２０２０数学オリンピック予選　問題　１２

梅山新一郎 2020 年 2 月

以下の記述にあるプログラムは python で書かれています．ソースは梅屋萬年堂のホームページにあります．
`http://umeyamann.web.fc2.com/`

1　良い数列を観察する

問１２の条件の最大整数３０を変えて調べるために C とおく．

- すべて C 以下で相異なる　$\cdots$ 条件 A
- $i=1,2,\cdots,k-1$ に対し、i が奇数ならば a_{i+1} は a_i の倍数である．i が偶数ならば a_{i+1} は a_i の約数である．$\cdots$ 条件 B

では、$C=1$ から

$C=1$ のとき　　[1]　の長さ 1
$C=2$ のとき　　[1 2]　の長さ 2
$C=3$ のとき　　[1 2] または [1 3]　の長さ 2
$C=4$ のとき　　[2 4 1 3]　の長さ 4
$C=5$ のとき　　[2 4 1 3] または [2 4 1 5]　の長さ 4

$C=40$ までを、下の B_1、B_2 と B_3 を使ってコンピュータで計算してみる．資料 1（6 ページ）が計算結果、資料 2（7 ページ）がプログラムのソースリストである．

$\dfrac{C}{2}$ の整数部分 $\left[\dfrac{C}{2}\right]=M$ とし、1 から C までの整数を S_c、それらの数を M 以下と M より大きい数の 2 つの集合 $S_d=\{1,2,\cdots,M\}$ と $S_m=\{M+1,M+2\cdots,C\}$ に分け、条件 B を調べると

- 良い数列は、(約数、倍数、約数、倍数、　、、、）とならび、奇数番目の項は約数で偶数番目の項は倍数 $\cdots B_1$
- $n\in S_m$ となる整数 n は倍数の項、つまり偶数番目の項にしかない．$\cdots B_2$
- $n\in S_d$ となる整数 n は、ほとんどが約数の項（奇数番目の項）に入る．$\cdots B_3$
- $p\in S_m$ となる素数 p は良い数列の中には $a_1,\cdots,1,p$ の形で最終項に 1 つだけしか入れない．$\cdots B_4$

2　良い数列のパターン

2.1　C が奇数と偶数

C が奇数のときは S_d が $C-1$ と同じだから、良い数列の長さは同じになるはず．C が偶数の時だけを考えれば良いのではないかと考えたが、（資料１）から数 C を偶数と奇数に分け、良い数列の長さ k でまとめた下の表を見ると、

偶数 C	2	4	6	8	10	12	14	16	18	20	22	24	26	28	30	32	34	36	38
長さ k	2	4	6	6	9	10	11	13	15	15	17	19	19	21	23	23	25	27	27
長さ k	2	4	6	8	9	10	13	13	15	17	17	19	21	21	23	25	25	27	29
奇数 C	3	5	7	9	11	13	15	17	19	21	23	25	27	29	31	33	35	37	39

となり、$C = 3 \times (\text{奇数})$ のときには長さは 2 だけ増える．これは次に述べる公約数の個数が $C-1$ より 1 個多くなるからで、C の偶奇を考えても良い数列の長さを求める有効なものはならない．

2.2　真約数の公約数

1 から C までの数の真約数[*1]の中で異なる公約数の個数を調べてみる．1 から調べて $1, 2, 3$ になったところで真約数の公約数 1 が現れる．このようにして順に調べていくとき、数 d が初めて真約数の公約数として現れるのは、$1 \times d, , , 2 \times d, , , 3 \times d$ となった数 $3 \times d$ のときで、1 から 30 までの数の真約数の公約数の個数は $1, 2, 3, \cdots, 10$ の 10 個．

この異なる公約数の個数が ℓ 個のとき、（上段が倍数の偶数項、下段が約数の奇数項）

$$\begin{array}{ccccccccc} a_2 & & a_4 & & a_6 & \cdots & a_{2\ell} & & a_{2\ell+2} \\ & \boldsymbol{d}_1 & & \boldsymbol{d}_2 & & & & \boldsymbol{d}_\ell & \end{array} \qquad \text{(P0)}$$

