M1801 - Two-cube Calendar

Alex Tung
alex20030190@yahoo.com.hk

15 February 2018

Step 1: Storing cube digits

- Easiest way: use (Boolean) arrays $A[]$ and $B[]$ to mark appearances of digits.

Step 1: Storing cube digits

- Easiest way: use (Boolean) arrays $A[]$ and $B[]$ to mark appearances of digits.
- For example, if first two lines of input are

  1 1 2 2 6 6

  0 0 0 0 0 9,

  Then mark $A[1] = A[2] = A[6] = 1$ and $B[0] = B[9] = 1$.

Step 1: Storing cube digits

- Easiest way: use (Boolean) arrays $A[]$ and $B[]$ to mark appearances of digits.
- For example, if first two lines of input are

  1 1 2 2 6 6

  0 0 0 0 0 9,

  Then mark $A[1] = A[2] = A[6] = 1$ and $B[0] = B[9] = 1$.
- Afterwards,
  - If either $A[6]$ or $A[9]$ equals 1, set both as 1.
  - If either $B[6]$ or $B[9]$ equals 1, set both as 1.

Step 2: Answering queries

- Two ways to read the queries: as integers, or as strings.

Step 2: Answering queries

- Two ways to read the queries: as integers, or as strings.
- As integer: say the input is K.
  Its tens digit is $K/10$ and its unit digit is $K\%10$.

Step 2: Answering queries

- Two ways to read the queries: as integers, or as strings.
- As integer: say the input is K.
  Its tens digit is $K/10$ and its unit digit is $K\%10$.
- As string: say the input is $str[]$.
  Its tens digit is ($str[0]$ - '0') and its unit digit is ($str[1]$ - '0').

Step 2: Answering queries

- Two ways to read the queries: as integers, or as strings.
- As integer: say the input is K.
  Its tens digit is $K/10$ and its unit digit is $K\%10$.
- As string: say the input is $str[]$.
  Its tens digit is ($str[0]$ - '0') and its unit digit is ($str[1]$ - '0').
- The answer is `Yes` if and only if:
  - $A[tens] = 1$ and $B[unit] = 1$, **OR**
  - $A[unit] = 1$ and $B[tens] = 1$.

Comments

- Some contestants failed to handle digits 6 and 9 correctly.

Comments

- Some contestants failed to handle digits 6 and 9 correctly.
- Suggestion: in contests without instant feedback (or if incorrect submission leads to penalty), test boundary cases.

Comments

- Some contestants failed to handle digits 6 and 9 correctly.
- Suggestion: in contests without instant feedback (or if incorrect submission leads to penalty), test boundary cases.
- Try dice with:
  - Both 6 and 9
  - 6 only
  - 9 only

M1802 Origami Hearts

Steven Lau

Area of the heart

- A correct rectangular paper is W * W/4 unit square
- area of the rectangular paper = $W^2/4$
- then area of the heart = $W^2/4 * 5 / 8$
  (by measuring with a ruler, counting etc)

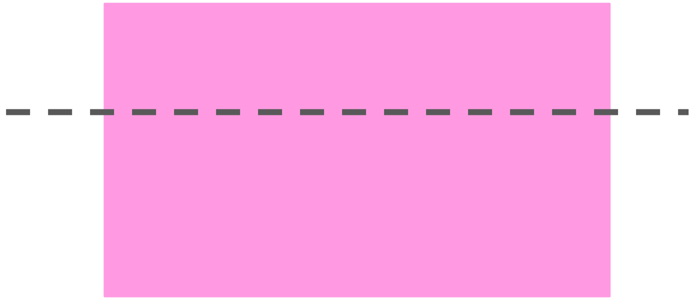
Subtask 1 W = H

- Divide the square paper into four stripes
- Each one's area = W * W/4, which can be folded into a heart of area W * W/4 * 5 / 8
- Guaranteed at least one heart can be made means T <= W * W/4 * 5 / 8, i.e. we can always fold four hearts
- final answer = 4 * W * W/4 * 5 / 8

