



# Programming Contest 2017

解説



# 問題一覧

A. MEN☆G

B. 製本

C. 片付け

ほぼ全員に解けて欲しかった

D. WNCS

E.  $P=NP$

F. 2017

G. NOKEMON GO

だんだん難易度が上がっていく

H. PPAP

(´・ω・`) (経験者用)



A. MEN ☆ G



# 問題概要(A)

- ・並盛は $a$ グラム、大盛は $a+b$ グラム、神盛は $a+b+c$ グラム
- ・並盛、大盛、神盛の合計を求めよう

# 解説(A)

- ・ $a+(a+b)+(a+b+c)$ を計算するだけ
- ・書くのが面倒なので $3a+2b+c$ とするとい

(ところで、某油そば屋はエゴサが激しいことが知られているので、  
みんなも注意しよう)



## B. 製本

# 問題概要(B)

- ・今、 $N$ ページが完成している。  
ページを追加して、ページ数を4の倍数にしたい



- ・ $N+x$ が4の倍数になる最小の非負整数 $x$ は？

# 解説(B)

ポイント: 4の倍数を考える  $\Rightarrow$  演算子 “%” を使おう

**方針1.**  $i=0 \sim 3$ をfor文で試して、 $(x+i) \% 4 == 0$ となる $i$ を探す

**方針2.**  $x$ を4で割った余りで場合分け

などなど色々な方法があります.





# 問題概要(C)

- ・文字"○"と"."からなる文字列 $S$ がある
- ・文字"○"を右に全部寄せた同じ長さの文字列を求めよう

例えば



# 解説(C)

- ・答えの文字列はS と比較して、"\_○\_"と"\_.\_"の数と一緒に！

- ・だから

「"\_.\_"をS と同じ数だけ並べて、残りを"○\_"で埋めたもの」

が答え→ループを使おう！

# 別解(C)

実は、"○"と"."では"."のほうがASCIIコードが小さい

"."(小さい方)を手前に"○"(大きい方)を奥に並べたい

ソート

するだけでなんと答えが出る



D. WNCS



# 問題概要(D)

N人が総当りリーグ戦を行う(必ず勝敗がつく)

勝ち数が多い順に上位になれる

勝ち数が同じなら予選順位が高い方が上位になる

リーグ戦の順位はどのようになるか？

# 解説(D)

1. 勝ち数が多い方から見ていく

その勝ち数と一致すればその順位を出力

2. 予選順位が高い方から見ていく

「同じ勝ち数なら予選順位が高い方が上位」という条件もカバーできる！

```
int rank[100], r=0;

for(int i=N-1; i>=0; i--){
    for(int j=1; j<=N; j++){
        if(win[j] == i){
            rank[r++] = j;
        }
    }
}
```

# 別解(D)

勝ち数が多い順に、


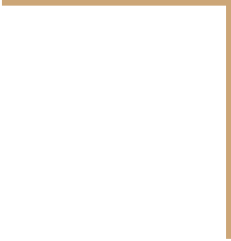
勝ち数が同じ中では 予選順位が小さい順になるように並べたい

ソート

バブルソート、選択ソート、挿入ソート、クイックソート...

$N \leq 100$  なので どれでも十分に速く動作します





E.  $P=NP$

# 問題概要(E)

文字列Sが与えられる

$P \rightarrow NP$ という変換を繰り返すことで、PPAPからSにできるか？

(例)  $S = NPNPAP$

$PAP \rightarrow PNAP \rightarrow NPNAP$

# 解説(E)

変換( $P \rightarrow NP$ )の意味について考える

要するにPの左隣にNを入れる操作

そうすると初期状態(PPAP)から作ることのできる文字列は...

$N \dots NPN \dots NPN \dots NP$

各Pの左隣にNが0個以上連続する文字列のみ作れる

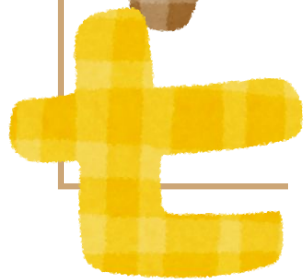
# 解説(E)

どうやって判定する？

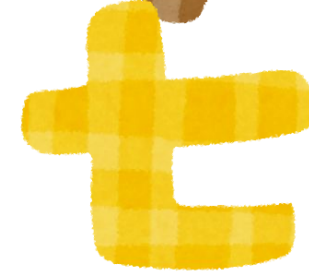
例えば...