このならびの両端に項をつけて、最長の良い数列は次の (P1) か (P2) のいずれかのパターンになる．

$$\begin{array}{ccccccccccc} & a_2 & & a_4 & & a_6 & \cdots & a_{2\ell} & & a_{2\ell+2} & \\ \boldsymbol{a}_1 & & \boldsymbol{d}_1 & & \boldsymbol{d}_2 & & & & \boldsymbol{d}_\ell & & \boldsymbol{a}_{2\ell+3} \end{array} \qquad \text{(P1)}$$

$$\begin{array}{cccccccccc} & a_2 & & a_4 & & a_6 & \cdots & a_{2\ell} & & a_{2\ell+2} \\ \boldsymbol{a}_1 & & \boldsymbol{d}_1 & & \boldsymbol{d}_2 & & & & \boldsymbol{d}_\ell & \end{array} \qquad \text{(P2)}$$

$C = 30$ のとき、$\ell = 10$ だから $\boldsymbol{d}_1, \cdots, \boldsymbol{d}_{10}$ をうまくつなぐことができれば (P1) で長さ
$1 + 10 + (10 + 1) + 1 = 23$、(P2) の方は $1 + 10 + (10 + 1) = 22$ となる．

[*1] 自分自身を除く約数

1 節で使った $M=\left[\dfrac{C}{2}\right]$ と S_d、S_m とで記述された条件 B_2、B_3、B_4、それと $\ell=\left[\dfrac{C}{3}\right]$（$\ell$ が真約数の異なる公約数の個数）とで $1\cdots C$ までの数 S_c の中に以下の 3 つのグループにつくる．

S_ℓ　: $1,2,\cdots\ell$．この数は約数の項（奇数項）にだけ入る．
S_{m-p} : S_m から、この中の素数 p を除いたもの．この数は倍数の項（偶数項）だけに入る．
　S_m 中の素数は高々 1 個しか使えない．良い数列のパターン (P1) ではこの素数は使えない．
$S_{d-\ell}$: S_d 内の ℓ までの数を除いた残り $\ell+1,\cdots M$．この数は約数や倍数の項となりうる．

資料 1（6 ページ）や表 1 [*2]（5 ページ）から $C=2,3,4,5,6,7,8,9$ と $C=12,13$ で良い数列の長さが偶数のパターン (P2) で、それ以外では (P1) の奇数である．この偶数になる条件と良い数列の長さを考える．

- **C0** $S_{d-\ell}$ の数 d の倍数は S_c の中では $2d$ の一つだけである．パターン (P0) の d_1 から d_ℓ は倍数を 2 個もつから、この数 d が約数の項として使えるのは初項か奇数項の末項のみである．
- **C1** パターン (P0) から (P1) を作るためには、初項 a_1 と末項 $a_{2\ell+3}$ の 2 個の約数が必要になる．$C=2,3,\cdots,7$ と 9 は $n(S_{d-\ell})<2$ であり、(P1) は作れないが．(P2) はできて、長さは偶数になる．
- **C2** 次に $n(S_{d-\ell})=2$ のときは、この 2 個で初項と末項を作る．d_1 から d_ℓ の全てが良い数列の中でならべられるためには倍数の項が $l+1$ 個必要になる．$n(S_{d-\ell})=2$ でこの 2 個を使ってしまうと倍数の項として使えるのは S_{m-p} の数のみ．$C=8$、12、13 では倍数になる数が不足しているからパターン (P1) は作れない．**C1** と同様に (P2) ができて、その長さが偶数になる．
- **C3** $C=10$、11、15 では $n(S_{d-\ell})=2$ だが、$S_{m-p}\geq\ell+1$ で倍数になる数が充足されるので (P1) を作ることができる．
- **C4** $n(S_{d-\ell})\geq 3$ のときは初項と末項の 2 個の約数も確保でき、$S_{d-\ell}$ からこの 2 個を除いた数も使えば倍数の項も $n(S_{m-p})+n(S_{d-\ell})-2\geq\ell+1$ になるから倍数の項も充足される．$C\geq 16$ では $n(S_{m-\ell})\geq 3$ であり **C1** や **C2** の偶数になるものは限定的である．

以上のことから、初項、公約数の個数、倍数の個数、末項と数えることで、次のように考えた．

良い数列の最大の長さ k と上限 k_{max}

$\left[\dfrac{C}{3}\right]=\ell$　として
$\boldsymbol{C}=2,3,\cdots,9$ と $12,13$　のとき　$k=1+\ell+(\ell+1)$
$\boldsymbol{C}=10,11$ と $C\geq 16$　のとき　$k\leq k_{max}=1+\ell+(\ell+1)+1$

[*2] MO2020P12.py からのデータを数 C、良い数列の長さ k、S_d の個数 M、真約数の公約数の個数 ℓ、$S_{d-\ell}$、さらに S_{m-p} の個数、良い数列の中では使われなかった数（not-used）などでまとめたものである

3　良い数列を作る

次のような「作成手順」に従って作ってみる.

