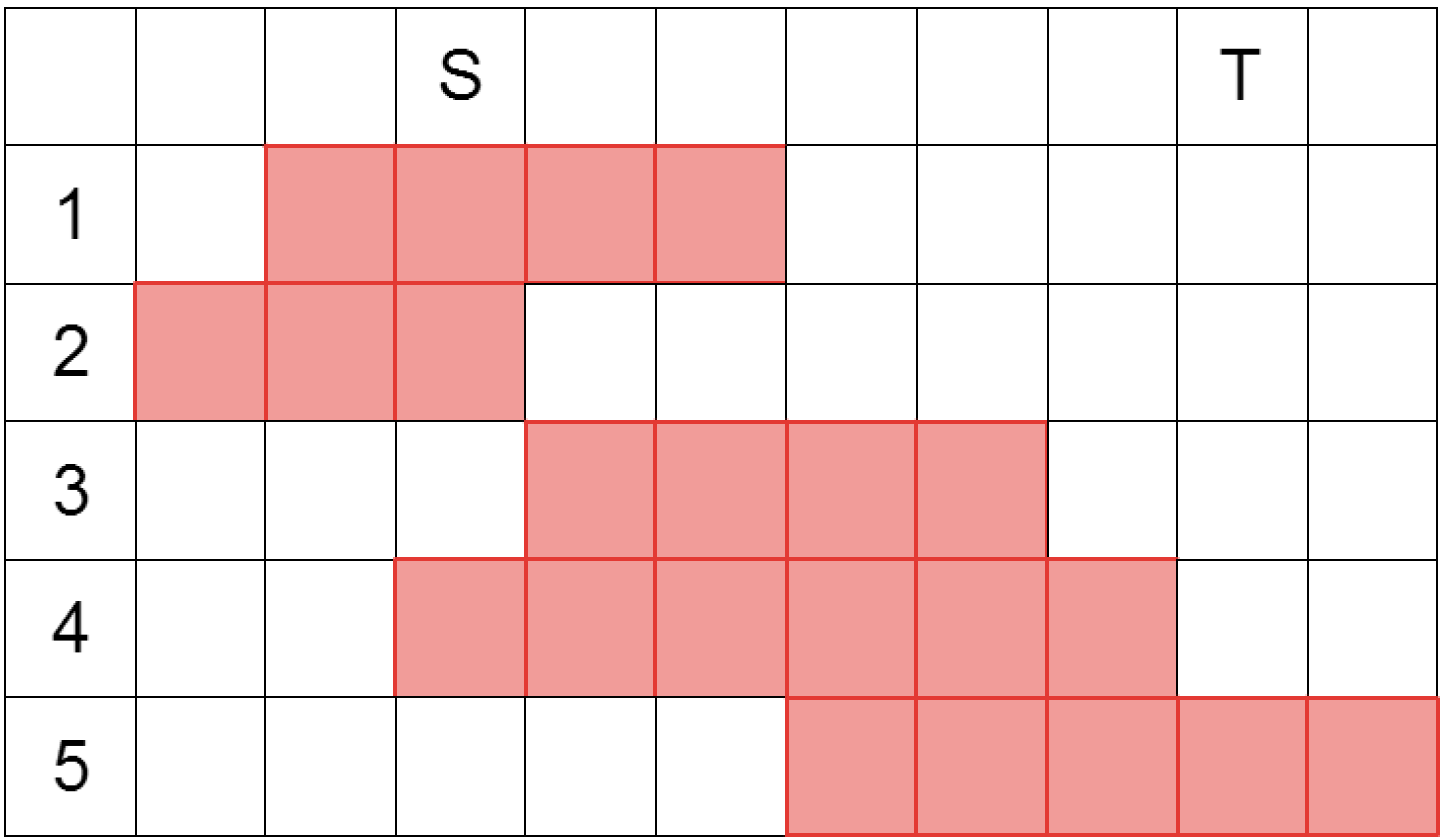
C-考察

- 街が一直線に並んでいるので、民族はSとTの間のすべての街を通過しなければならない
- 民族は近づく方向にのみ進めば良い
  - 再び同じ所に帰ってくるならば、その間移動しなくてもよい
- 近づけるだけ近づいたほうが良い
  - 進みすぎたせいで、他の移動方法より近づけないということはない