- (1) Pがちょうど3つあることを確認
- (2) 最後の文字がPであることを確認
- (3) 2つ目のPの直後がAであることを確認
- (4) それ以外の文字は全てNであることを確認

他にも、正規表現を書くとか



F. 2017



# 問題概要(F)

$1 \times 2 \times 3 \times 4 \times \dots \times N \bmod 2017$

例えば  $29! = 8841761993739701954543616000000$

↑ **64bit整数(long long型)でも入らない**

$$(A \times B) \% M = ((A \% M) \times B) \% M$$

掛け算のあとに、毎回 mod を取ることで、  
常に 2017 以下で扱うことができる

# 解説(F)

```
int ans = 1;

for(int i=2; i<=N; i++){
    ans = (ans * i ) % 2017
}

printf("%d\n", ans);
```

Nがとても大きいので、終わらない！

ちなみにこの実装だとNがとても大きいときオーバーフローも起こしているので、long long型などの64bitの型を使おう

# 解説(F)

よく考えてみると...

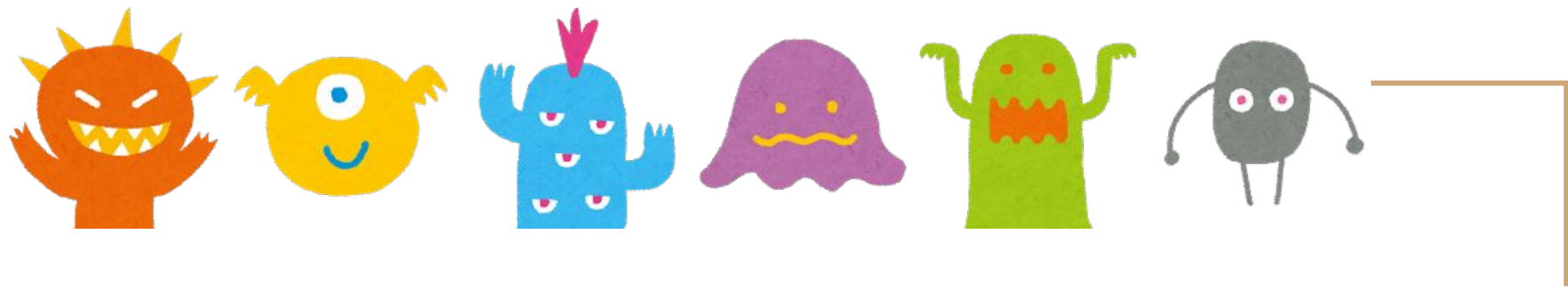
$$1 \times 2 \times 3 \times \dots \times 2016 \times 2017 \times 2018 \times \dots \times N = ?$$