- 真約数の公約数を大きい数から順に 3 まで、次のセットを作り、倍数のつなぎに注意して並べる.
- 倍数 - **公約数** - 倍数　のセットを作る. 倍数は大きい数から、同じ数は 2 回まで使い、3 回目になるときはより小さな数を使う. それができないときには、今作っている公約数の前の（1 つ大きい）セットを作り直す.
- 両端に 1 と 2 をおき、$S_{d-\ell}$ の数で初項と末項を作る. 作ったセットがつながらないときは、1 を途中の入れ $S_{d-\ell}$ の数を約数や倍数に使い調整する.
- うまくいかなくなったら入れ換えなどでやり直す.

$C=10$、20 と 30 を作る. 以下の記述で太字の項は、数列の奇数項（約数の項）としている.

- $C=10$: 異なる公約数の個数 $\ell=3(1,2,3)$ 、目標の良い数列の長さ $k=1+3+(3+1)+1=9$
  作るセットは $9-\mathbf{3}-6$ だけ. 両端に 1、2 をおき、$\mathbf{4}-8-\mathbf{1}$、$\mathbf{2}-10-\mathbf{5}$ を作り完成.
  $\mathbf{4}-8-\mathbf{1}-9-\mathbf{3}-6-\mathbf{2}-10-\mathbf{5}$.

- $C=20$: 異なる公約数の個数 $\ell=6(1,2,3,4,5,6)$ 、目標の良い数列の長さ $k=1+6+(6+1)+1=15$
  作るセットは $18-\mathbf{6}-12$ から始めて、$20-\mathbf{5}-15$、$20-\mathbf{4}-16$、$18-\mathbf{3}-15$. これをつなぐと $16-\mathbf{4}-20-\mathbf{5}-15-\mathbf{3}-18-\mathbf{6}-12$. 2 は端に入れて $\mathbf{2}-14-\mathbf{7}$ を置けるのですが、1 を端においても完成しないので、途中の 5 と 15 の間に $\mathbf{5}-10-\mathbf{1}-15-\mathbf{3}$ と入れることにする.
  $\mathbf{8}-16-\mathbf{4}-20-\mathbf{5}-10-\mathbf{1}-15-\mathbf{3}-18-\mathbf{6}-12-\mathbf{2}-14-\mathbf{7}$

- $C=30$: 異なる公約数の個数 $\ell=10(1,2,\cdots,10)$ 、目標の良い数列の長さ $k=1+10+(10+1)+1=23$
  作るセットは $30-\mathbf{10}-20$、$27-\mathbf{9}-18$、$24-\mathbf{8}-16$、$28-\mathbf{7}-21$、$30-\mathbf{6}-24$、$25-\mathbf{5}-20$、$28-\mathbf{4}-16$、$27-\mathbf{3}-21$. このセットを倍数の繋がりに注意してならべて、
  $18-\mathbf{9}-27-\mathbf{3}-21-\mathbf{7}-28-\mathbf{4}-16-\mathbf{8}-24-\mathbf{6}-30-\mathbf{10}-20-\mathbf{5}-25$
  両端に 1 と 2 を $\mathbf{11}-22-\mathbf{2}-18$ と $25-\mathbf{1}-26-\mathbf{13}$ つなぎ完成
  $\mathbf{11}-22-\mathbf{2}-18-\mathbf{9}-27-\mathbf{3}-21-\mathbf{7}-28-\mathbf{4}-16-\mathbf{8}-24-\mathbf{6}-30-\mathbf{10}-20-\mathbf{5}-25-\mathbf{1}-26-\mathbf{13}$

  表 2 (8 ページ) は $C=30$ 約数一覧で、このセットを作成するときに使った.