  →貪欲法

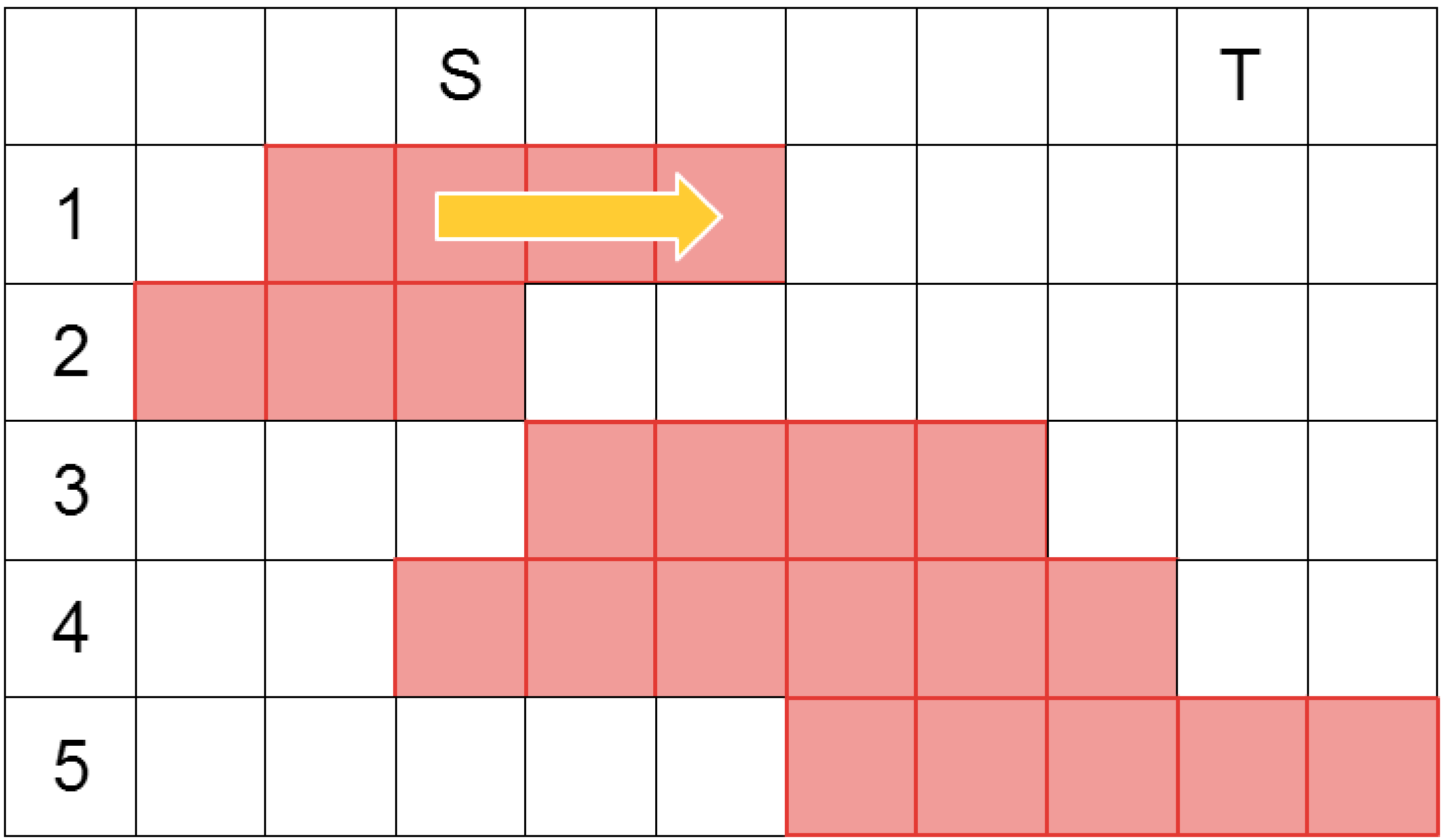
C-貪欲法

- 先程の例



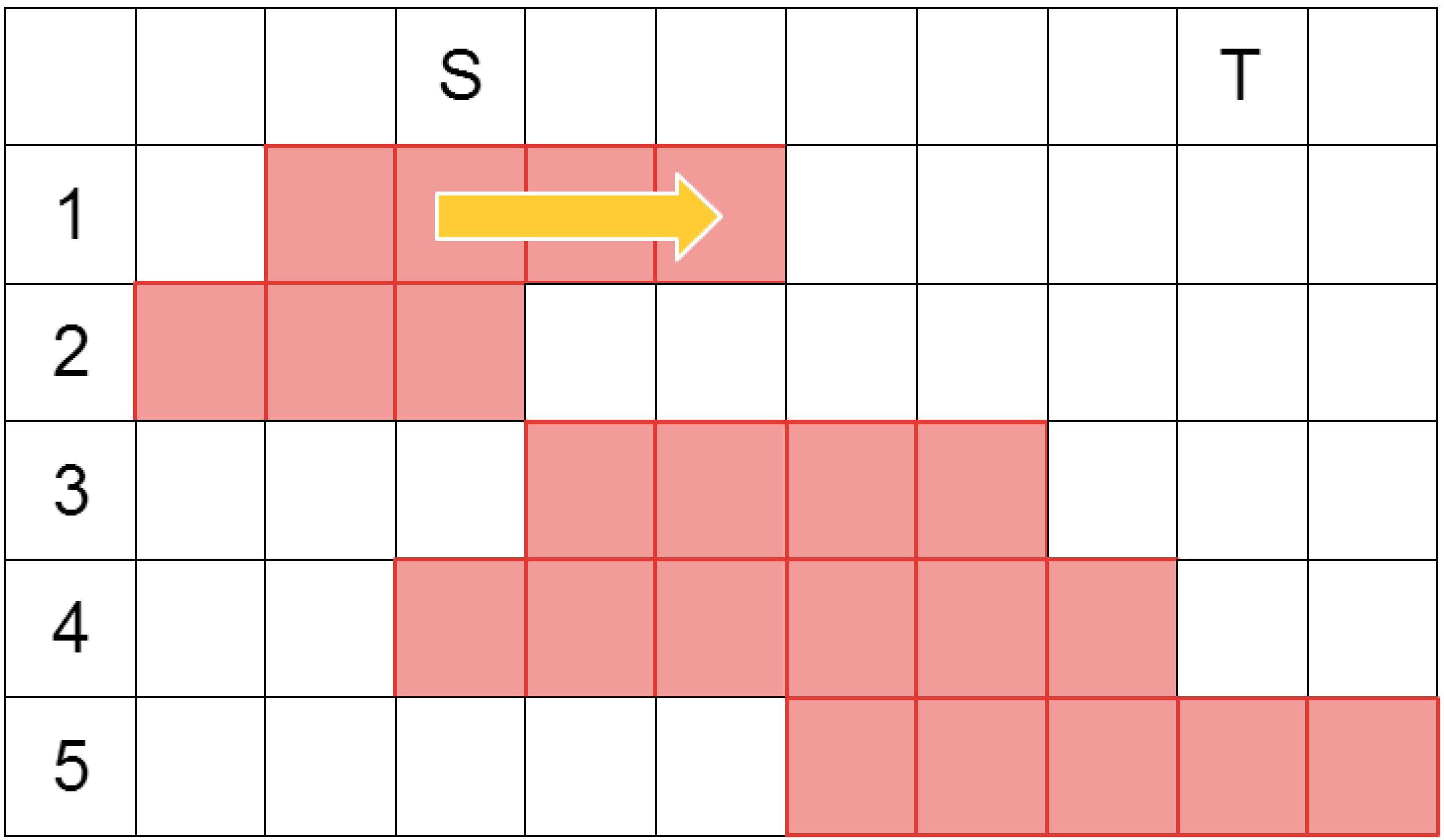
C-貪欲法

- 1日目:できるだけTに近づく



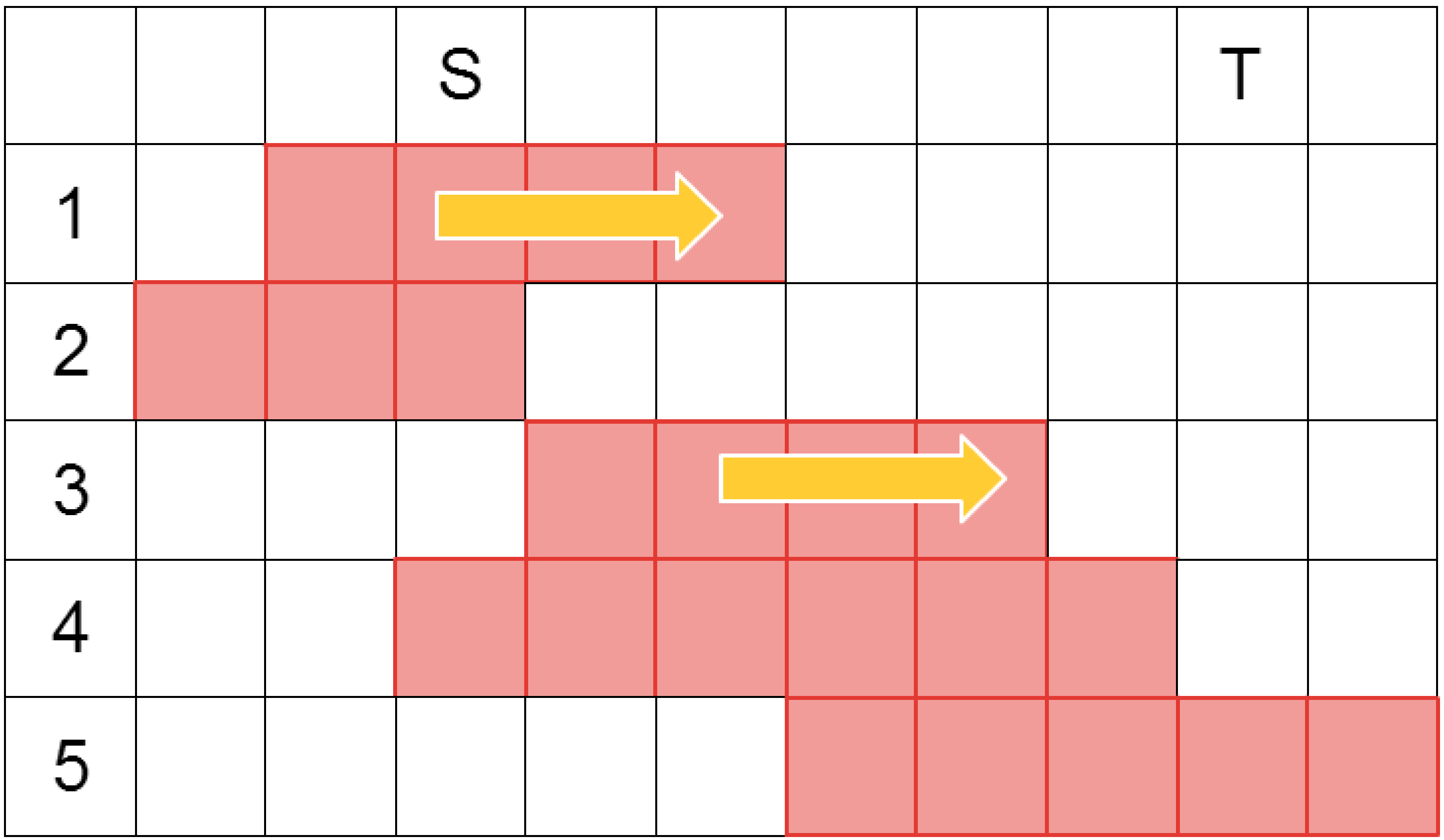
C-貪欲法

- 2日目：移動できない



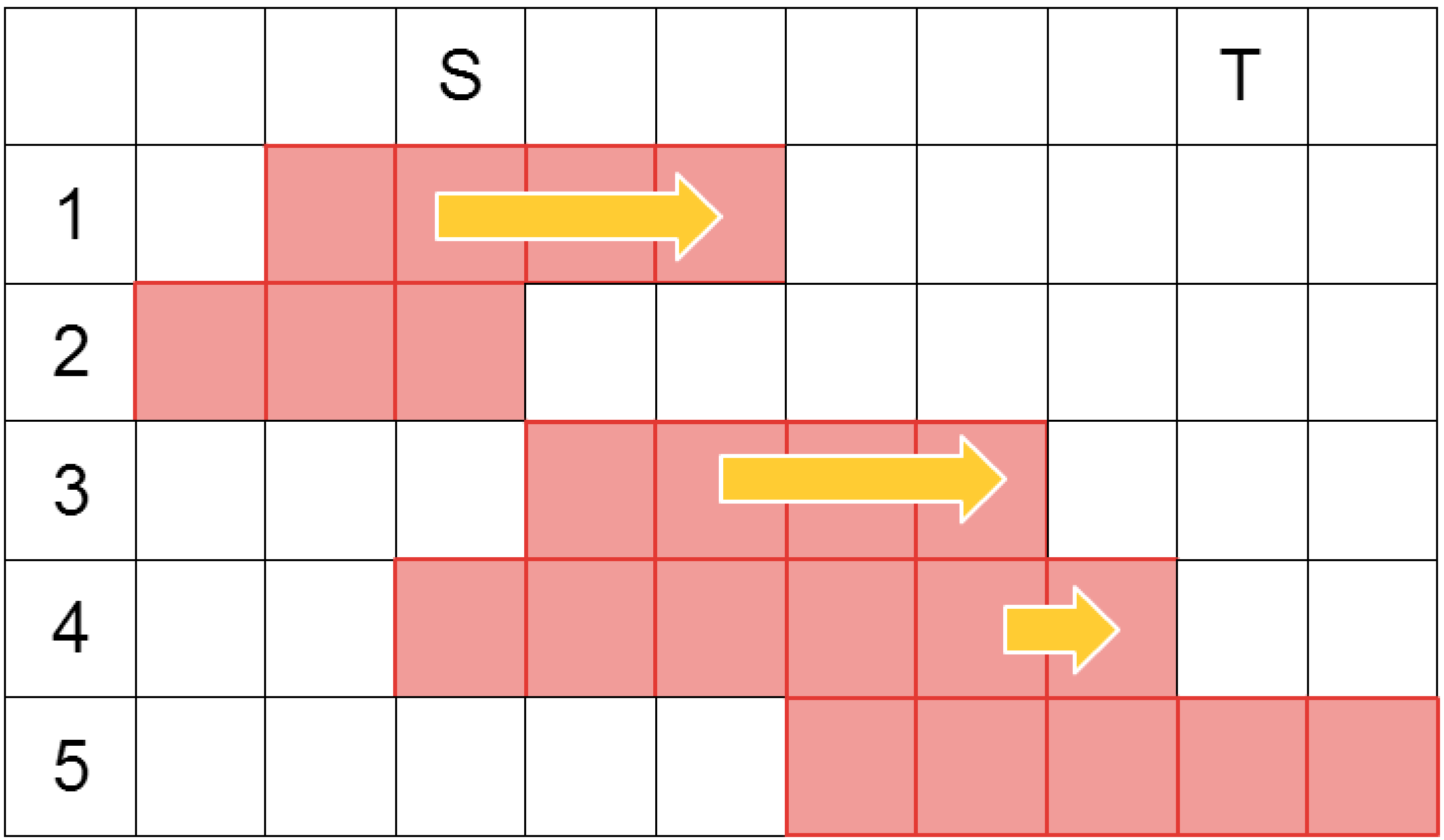