$$\uparrow \text{mod } 2017 = 0$$

よって,  **$N \geq 2017$  のとき, 絶対に答えは0 !**

$N < 2017$  のときは愚直に計算しよう



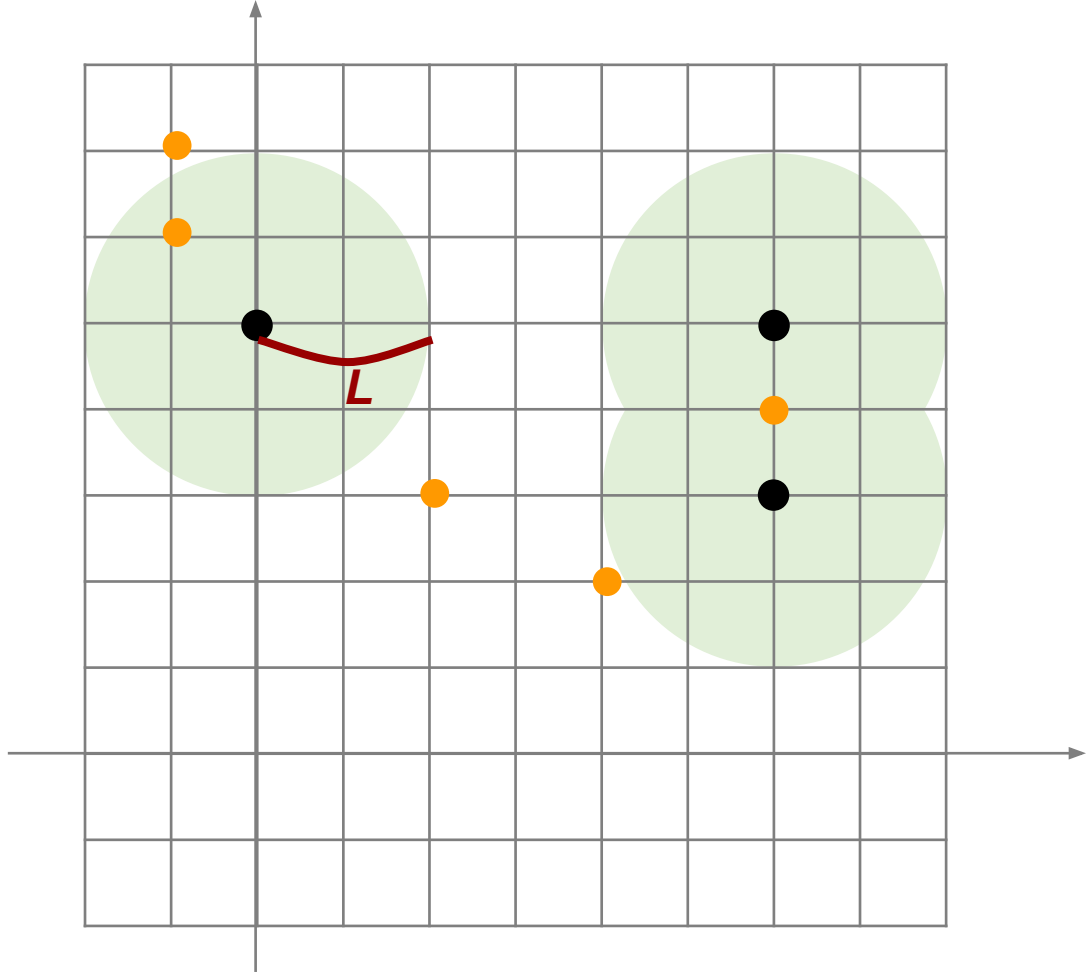


# G. NOKEMON GO



# 問題概要(G)

- ・ $(x,y)$ 座標系で考える
- ・ポケモンが $M$ 匹いる
- ・ポケストップが $N$ 回ある
- ・ポケストップから、**距離 $L$** 以内の  
ポケモンをゲットできる
- ・何匹のポケモンをゲットできるか？



# 解説(G)

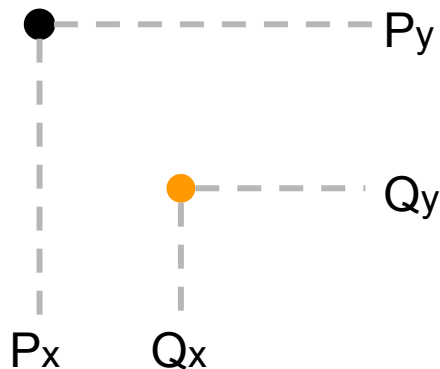
「距離L以内」

→ みんな大好き「**三平方の定理**」より

$$(P_x - Q_x)^2 + (P_y - Q_y)^2 \leq L^2$$

が成り立てば、距離L以内

(ルートとか小数とか考えなくてOK)