表 1　それぞれの C での良い数列の長さ k、$M = \left[\frac{C}{2}\right]$、$S_m$ 内の素数 p など

C	k	M	ℓ	$n(S_{d-\ell})$	$n(S_{m-p}) - n(S_{d-\ell}) - 2$	S_m	p in S_m	not-used
2	2	1	0	1	(1 - 1) + 1 - 2 = -1	2 ... 2	2	()
3	2	1	1	0	(2 - 2) + 0 - 2 = -2	2 ... 3	2 3	()
4	4	2	1	1	(2 - 1) + 1 - 2 = 0	3 ... 4	3	()
5	4	2	1	1	(3 - 2) + 1 - 2 = 0	3 ... 5	3 5	()
6	6	3	2	1	(3 - 1) + 1 - 2 = 1	4 ... 6	5	()
7	6	3	2	1	(4 - 2) + 1 - 2 = 1	4 ... 7	5 7	()
8	6	4	2	2	(4 - 2) + 2 - 2 = 2	5 ... 8	5 7	()
9	8	4	3	1	(5 - 2) + 1 - 2 = 2	5 ... 9	5 7	()
10	9	5	3	2	(5 - 1) + 2 - 2 = 4	6 ... 10	7	(7)
11	9	5	3	2	(6 - 2) + 2 - 2 = 4	6 ... 11	7 11	(7 11)
12	10	6	4	2	(6 - 2) + 2 - 2 = 4	7 ... 12	7 11	()
13	10	6	4	2	(7 - 3) + 2 - 2 = 4	7 ... 13	7 11 13	()
14	11	7	4	3	(7 - 2) + 3 - 2 = 6	8 ... 14	11 13	(11 13)
15	13	7	5	2	(8 - 2) + 2 - 2 = 6	8 ... 15	11 13	(11 13)
16	13	8	5	3	(8 - 2) + 3 - 2 = 7	9 ... 16	11 13	(11 13)
17	13	8	5	3	(9 - 3) + 3 - 2 = 7	9 ... 17	11 13 17	(11 13 17)
18	15	9	6	3	(9 - 3) + 3 - 2 = 7	10 ... 18	11 13 17	(11 13 17)
19	15	9	6	3	(10 - 4) + 3 - 2 = 7	10 ... 19	11 13 17 19	(11 13 17 19)
20	15	10	6	4	(10 - 4) + 4 - 2 = 8	11 ... 20	11 13 17 19	(11 13 17 19)
21	17	10	7	3	(11 - 4) + 3 - 2 = 8	11 ... 21	11 13 17 19	(11 13 17 19)
22	17	11	7	4	(11 - 3) + 4 - 2 = 10	12 ... 22	13 17 19	(13 17 19)
23	17	11	7	4	(12 - 4) + 4 - 2 = 10	12 ... 23	13 17 19 23	(13 17 19 23)
24	19	12	8	4	(12 - 4) + 4 - 2 = 10	13 ... 24	13 17 19 23	(13 17 19 23)
25	19	12	8	4	(13 - 4) + 4 - 2 = 11	13 ... 25	13 17 19 23	(13 17 19 23)
26	19	13	8	5	(13 - 3) + 5 - 2 = 13	14 ... 26	17 19 23	(17 19 23)
27	21	13	9	4	(14 - 3) + 4 - 2 = 13	14 ... 27	17 19 23	(17 19 23)
28	21	14	9	5	(14 - 3) + 5 - 2 = 14	15 ... 28	17 19 23	(17 19 23)
29	21	14	9	5	(15 - 4) + 5 - 2 = 14	15 ... 29	17 19 23 29	(17 19 23 29)
30	23	15	10	5	(15 - 4) + 5 - 2 = 14	16 ... 30	17 19 23 29	(17 19 23 29)
31	23	15	10	5	(16 - 5) + 5 - 2 = 14	16 ... 31	17 19 23 29 31	(17 19 23 29 31)
32	23	16	10	6	(16 - 5) + 6 - 2 = 15	17 ... 32	17 19 23 29 31	(17 19 23 29 31)
33	25	16	11	5	(17 - 5) + 5 - 2 = 15	17 ... 33	17 19 23 29 31	(17 19 23 29 31)
34	25	17	11	6	(17 - 4) + 6 - 2 = 17	18 ... 34	19 23 29 31	(19 23 29 31)
35	25	17	11	6	(18 - 4) + 6 - 2 = 18	18 ... 35	19 23 29 31	(19 23 29 31)
36	27	18	12	6	(18 - 4) + 6 - 2 = 18	19 ... 36	19 23 29 31	(19 23 29 31)
37	27	18	12	6	(19 - 5) + 6 - 2 = 18	19 ... 37	19 23 29 31 37	(19 23 29 31 37)
38	27	19	12	7	(19 - 4) + 7 - 2 = 20	20 ... 38	23 29 31 37	(23 29 31 37)
39	29	19	13	6	(20 - 4) + 6 - 2 = 20	20 ... 39	23 29 31 37	(23 29 31 37)
40	29	20	13	7	(20 - 4) + 7 - 2 = 21	21 ... 40	23 29 31 37	(23 29 31 37)