C-貪欲法

- 3日目：できるだけTに近づく



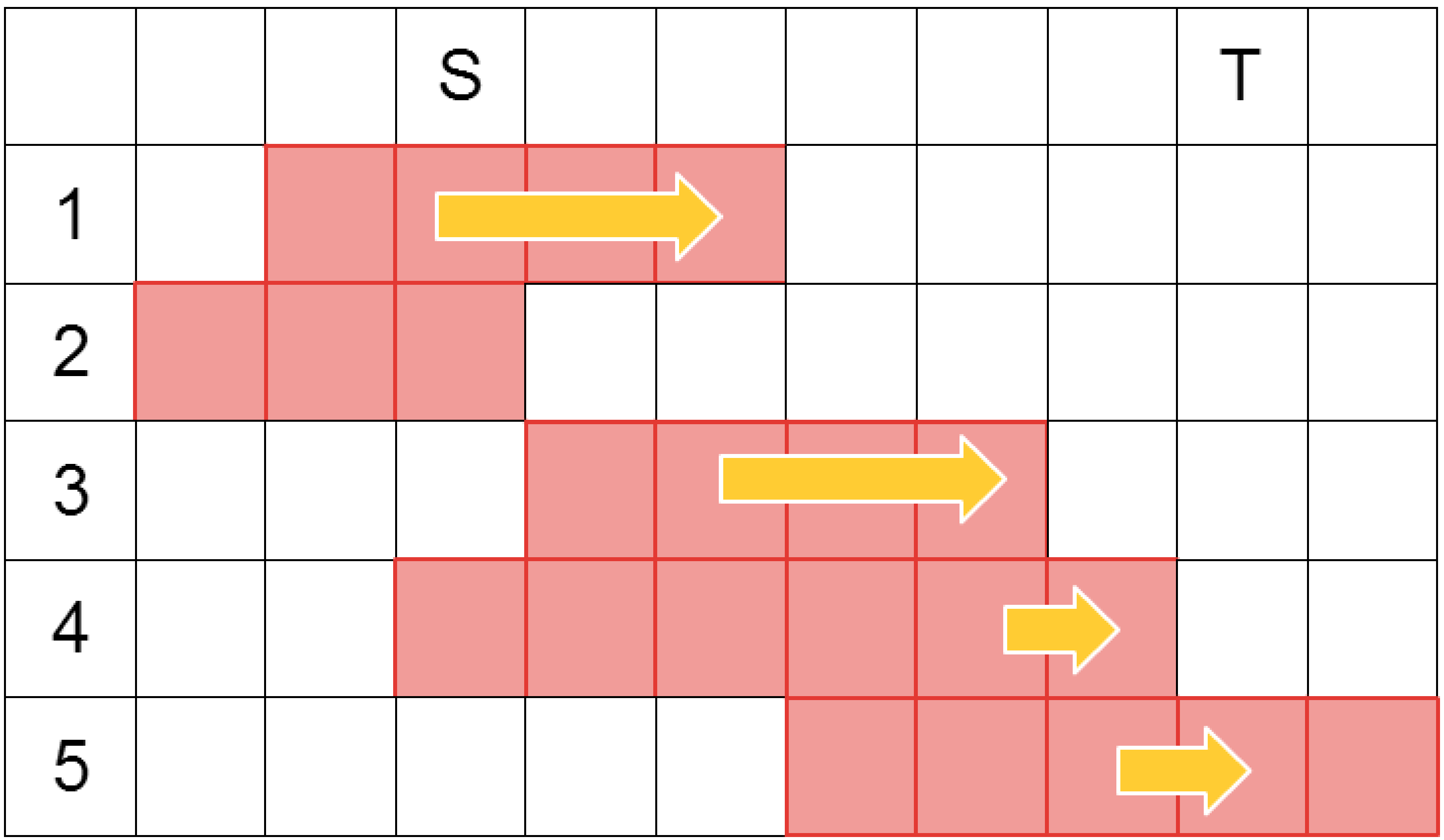
C-貪欲法

- 4日目：できるだけTに近づく



C-貪欲法

- 5日目：できるだけTに近づく　目的地到着



C-想定解法

- 各民族について別々に到達日を求める

- 1日目から順番にTにできるだけ近づくように移動する
  - 始めてTに到着する日が答え

- O(DK)

ABC024解説