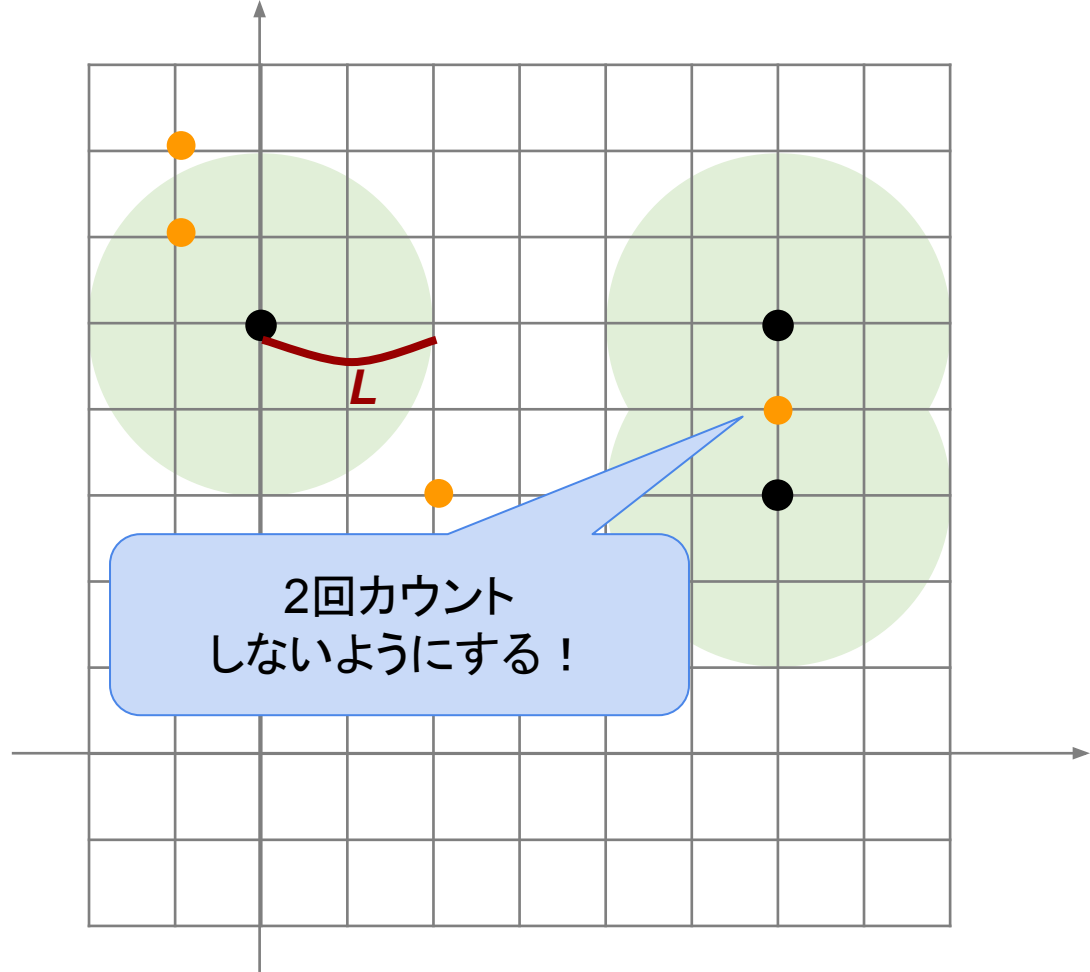


# 解説(G)

- ・2重ループを回そう

```
int count = 0;
```

```
for ( i : 全部のポケストップについて )  
  for ( j : 全部のポケモンについて )  
    if( 距離がL以内 )  
      count++;
```



# 解説(G)

- ・2重ループを回そう

```
int count = 0;
```

```
int checked[M] = {0};
```

```
for ( i : 全部のノケストップについて )
```