Cutting paper - two situations only

First, rotate the paper so that W > H

1. If W/4 < H, cut horizontally;
   area increment W * W/4 * 5 / 8;
   repeat with W' = W, H' = H - W/4



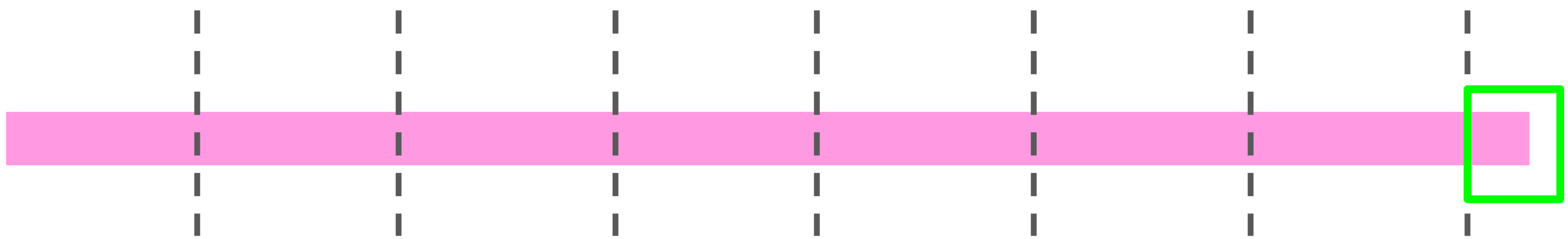
Cutting paper - two situations only

2. If $W/4 > H$, cut vertically;
   area increment $4H * H * 5 / 8$;
   repeat with $W' = W - 4H$, $H' = H$

- Terminate when heart area < T
- Each iteration reduces W or H a little.
  Time complexity = $O(W + H)$ TLE

Cutting paper - speed up

- For long stripes, cut a whole bunch at once



- 7 identical correct ratio papers, each with area $4H^2$
- area increment $7 * 4H^2 * 5 / 8$
- Repeat the process with $W' = W \bmod 4H$, $H' = H$

means remainder of $W \div (4H)$

Time complexity

- In each iteration, the paper area is reduced into smaller than half of its size
- O(log(WH))

M1803 I love you I love you

Steven Lau

Subtask 1, 2, 3

- Expected solution: repeat **string.erase** until done
- Need to carefully handle many corner cases mixed
  - "HKOI love your sister." (stick to other words)
  - "I love you very much." (beginning of sentence)
  - "Alice I love you very much." (middle of sentence)
  - "Alice I love you." (end of sentence)
  - "Alice I love I love you xd." (multiple deletion)
  - "I love you." (delete entire sentence)
  - x(

Simplification

- This kind of string processing problems can be extremely annoying if you do not simplify the complexity
- Observation: sentences are independent with each other
- Idea: solve the problem sentence by sentence
- greatly simplified fullstop handling

