Mini Comp 3 Editorial

Credits: Steven, Tony, LMH, Sampson

Unable to connect to the Internet

Setter: Steven

Prepared By: Steven

Unable to connect to the Internet

Inspiration:

The dinosaur game in Google Chrome which appears when it is unable to connect to the Internet.

Unable to connect to the Internet

- Problem: optimally control the dinosaur to jump to avoid cactus, how far can it go?

```
4    D     D         D
3   D D   D D       D D
2  D   D D C D     D   D
1 D  C  D  C  DDDDDC    DDDD
  12345678901234567890123456
```

Unable to connect to the Internet

Subtask 1: H=2; Only 3 cases to handle:

- D
  DDDDCDDDDD
- C
  DDDDC
- D
  DDDDCC

Unable to connect to the Internet

Subtask 2: h_i=H-1; more calculations on index

-
```
         D
        DCD  C
       D C D C
      D  C  DC
     D   C   C
    D    C   C
DDDD     C   C
```

Unable to connect to the Internet

Subtask 3:

- Too many cases. Use new approach.
- (1,1) is reachable
- Suppose (i,1) is reachable,
  - if (i+1,1) has no cactus, then (i+1,1) is reachable
  - if the jumping path from (i,1) to (i+2H-2,1) has no cactus, (i+2H-2,1) is reachable

Unable to connect to the Internet

Subtask 3:

- Now we obtain the farthest reachable cell on the ground.
- To obtain the real answer, keep recording the farthest reachable column while checking jumping paths.
- Number of values of i: O(W)
- Checking a jumping path: O(H)
- Overall time complexity: O(WH)

Unable to connect to the Internet

Full solution:

- Checking a jumping path: O(H)
  Speed it up!

- Suppose $cactus_i$ locates at x_i and is h_i tall

Unable to connect to the Internet

Full solution:

- Jumping from $(x_i-h_i+1,1)$, $(x_i-h_i+2,1)$, ..., $(x_i-1,1)$ will hit $cactus_i$

```
    D
   D D
  DC  D
 D C   D
DXXC    D
```

Unable to connect to the Internet

Full solution:

- Jumping from $(x_i-2H+2,1)$, $(x_i-2H+3,1)$, ..., $(x_i-2H+h_i+1,1)$ will also hit cactus_i

```
       D
      D D
     D  CD
    D   C D
XXXD    C  D
```

Unable to connect to the Internet

Full solution:

- Use prefix sum to obtain the boolean arrays:
- j[i] = true if jumping from i will hit a cactus

- Checking a jumping path: O(1)

Unable to connect to the Internet

Full solution:

- We now obtain the farthest reachable cell on the ground. Let it be (G,1).
- Observe that the dinosaur may make a 'suicidal jump' at the end.
- Suppose the 'suicidal jump' starts at (J,1). Then G-2H+2 <= J <= G.
- For all these possible values of J, simulate a jump.

Unable to connect to the Internet

Full solution:

- Prefix sum: O(W)
- Dynamic programming: O(W)
- Number of possible 'suicidal jump': O(H)
- Simulate a jump: O(H)
- Overall time complexity:
  $O(W) + O(W) + O(H) * O(H) = O(W + H^2)$

Inverse Problem 10

Setter: Tony

Prepared By: Tony

Inverse Problem 10

Take N = 20 as an example

20 = 10100(2)

The choice of M will be between

- 2 (number of 1s in the binary representation of N)
- 20 (N)

Inverse Problem 10

If N = 20, M = 5

Start with 10100 (16 + 4), we can see that there are 2 ones

To create a one, we can deduct any bit (except the 2^0 bit) by 1 and add two to the bit on the right

For example, change 16 to 8 + 8 (02100)

This keeps the total sum the same

Repeat this process until we have M ones

Inverse Problem 10 - Outputting the numbers

02100 -> 01300 -> 00500 (4+4+4+4+4)

The number of numbers required is equal to the maximum frequency of all bits (frequency of 2^2 is 5 here)

One solution is to output 2^i freq[i] times

```
for (int i = 0; i < 20; i++) {
    for (int j = 0; j < freq[i]; j++) {
        printf("%d\n", 1 << i);
    }
}
```

Inverse Problem 10 - Full solution

When subtracting and adding bits, greedily select, among the possible bits, the bit which its bit on the right has minimum frequency

- For example, instead of 02100 -> 01300, perform 02100 -> 02020 -> 01220 or 02012

As you can see, required number of numbers become 2 (14 + 6 or 9 + 11)

Inverse Problem 10

Setter: Tony

Prepared By: Tony

Bridge Routing

Setter: LMH

Prepared By: LMH

Bridge Routing - 20% Solution

Exhaustion all possible distributions of people simply by recursion

O((n+m-1) choose (m - 1))

Bridge Routing - 55% Solution

Observe the fact that

- We can treat original problem as route each citizen to different bridge one by one
- For each selection, it is always optimal to choose the bridge that contribute least extra cost while adding one more people on that bridge

Extra cost when adding one more person to bridge i that originally have x people

- Extra cost for x people = $x * c_i$
- Extra cost for the new citizen = $t_i + ((x + 1) - 1) * t_i$
- Total extra cost = $t_i + 2 * c_i * x$

For each selection, search M possible bridges to find which has the least

Bridge Routing - 100% Solution

Using Priority Queue (PQ)

Push all extra costs of adding first person (t_i) to bridge i into PQ

- Each PQ node stores the extra cost and the bridge label

Repeat the following process for N times

- Find the bridge with least extra cost (Remove min of PQ)
- Add the extra cost of chosen bridge
- Update the extra cost of adding one more people and push it into the queue again

O(N log M)

Edges in MST

Setter: Authors from Codeforces Round #111
Prepared By: Sampson

Solution: http://codeforces.com/blog/entry/4108