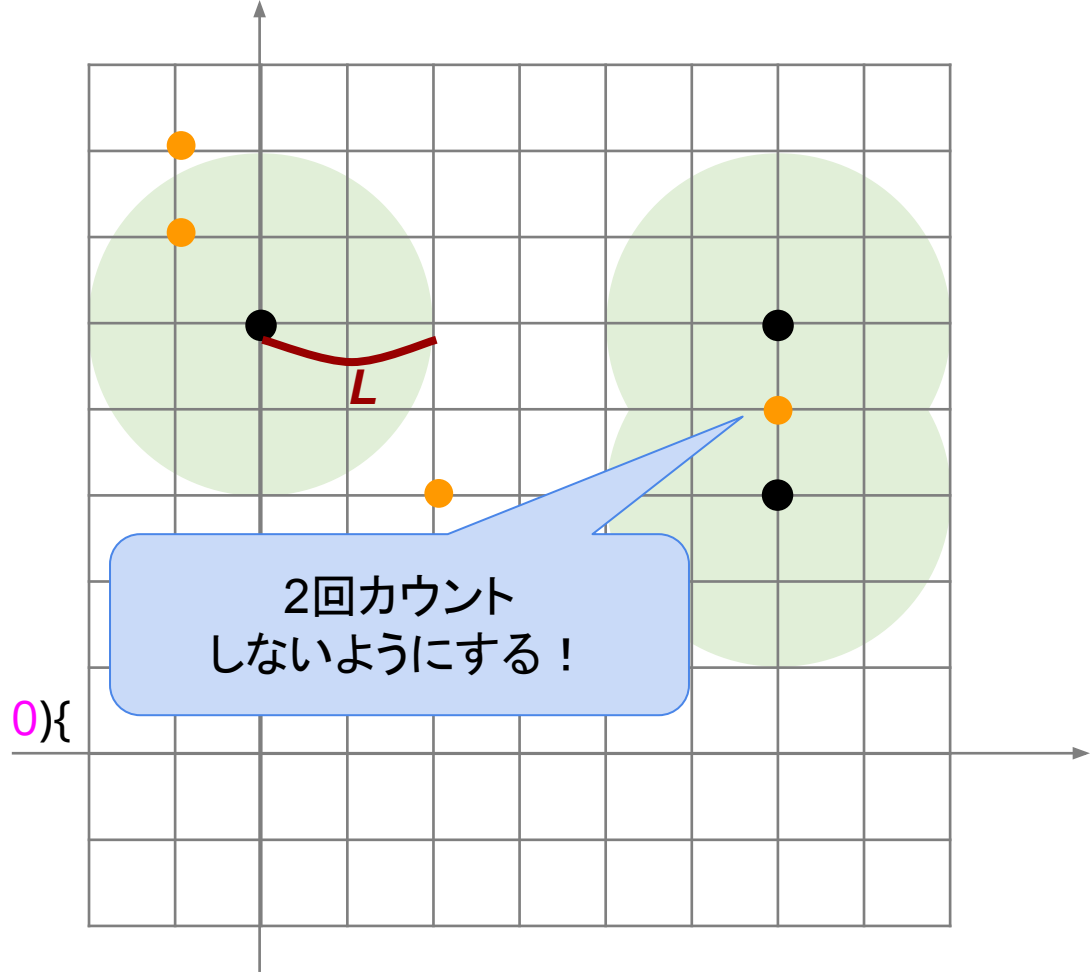
```
    for ( j : 全部のノケモンについて )
```

```
        if( 距離がL以内 && checked[j] == 0 ){
```

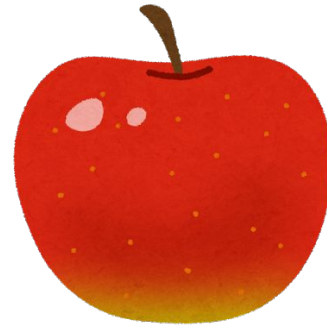
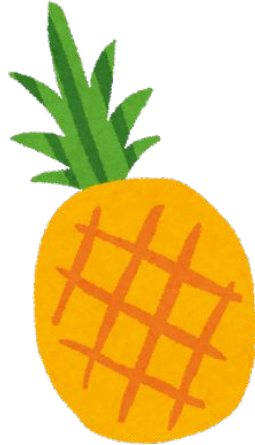
```
            count++;
```

```
            checked[j] = 1;
```

```
        }
```



H. PPAP



# 問題概要

この問題に目を通してくれた方がどれだけいるのか... (´・ω・`)

$N$ 軒の店がある

それぞれの店ではペン  $a_i$  本・リンゴ  $b_i$  個・パイナップル  $c_i$  個のセットを  $r_i$  ルビーで売っている(1軒の店から何セットでも買える)