Simplification

- Organize the whole email into an **array of array of string**
- `Alice I love you. Believe me. I love you.`
- ```
  [ ["Alice", "I", "love", "you"]
  , ["Believe", "me"]
  , ["I", "love", "you"]
  ]
  ```
- No more spaces, no more fullstops :)
  ```

Solving a sentence

- So each sentence is just an array of string
- Time to delete "I love you"
- We need to efficiently handle a lengthy

  `I love I I love you love you you`
- looks like... `((()))` ?
- let's use stack! (learn more in Data structures (I))

Stack

- When encountered "I", "love" or "you" that is a continuation of the top of stack, push into stack
  - continuation:
    I -> I
    I -> love
    love -> I
    love -> you
  - if top of stack is ["I", "love", "you"], pop the three words
- Otherwise, output the whole stack and clear the stack

I love I I love you love you love

[]

["I"]

["I","love"]

["I","love","I"]

["I","love","I","I"]

["I","love","I","I","love"]

["I","love","I",~~"I","love","you"~~] pop!

["I","love","I","love"]

["I","love",~~"I","love","you"~~] pop!

["I","love","love"] output!

[]

Output

- Eventually, you get
```
[ ["Alice"]
, ["Believe", "me"]
, []                <- empty sentence
]
```
- Simply output words separated by space and sentences separated by space and ends with fullstop. Remember to skip empty sentences.

- `Alice. Believe me.`

Time complexity

- Each word is pushed into the stack once and popped once
- O(Text size)

Common bug

- Forgot to clear the stack at the end

M1804 - Programmer's Dream

Alex Tung
alex20030190@yahoo.com.hk

15 February 2018

Subtask 1: $N = 2$

- Bob will move to 0 or N in one step.

Subtask 1: $N = 2$

- Bob will move to 0 or N in one step.
- Therefore the answer is just $\frac{p}{q}$.

Subtask 1: $N = 2$

- Bob will move to 0 or N in one step.
- Therefore the answer is just $\frac{p}{q}$.
- Don't forget to reduce the fraction!
  e.g. $Pr(2, 1, 2, 4) = \frac{2}{4}$, but output should be 1 2.

Subtask 2: $N \leq 3$

- For fixed N, p, q, let $f(X) := Pr(N, X, p, q)$.
- Let $L := \frac{p}{q}$ and $R := 1 - L$.

Subtask 2: $N \leq 3$

- For fixed N, p, q, let $f(X) := Pr(N, X, p, q)$.
- Let $L := \frac{p}{q}$ and $R := 1 - L$.
- Idea: try to find $f(0), f(1), \ldots, f(N)$ simutaneously.

Subtask 2: $N \le 3$

- For fixed N, p, q, let $f(X) := Pr(N, X, p, q)$.
- Let $L := \frac{p}{q}$ and $R := 1 - L$.
- Idea: try to find $f(0), f(1), \ldots, f(N)$ simutaneously.
- We know that:

$$\begin{cases} f(0) &= 1 \\ f(N) &= 0 \\ f(X) &= L \times f(X-1) + R \times f(X+1) \end{cases}$$

Subtask 2: $N \leq 3$

- For fixed N, p, q, let $f(X) := Pr(N, X, p, q)$.
- Let $L := \frac{p}{q}$ and $R := 1 - L$.
- Idea: try to find $f(0), f(1), \ldots, f(N)$ simutaneously.
- We know that:

$$\begin{cases} f(0) & = & 1 \\ f(N) & = & 0 \\ f(X) & = & L \times f(X-1) + R \times f(X+1) \end{cases}$$

- We will discuss the general solution later.

Subtask 2: $N \leq 3$

- For $N = 3$ it is simpler:

$$\begin{cases} f(1) & = & L + R \times f(2) \\ f(2) & = & L \times f(1) \end{cases}$$

Subtask 2: $N \leq 3$

- For $N = 3$ it is simpler:

$$\begin{cases} f(1) & = & L + R \times f(2) \\ f(2) & = & L \times f(1) \end{cases}$$

- Solving gives $f(1) = \frac{L}{1-LR}$ and $f(2) = \frac{L^2}{1-LR}$.

Subtask 3: $N \leq 10$

- Remains to solve the following for $f(x)$:

$$\begin{cases} f(0) & = & 1 \\ f(N) & = & 0 \\ f(X) & = & L \times f(X-1) + R \times f(X+1) \end{cases}$$