資料１　２から 40 までの良い数列の計算結果

```
c：良い数列の最大長、([良い数列の例] ... 良い数列の個数)
 2 :  2  ([1 2] ... 1)
 3 :  2  ([1 2] ... 2)
 4 :  4  ([2 4 1 3] ... 1)
 5 :  4  ([2 4 1 3] ... 2)
 6 :  6  ([3 6 2 4 1 5] ... 1)
 7 :  6  ([3 6 2 4 1 5] ... 2)
 8 :  6  ([3 6 1 4 2 8] ... 11)
 9 :  8  ([4 8 2 6 3 9 1 5] ... 2)
 10 :  9  ([4 8 1 9 3 6 2 10 5] ... 4)

 11 :  9  ([4 8 1 9 3 6 2 10 5] ... 4)
 12 :  10  ([5 10 1 6 2 8 4 12 3 9] ... 23)
 13 :  10  ([5 10 1 6 2 8 4 12 3 9] ... 26)
 14 :  11  ([5 10 1 6 3 12 4 8 2 14 7] ... 12)
 15 :  13  ([6 12 4 8 1 9 3 15 5 10 2 14 7] ... 4)
 16 :  13  ([6 12 3 15 5 10 1 8 4 16 2 14 7] ... 38)
 17 :  13  ([6 12 3 15 5 10 1 8 4 16 2 14 7] ... 38)
 18 :  15  ([7 14 1 8 4 16 2 10 5 15 3 12 6 18 9] ... 14)
 19 :  15  ([7 14 1 8 4 16 2 10 5 15 3 12 6 18 9] ... 14)
 20 :  15  ([7 14 1 8 2 16 4 12 6 18 3 15 5 20 10] ... 196)

 21 :  17  ([8 16 1 10 2 14 7 21 3 15 5 20 4 12 6 18 9] ... 68)
 22 :  17  ([8 16 1 10 2 14 7 21 3 15 5 20 4 12 6 18 9] ... 332)
 23 :  17  ([8 16 1 10 2 14 7 21 3 15 5 20 4 12 6 18 9] ... 332)
 24 :  19  ([9 18 1 10 5 20 4 16 8 24 6 12 3 21 7 14 2 22 11] ... 388)
 25 :  19  ([9 18 1 10 5 20 4 16 8 24 6 12 3 21 7 14 2 22 11] ... 562)
 26 :  19  ([9 18 1 10 5 20 4 16 8 24 6 12 3 21 7 14 2 22 11] ... 1132)
 27 :  21  ([10 20 4 16 8 24 6 18 9 27 1 25 5 15 3 21 7 14 2 22 11] ... 148)
 28 :  21  ([10 20 1 25 5 15 3 27 9 18 6 24 8 16 4 28 7 14 2 22 11] ... 2116)
 29 :  21  ([10 20 1 25 5 15 3 27 9 18 6 24 8 16 4 28 7 14 2 22 11] ... 2116)
 30 :  23  ([11 22 1 14 7 28 4 16 8 24 6 18 9 27 3 15 5 20 10 30 2 26 13] ... 1348)

 31 :  23  ([11 22 1 14 7 28 4 16 8 24 6 18 9 27 3 15 5 20 10 30 2 26 13] ... 1348)
 32 :  23  ([11 22 1 12 2 14 7 28 4 20 10 30 5 15 3 27 9 18 6 24 8 32 16] ... 8174)
 33 :  25  ([12 24 4 16 8 32 2 22 11 33 1 15 5 20 10 30 6 18 9 27 3 21 7 28 14] ... 426)
 34 :  25  ([12 24 4 16 8 32 2 22 11 33 1 15 5 20 10 30 6 18 9 27 3 21 7 28 14] ... 566)
 35 :  25  ([12 24 1 16 8 32 4 28 7 35 5 20 10 30 6 18 9 27 3 33 11 22 2 26 13] ... 10446)
 36 :  27  ([13 26 1 14 2 22 11 33 3 15 5 35 7 28 4 20 10 30 6 18 9 36 12 24 8 32 16] ... 5222)
 37 :  27  ([13 26 1 14 2 22 11 33 3 15 5 35 7 28 4 20 10 30 6 18 9 36 12 24 8 32 16] ... 5222)
 38 :  27  ([13 26 1 14 2 22 11 33 3 15 5 35 7 28 4 20 10 30 6 18 9 36 12 24 8 32 16] ... 8224)
 39 :  29  ([14 28 4 16 8 24 12 36 6 18 9 27 1 33 11 22 2 26 13 39 3 21 7 35 5 20 10 30 15] ... 312)
 40:   29  ([14 28 1 33 11 22 2 26 13 39 3 21 7 35 5 20 4 24 12 36 9 18 6 30 10 40 8 32 16] ... 8748)
```