ペンを  $2x$  本以上、リンゴを  $x$  個以上、パイナップルを  $x$  個以上買いたい

必要な金額の最小値は？

# 解説

「何が分かれば答えを決められるか」を考えてみる

$\text{cost}[i][j][k]$  := ペンを $i$ 本以上、リンゴを $j$ 個以上、パイナップルを $k$ 個以上( $(i,j,k)$ と表す)買うために必要な金額の最小値

とおけば、答えは $\text{cost}[2x][x][x]$

1つずつ状況を整理していく



# 解説

“1軒のお店で何セットでも買うことが出来る” ← このままだと分かりにくい

自由にお店をまわって、1セットずつ買っていく

という解釈にしよう

(例)1軒目で3セット、2軒目で2セット買う

→ 1,1,1,2,2 という順番で店を訪れる という解釈

# 解説

今、ペンを*i*本、リンゴを*j*個、パイナップルを*k*個持っている時：

次に行くお店を*N*軒の中から1つ選ぶ → 店*s*に行くとしたら...

$$\text{cost}[i+as][j+bs][k+cs] = \min(\text{cost}[i+as][j+bs][k+cs], \text{cost}[i][j][k]+rs)$$



# 解説

今、ペンを*i*本、リンゴを*j*個、パイナップルを*k*個持っている時：

次に行くお店を*N*軒の中から1つ選ぶ → 店*s*に行くとしたら...

店*s*で1セット購入 → ペン*i+as*本、リンゴ*j+bs*個、パイナップル*k+cs*個になる

$$\text{cost}[i+as][j+bs][k+cs] = \min(\text{cost}[i+as][j+bs][k+cs], \text{cost}[i][j][k] + rs)$$

# 解説

漸化式の形を見ると、 $\text{cost}[i][j][k]$ は自分より添字の小さい位置にしか依存していない  
→ 小さい位置が全部決まっていれば $\text{cost}[i][j][k]$ も決まる

あとは、この更新ルールに従って $(i,j,k)$ が小さい方から更新していく

$\text{cost}[0][0][0] = 0$  を初期値として、 $\text{cost}[2x][x][x]$ が答えになる










次元を落として考えよう

# 2次元で考える

リンゴとパイナップルだけで考えよう



(具体例)  $N=2, x=4$

店1:   +  = 7 [Rubee]

店2:  +    = 11 [Rubee]

$\text{cost}[x][x]$  はどうなる ???