Subtask 3: $N \leq 10$

- Remains to solve the following for $f(x)$:

$$\begin{cases} f(0) & = & 1 \\ f(N) & = & 0 \\ f(X) & = & L \times f(X-1) + R \times f(X+1) \end{cases}$$

- Rewrite as $L \times (f(X) - f(X-1)) = R \times (f(X+1) - f(X))$.

Subtask 3: $N \leq 10$

- Remains to solve the following for $f(x)$:

$$\begin{cases} f(0) & = & 1 \\ f(N) & = & 0 \\ f(X) & = & L \times f(X-1) + R \times f(X+1) \end{cases}$$

- Rewrite as $L \times (f(X) - f(X-1)) = R \times (f(X+1) - f(X))$.
- That means $f(X+1) - f(X) = \gamma^X (f(1) - f(0))$, where $\gamma := \frac{L}{R}$.

Subtask 3: $N \leq 10$

- Remains to solve the following for $f(x)$:

$$\begin{cases} f(0) & = & 1 \\ f(N) & = & 0 \\ f(X) & = & L \times f(X-1) + R \times f(X+1) \end{cases}$$

- Rewrite as $L \times (f(X) - f(X-1)) = R \times (f(X+1) - f(X))$.
- That means $f(X+1) - f(X) = \gamma^X (f(1) - f(0))$, where $\gamma := \frac{L}{R}$.
- Sum for $X = 0, 1, 2, \ldots, N-1$,
  $$\begin{aligned} \text{LHS} &= (f(1) - f(0)) + (f(2) - f(1)) + \cdots + (f(N) - f(N-1)) \\ &= f(N) - f(0) = -1 \end{aligned}$$
  $$\text{RHS} = (1 + \gamma + \gamma^2 + \cdots + \gamma^{N-1})(f(1) - f(0))$$

Subtask 3: $N \leq 10$

- Remains to solve the following for $f(x)$:

$$\begin{cases} f(0) & = & 1 \\ f(N) & = & 0 \\ f(X) & = & L \times f(X-1) + R \times f(X+1) \end{cases}$$

- Rewrite as $L \times (f(X) - f(X-1)) = R \times (f(X+1) - f(X))$.
- That means $f(X+1) - f(X) = \gamma^X (f(1) - f(0))$, where $\gamma := \frac{L}{R}$.
- Sum for $X = 0, 1, 2, \ldots, N-1$,
  $\text{LHS} = (f(1) - f(0)) + (f(2) - f(1)) + \cdots + (f(N) - f(N-1))$
  $= f(N) - f(0) = -1$
  $\text{RHS} = (1 + \gamma + \gamma^2 + \cdots + \gamma^{N-1})(f(1) - f(0))$
- Therefore, $f(1) - f(0) = \frac{-1}{1+\gamma+\gamma^2+\cdots+\gamma^{N-1}}$.

Subtask 3: $N \leq 10$

- Now sum for $X = 0, 1, 2, \ldots, K-1$:
  LHS $= f(K) - 1$;
  RHS $= (1 + \cdots + \gamma^{K-1})(f(1) - f(0))$.

Subtask 3: $N \leq 10$

- Now sum for $X = 0, 1, 2, \ldots, K-1$:
  LHS $= f(K) - 1$;
  RHS $= (1 + \cdots + \gamma^{K-1})(f(1) - f(0))$.
- Therefore we get
  $f(K) = 1 - \frac{1+\cdots+\gamma^{K-1}}{1+\cdots+\gamma^{N-1}}$
  $= \frac{\gamma^K+\cdots+\gamma^{N-1}}{1+\cdots+\gamma^{N-1}}$.

Subtask 3: $N \leq 10$

- Now sum for $X = 0, 1, 2, \ldots, K-1$:
  LHS $= f(K) - 1$;
  RHS $= (1 + \cdots + \gamma^{K-1})(f(1) - f(0))$.
- Therefore we get
  $f(K) = 1 - \frac{1+\cdots+\gamma^{K-1}}{1+\cdots+\gamma^{N-1}}$
  $= \frac{\gamma^K+\cdots+\gamma^{N-1}}{1+\cdots+\gamma^{N-1}}$.
- If $\gamma = 1$ (i.e. $L = R$), then $f(K) = \frac{N-K}{N}$.

Subtask 3: $N \leq 10$

- Now sum for $X = 0, 1, 2, \ldots, K-1$:
  LHS $= f(K) - 1$;
  RHS $= (1 + \cdots + \gamma^{K-1})(f(1) - f(0))$.
- Therefore we get
  $f(K) = 1 - \frac{1+\cdots+\gamma^{K-1}}{1+\cdots+\gamma^{N-1}}$
  $= \frac{\gamma^K+\cdots+\gamma^{N-1}}{1+\cdots+\gamma^{N-1}}$.
- If $\gamma = 1$ (i.e. $L = R$), then $f(K) = \frac{N-K}{N}$.
- Otherwise,
  - $\gamma^K + \cdots + \gamma^{N-1} = \gamma^K(\frac{\gamma^{N-K}-1}{\gamma-1})$
  - $1 + \gamma + \cdots + \gamma^{N-1} = \frac{\gamma^N-1}{\gamma-1}$

  and we get $f(K) = \frac{\gamma^N-\gamma^K}{\gamma^N-1} = \frac{L^N-L^KR^{N-K}}{L^N-R^N} = \frac{p^N-p^K(q-p)^{N-K}}{p^N-(q-p)^N}$.

M1805 - Heart Shaper

Percy Wong {percywtc}

Partial Solution - Exhaustion

Simply try all possible places of the “heart-shape”

PSEUDOCODE