資料 2　　MO2020P.py

```
# coding: utf-8
import gc
import csv
import sympy
import time
from dateutil.relativedelta import relativedelta


class maxLength:

    def __init__(self, C):
        self.C = C
        self.maxK = 0
        self.maxList = []
        self.resultsL = []

    def multiples(self, a):
        return [i * a for i in range(2, self.C // a + 1)]

    def checkK(self, k):
        if self.maxK < k:
            self.maxK = k
            self.maxList = self.L[k]
            self.resultsL = [self.L[k]]
        elif self.maxK == k:
            self.resultsL  += [self.L[k]]

    def evenK(self, k, preAk):
        if k > self.C:
            return
        for ak in self.multiples(preAk):
            if ak in self.L[k - 1]:
                continue
            else:
                self.L[k] = self.L[k -1] + [ak]
                self.checkK(k)
                self.oddK(k + 1, ak)

    def oddK(self, k, preAk):
        if k > self.C:
            return
        for ak in sympy.divisors(preAk)[:-1]:
            if ak in self.L[k - 1]:
                continue
            else:
                self.L[k] = self.L[k - 1] + [ak]
                self.checkK(k)
                self.evenK(k + 1, ak)

    def search(self):
        self.maxK = 1
        self.maxList = [1]
        for a1 in range(1, self.C // 2 + 1):
            self.L = [[] for i in range(self.C + 1)]
            K = 1
            self.L[1] = [a1]
            self.evenK(2, a1)

#  ---------------------------------------------------------------
if __name__ == '__main__':
    startTime = time.time()
    c = 30
    with open('MO2020P12C' + str(c) + '.csv', 'w', encoding = 'utf-8') as fo:
        writer = csv.writer(fo)
        S = maxLength(c)
        S.search()
        print('c = {:2d} : max length = {:2d}  ,  example {}'.format(c, S.maxK, ' '.join(map(str, S.resultsL[0]))))
        writer.writerow(['c = ' , str(c) , ' , max length = ', str(S.maxK)])
        writer.writerow([' '.join(map(str, nums)) for nums in  S.resultsL])


    elapsed_time = time.time() - startTime
    print()
    print('   -->   {0.hours:02}:{0.minutes:02}:{0.seconds:02} '.format(relativedelta(seconds=elapsed_time)))
```

表 2 divisors of C (1 ... 30)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	2		2		2		2		2		2		2		2		2		2		2		2		2		2		2
		3			3			3			3			3			3			3			3			3			3
			4				4				4				4				4				4				4		
				5					5					5					5					5					5
					6						6						6						6						6
						7							7							7							7		
							8								8								8						
								9									9									9			
									10										10										10
										11											11								
											12												12						
												13													13				
													14														14		
														15															15
															16														
																17													
																	18												
																		19											
																			20										
																				21									
																					22								
																						23							
																							24						
																								25					
																									26				
																										27			
																											28		
																												29	
																													30