 	0	1	2	3	4
0	0	INF	INF	INF	INF
1	INF	INF	INF	INF	INF
2	INF	INF	INF	INF	INF
3	INF	INF	INF	INF	INF
4	INF	INF	INF	INF	INF

$$2 \text{ apples} + 1 \text{ pineapple} = 7$$

$$1 \text{ apple} + 3 \text{ pineapples} = 11$$

$$\text{⌘} \text{INF} = \infty$$



 	0	1	2	3	4
0	0	INF	INF	INF	INF
1	INF	INF	INF	INF	INF
2	INF	INF	INF	INF	INF
3	INF	INF	INF	INF	INF
4	INF	INF	INF	INF	INF

$$2 \text{ apples} + 1 \text{ pineapple} = 7$$

$$1 \text{ apple} + 3 \text{ pineapples} = 11$$

・(0,0)のセルは確定



※INF =  $\infty$

 	0	1	2	3	4
0	0	INF	INF	INF	INF
1	INF	INF	INF	11	INF
2	INF	7	INF	INF	INF
3	INF	INF	INF	INF	INF
4	INF	INF	INF	INF	INF

$$2 \text{ apples} + 1 \text{ pineapple} = 7$$

$$1 \text{ apple} + 3 \text{ pineapples} = 11$$



・(0,0)のセルは確定

 	0	1	2	3	4
0	0	INF	INF	INF	INF
1	INF	INF	INF	11	INF
2	INF	7	INF	INF	INF
3	INF	INF	INF	INF	INF
4	INF	INF	INF	INF	INF

$$2 \text{ apples} + 1 \text{ pineapple} = 7$$

$$1 \text{ apple} + 3 \text{ pineapples} = 11$$

・(0,1)のセルは確定



 	0	1	2	3	4
0	0	INF	INF	INF	INF
1	INF	INF	INF	11	INF
2	INF	7	INF	INF	INF
3	INF	INF	INF	INF	INF
4	INF	INF	INF	INF	INF

$$2 \text{ apples} + 1 \text{ pineapple} = 7$$

$$1 \text{ apple} + 3 \text{ pineapples} = 11$$

・(0,1)のセルは確定



INFに何を足しても無意味

 	0	1	2	3	4
0	0	INF	INF	INF	INF
1	INF	INF	INF	11	INF
2	INF	7	INF	INF	INF
3	INF	INF	INF	INF	INF
4	INF	INF	INF	INF	INF

$$2 \text{ apples} + 1 \text{ pineapple} = 7$$

$$1 \text{ apple} + 3 \text{ pineapples} = 11$$

・(0,2)のセルは確定

 	0	1	2	3	4
0	0	INF	INF	INF	INF
1	INF	INF	INF	11	INF
2	INF	7	INF	INF	INF
3	INF	INF	INF	INF	INF
4	INF	INF	INF	INF	INF

$$2 \text{ apples} + 1 \text{ pineapple} = 7$$

$$1 \text{ apple} + 3 \text{ pineapples} = 11$$

・(0,2)のセルは確定



(0,2)、青を買う → (1,5)