```
Ans = infinity
For i = 3*x+1 .. R
  For j = 4*x+1 .. R
    If (Calc(i, j) < Ans)
      Ans = Calc(i, j)
```

Here, `Calc(i, j)` returns the cost (number of bits to toggle) if we place the “heart-shape” with bottom-right bit at the i-th row, j-th column

Partial Solution - Exhaustion

You should carefully implement the function `Calc(i, j)` by analyzing the relations between coordinates of the "heart-shape" and the value **N** (or **x**)

Time Complexity : $O(RCN^2)$ *// considering the whole rectangle as number of bits in a rectangle ~ $O(N^2)$*

Time Complexity : $O(RCN)$ *// considering only the heart-shape as number of bits in a heart-shape ~ $O(N)$*

Expected Score : 8 ~ 18 out of 20 *// based on your implementation*

Full Solution

Notice that the “heart-shape” is formed by six diagonal lines

Or six horizontal / vertical lines,

if you rotate the bitmap ~~(or your head)~~ by 45 degrees

```
   2     3
  6 2   5 3
 6   2 5   3
6     5     4
 1         4
  1       4
   1     4
    1   4
     1 4
      1
```

Full Solution

We know how to find the sum of elements in subarray with *prefix sum*

In case you do not know *prefix sum*...

For an 1-d array `arr[]`, we can build another 1-d array `pre[]`, such that

`pre[i]` is the value of `arr[1] + arr[2] + ... + arr[i]`

This can be calculated in O(N) with the formula

`pre[i] = pre[i-1] + arr[i]`

Thus we can query any subarray sum with O(1) now as

the sum from the x^{th} to y^{th} elements is `pre[y] - pre[x-1]`

Full Solution

We can actually apply the prefix sum technique to the bitmap,
but in diagonal directions instead of horizontal / vertical

In other words, by using the formula

```
pre[i][j] = pre[i-1][j-1] + bitmap[i][j]
```

We can find number of 1-bits of any diagonals from (i, j) to (x, y)

```
count = pre[x][y] - pre[i-1][j-1]
```

// $i \le x$; $j \le y$; $x - i = y - j$

Apply this technique to the other direction as well

Full Solution

Now, we can compute the number of 1-bits in "heart-shape"
with any given positions in O(1) per query *// and O(RC) pre-compute*

By knowing the number of 1-bits in the whole initial bitmap as well,
we can calculate the number of bits to toggle for any given positions too

Time Complexity : O(RC) *// O(RC) pre-compute + O(RC) exhaustion*

Expected Score : 20 out of 20