D:動的計画法

D-問題概要

- 10^8 × 10^8の方眼紙の上で有名な動的計画法の問題を解く
  - 左下のマスから開始して右、上に1マス動くというのを繰り返して各マスに到達する方法の個数をMOD 1,000,000,007で求めよ
  - その答えを各マスに書き込む
- 左からxマス下からyマスの位置を(x, y) とする
- マス(r, c), (r, c + 1), (r + 1, c)に書かれている整数を元にr, cを求めよ

D-有名事実

- (r, c)に書かれている値は $_{c+r}C_c$ であある
  - (0, 0) から (r, c)に到達するまでに r+c回の移動を行う
  - そのうちr回が右向きの移動
  - r+c回のうちどのr回に右向きの移動を行うか数えれば良い
  - $_{c+r}C_c$

D-問題概要言い換え

- $_{c+r}C_c$ 、$_{c+r+1}C_c$ 、$_{c+r+1}C_{c+1}$ が与えられる
  - ただし1,000,000,007で割った余り)
- r, cを求めよ
  - 0 ≦ r, c ＜ 99,999,999が保証されている

D-問題概要言い換え

- ${}_{c+r}C_c$ 、${}_{c+r+1}C_c$ 、${}_{c+r+1}C_{c+1}$ が与えられる
  - ただし1,000,000,007で割った余り)
- r, cを求めよ
  - 0 ≦ r, c ＜ 99,999,999が保証されている
- k ＝ r+cとすると

D-問題概要言い換え

- ${}_{c+r}C_{c}$ 、${}_{c+r+1}C_{c}$ 、${}_{c+r+1}C_{c+1}$ が与えられる
  - ただし1,000,000,007で割った余り)
- r, cを求めよ
  - 0 ≦ r, c ＜ 99,999,999が保証されている
- k ＝ r+cとすると
- ${}_{k}C_{c}$、${}_{k+1}C_{c}$ 、${}_{k+1}C_{c+1}$ が与えられる
- k-c, cを求めよ