x個以上持っている時は、  
x個とみなして構わない







 	0	1	2	3	4
0	0	INF	INF	INF	INF
1	INF	INF	INF	11	INF
2	INF	7	INF	INF	INF
3	INF	INF	INF	INF	INF
4	INF	INF	INF	INF	INF

$$2 \text{ apples} + 1 \text{ pineapple} = 7$$

$$1 \text{ apple} + 3 \text{ pineapples} = 11$$


・(1,3)のセルは確定

 	0	1	2	3	4
0	0	INF	INF	INF	INF
1	INF	INF	INF	11	INF
2	INF	7	INF	INF	22
3	INF	INF	INF	INF	18
4	INF	INF	INF	INF	INF

$$2 \text{ apples} + 1 \text{ pineapple} = 7$$

$$1 \text{ apple} + 3 \text{ pineapples} = 11$$

・(1,3)のセルは確定

 	0	1	2	3	4
0	0	INF	INF	INF	INF
1	INF	INF	INF	11	INF
2	INF	7	INF	INF	22
3	INF	INF	INF	INF	18
4	INF	INF	INF	INF	INF

$$2 \text{ apples} + 1 \text{ pineapple} = 7$$

$$1 \text{ apple} + 3 \text{ pineapples} = 11$$

・(2,1)のセルは確定

 	0	1	2	3	4
0	0	INF	INF	INF	INF
1	INF	INF	INF	11	INF
2	INF	7	INF	INF	22
3	INF	INF	INF	INF	18
4	INF	INF	14	INF	INF

$$2 \text{ apples} + 1 \text{ pineapple} = 7$$

$$1 \text{ apple} + 3 \text{ pineapples} = 11$$

・(2,1)のセルは確定

 	0	1	2	3	4
0	0	INF	INF	INF	INF
1	INF	INF	INF	11	INF
2	INF	7	INF	INF	22
3	INF	INF	INF	INF	18
4	INF	INF	14	INF	INF

$$2 \text{ apples} + 1 \text{ pineapple} = 7$$

$$1 \text{ apple} + 3 \text{ pineapples} = 11$$

・(2,4)のセルは確定

 	0	1	2	3	4
0	0	INF	INF	INF	INF
1	INF	INF	INF	11	INF
2	INF	7	INF	INF	22
3	INF	INF	INF	INF	18
4	INF	INF	14	INF	29

$$\text{apple} + \text{apple} = 7$$

$$\text{apple} + 3 \times \text{pineapple} = 11$$

・(2,4)のセルは確定

(2,4)、青を買う → 22+11



各セルは、**最小値**を保存  
するので、更新は発生しない

 	0	1	2	3	4
0	0	INF	INF	INF	INF
1	INF	INF	INF	11	INF
2	INF	7	INF	INF	22
3	INF	INF	INF	INF	18
4	INF	INF	14	INF	29

$$2 \text{ apples} + 1 \text{ pineapple} = 7$$

$$1 \text{ apple} + 3 \text{ pineapples} = 11$$

・(3,4)のセルは確定

 	0	1	2	3	4
0	0	INF	INF	INF	INF
1	INF	INF	INF	11	INF
2	INF	7	INF	INF	22
3	INF	INF	INF	INF	18
4	INF	INF	14	INF	25

$$2 \text{ apples} + 1 \text{ pineapple} = 7$$

$$1 \text{ apple} + 3 \text{ pineapples} = 11$$

・(3,4)のセルは確定

赤でも青でも、(4,4)



各セルは、**最小値**を保存するので、**赤**を採用





 	0	1	2	3	4
0	0	INF	INF	INF	INF
1	INF	INF	INF	11	INF
2	INF	7	INF	INF	22
3	INF	INF	INF	INF	18
4	INF	INF	14	21	25

$$\text{apple} + \text{apple} + \text{pineapple} = 7$$

$$\text{apple} + \text{pineapple} + \text{pineapple} + \text{pineapple} = 11$$

・(4,4)のセルは確定

・答えは25

# 2次元で考える

今、動かしてみたこと

配列とfor文で実現できる

(INF は 十分に大きい数で代用)

3次元になっても(ペンが要素として増えても)、同じ要領で考えられる

# 補足

動的計画法 / Dynamic Programming(DP) と呼ばれるアルゴリズム

初見での理解は難しかったかも...

2016年の最終問(ラボライフ！)は、これの簡単なバージョン。

似た問題たち → <http://judge.u-aizu.ac.jp/onlinejudge/finder.jsp?course=DPL>



# 表彰式



# 上位入賞

1位～9位

# 上位者紹介

1位	とまと	1628
2位	@hidollara	1522
3位	マイケル	985
4位	ikarostech	795
5位	seri911	746

# 上位者紹介

6位	zhacro	691
7位	kharu45	688
8位	mi	678
9位	hitofish	546
10位	Keito120607	538
11位	alanc	495





# 上位賞

10位～15位



# 上位者紹介

12位    あたふた/at0x0ft    486

13位    shi    481

14位    ruikyon    477

15位    くろゆきひめ    468

# 2のべき 賞

$2^4, 2^5, 2^6$

## 2のべき賞

16位	kojak	461
32位	kazu10	267
64位	futafuta22	25

# Last Large AC賞

各問題で最後にLargeをACした人  
(上位入賞者除く)

# Last Large AC賞

MEN☆G	hinafunahashi	15:42:43
製本	mira	15:41:10
片付け	dyuhuhuhuhuhu	15:34:27
P=NP	tyougenni	15:29:38

# 手作業賞

satok

s162604912

		SLOT 1512							
63	s162604912	8	8 (1)	-	-	-	-	28	44
		20:50	26:40	-	-	-	-	(1)	
		(3)	-	-	(1)	-	-	52:44	
		-	-	-	-	-	-	-	





以下、宣伝です

# ICPCに参加しよう！

ICPCとは・・・

- 大学対抗のチーム戦プログラミングコンテスト

(3分でわかるicpc : <http://icpc.iisf.or.jp/acm-icpc/3min/>)

ICPCのここがアツイ！

- チーム戦だから、仲間と一緒に戦える！



# Programming Daisuki Clubって、、？？

本学初(?)の競技プログラミングサークル

- U-30プログラミングコンテスト優勝 など

以下の条件を満たす人はぜひPDCへ！

1. プログラミングが好きだ！

(→@DAyamaCTFにDMしてね)



# セキュリティキャンプに行こう！

セキユキャンとは...


- ・8/14-18の5日間
- ・**最高レベルの講師陣**による講義・実習
- ・朝から晩まで
- ・しかも**無料**で！
- ・今年は何のづくりコースもあるよ！  
自作OS・自作言語・**自作CPU**！！



応募締切: **5/29 正午**



# お疲れ様でした！



作問したTAは懇親会にも参加しているので  
気軽に声掛けてね