D-とりあえず

「1,000,000,007で割った余り」という条件を忘れて、実際の値が与えられると仮定して問題を解いてみよう

D-結論から言うと

- 結論を先に言うと

  $$ {}_{c+r}C_{c} \quad {}_{c+r+1}C_{c} \quad {}_{c+r+1}C_{c+1} $$

  が与えられた時、これらと四則演算によってr, cが求まる

- 適切な四則演算を導く方法はいくらでもある
- 今回はその一例を挙げる

D-コンビネーション

- コンビネーションは階乗を用いて以下のように定義される

$$ {}_kC_c = \frac{k!}{c!(k-c)!} $$

D-コンビネーション

- よって与えられた3つの値は以下のように書ける

$$ {}_{k}C_{c} = \frac{k!}{c!(k-c)!} $$

$$ {}_{k+1}C_{c} = \frac{(k+1)!}{c!(k+1-c)!} $$

$$ {}_{k+1}C_{c+1} = \frac{(k+1)!}{(c+1)!(k-c)!} $$

D-比を取る

- 以下の式が成り立つ

$$\frac{{}_{k}C_{c}}{{}_{k+1}C_{c}} = \frac{k+1-c}{k+1}$$

$$\frac{{}_{k+1}C_{c+1}}{{}_{k}C_{c}} = \frac{k+1}{c+1}$$

D-式変形

$$\frac{{}_{k}C_{c}}{{}_{k+1}C_{c}} = \frac{k+1-c}{k+1}$$

- 両辺を1から引く

$$1 - \frac{{}_{k}C_{c}}{{}_{k+1}C_{c}} = \frac{c}{k+1}$$

D-変数を減らす

$$\frac{{}_{k+1}C_{c+1}}{{}_{k}C_{c}} = \frac{k+1}{c+1}$$ 、 $$1 - \frac{{}_{k}C_{c}}{{}_{k+1}C_{c}} = \frac{c}{k+1}$$

- これらを掛けると

$$\left(1 - \frac{{}_{k}C_{c}}{{}_{k+1}C_{c}}\right)\left(\frac{{}_{k+1}C_{c+1}}{{}_{k}C_{c}}\right) = \frac{c}{c+1}$$

D-整理して

$$\left(1-\frac{A}{B}\right)\left(\frac{C}{A}\right)=\frac{c}{c+1}$$

D-続きは割愛

$\dfrac{c}{c+1}$ から c を求める方法は容易なので割愛

D-結果

$$c = \frac{BA - BC + AC}{BC - AC}$$

D-結果

$$c = \frac{BA - BC + AC}{BC - AC}$$

- rとcの対称性よりrも同様に求めることが出来る

$$r = \frac{CA - BC + AB}{BC - AB}$$

D-注意

$$c = \frac{BA - BC + AC}{BC - AC} \quad r = \frac{CA - BC + AB}{BC - AB}$$

- r, cを求める式は四則演算のみで構成されている。
- 途中で1,000,000,007の倍数で割ることがなければMOD 1,000,000,007 しても結果は変わらない

D-MODでの演算

- 1,000,000,007は素数なのでこれでMODをとっても四則演算が実行できる
- 以下 法を1,000,000,007として、$A \equiv a$, $B \equiv b$, $C \equiv c$とする
- 足し算、引き算
  - $A \pm B = C$ならば $a \pm b \equiv c$
- 掛け算
  - $A \times B = C$ならば $a \times b \equiv c$

D-MODでの演算

- 以下 法を1,000,000,007として、A ≡ a, B ≡ b, C ≡ cとする
- 割り算
  - B ≡ 0でなくA ÷ B = Cならば

    a × (bの逆元) ≡ c
- 逆元とは
  - Xの逆元とは X × Y ≡ 1となるYのこと
  - X≠0 で 法(ここでは1,000,000,007)が素数ならば逆元が唯一つ存在する
  - →常に割り算が出来る

D-逆元の求め方

- 法が素数Pのときの逆元は容易に求めることが出来る
- フェルマーの小定理より以下の関係式が導ける(証明割愛)

$$X^{-1} \equiv X^{P-2} (MODP)$$

- 今回ならばある値を 1,000,000,005乗すればその逆元が求まる

D-高速に累乗を求める

- 1,000,000,005乗はどのように求めればよいだろうか？
  - 愚直に掛け算を繰り返すとTLEしてしまう
- X^Nを高速に求める以下の様なアルゴリズムがある
  - $X^1, X^2, X^4, X^8, \ldots$ を予め求める
  - Nを2進数表記することを考えると上記の予め求めた値の積でX^Nを求めることが出来る
  - Nの2進数表記はO(logN)桁なので計算量もO(logN)
  - 速い

D-想定解法

$$c = \frac{BA - BC + AC}{BC - AC} \quad r = \frac{CA - BC + AB}{BC - AB}$$

- この計算をMOD 1,000,000,007の中で行う
  - 割り算は逆元を利用する
- 答えとなるrとcのMOD 1,000,000,007が求まる
- 0 ≦ r, c ＜ 99,999,999なのでr, cは一意に定まる
- それが出力する答え