



-  _____ •  _____
-  _____ •  _____
-  _____ •  _____

[illegible]

--	--	--	--	--

[illegible][illegible][illegible][illegible]

--	--	--	--	--	--

[illegible]

11

0.1 卦象与二进制

- 卦象 由 8 个爻组成，每个爻可以是 阴爻 (broken line) 或 阳爻 (solid line)。1 个爻 / 二进制位 2 个爻 / 二进制位 3 个爻 / 二进制位
- 卦象 可以用 Python 列表 [阴, 阴, 阴] 表示 0 个爻 2 个爻
- 卦象 可以用 3 个二进制位表示 LSB, Bit 0 阴 Bit 1 阴 MSB (Bit 2) 阴
$$\text{Value} = \text{bit}_1 \times 1 + \text{bit}_2 \times 2 + \text{bit}_3 \times 4$$

卦序	卦名	卦象	二进制 (阴=0, 阳=1)	十进制 (Bit 0~2)	五行	方位
1	乾 (Qian)	☰	[1, 1, 1]	7 (111b)	金	西北 (Northwest)
2	兑 (Dui)	☱	[1, 1, 0]	3 (011b)	金	西 (West)
3	离 (Li)	☲	[1, 0, 1]	5 (101b)	火	南 (South)
4	震 (Zhen)	☳	[1, 0, 0]	1 (001b)	木	东 (East)
5	巽 (Xun)	☴	[0, 1, 1]	6 (110b)	木	东南 (Southeast)
6	坎 (Kan)	☵	[0, 1, 0]	2 (010b)	水	北 (North)
7	艮 (Gen)	☶	[0, 0, 1]	4 (100b)	土	东北 (Northeast)
8	坤 (Kun)	☷	[0, 0, 0]	0 (000b)	土	西南 (Southwest)

0.2 卦象与时间

卦序	卦名	时间范围	五行	方位
1	乾	23:00 — 01:00	金	西北
2	兑	01:00 — 03:00	金	西
3	离	03:00 — 05:00	火	南
4	震	05:00 — 07:00	木	东
5	巽	07:00 — 09:00	木	东南
6	坎	09:00 — 11:00	水	北
7	艮	11:00 — 13:00	土	东北
8	坤	13:00 — 15:00	土	西南
9	乾	15:00 — 17:00	金	西北

日期	星期	时间段	地点	备注
10		17:00 — 19:00		
11		19:00 — 21:00		
12		21:00 — 23:00		

0.3 数据表

数据表 " " /
 " "

- 函数 `mod8_gua(N)` 返回 $\text{mod8_gua}(N) = \begin{cases} N & \text{if } 1 \leq N \leq 8 \\ N \bmod 8 & \text{if } N > 8 \text{ and } (N \bmod 8 \neq 0) \\ 0 & \text{if } N > 8 \text{ and } (N \bmod 8 = 0) \end{cases}$
- 函数 `mod6_yao(N)` 返回 $\text{mod6_yao}(N) = \begin{cases} N & \text{if } 1 \leq N \leq 6 \\ N \bmod 6 & \text{if } N > 6 \text{ and } (N \bmod 6 \neq 0) \\ 0 & \text{if } N > 6 \text{ and } (N \bmod 6 = 0) \end{cases}$

数据表

数据表

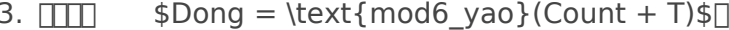
数据表 数据表 数据表 数据表 数据表

1.1 数据表

- 函数 `F1` 返回 `F2` 的数据表
- 函数 `Y` 返回 `M` 的数据表
 - 函数 `Y` 返回 `M` 的数据表
 - 函数 `M` 返回 `D` 的数据表
 - 函数 `D` 返回 `T` 的数据表
 - 函数 `T` 返回 `Y` 的数据表
- 函数 `Num_upper` 返回 `Num_lower` 的数据表
 - 函数 `Num_upper` 返回 `Num_lower` 的数据表
 - 函数 `Num_lower` 返回 `Num_upper` 的数据表
 - 函数 `Dong` 返回 `Num_upper` 的数据表

- 
 - 
 -  “ 4 “
 “ ”

1.2

-  F1  F2
-  \$Count\$  \$T \in [1, 12]\$
- 
- 
 1.  \$Num_{upper} = Count\$  \$= \text{mod8_gua}(Count)\$
 2.  \$Num_{lower} = T\$  \$= \text{mod8_gua}(T)\$
 3.  \$Dong = \text{mod6_yao}(Count + T)\$
-  F1  F2

1.3

-  F1  F2
-  \$SoundCount\$  \$T \in [1, 12]\$
- 
 1.  \$Num_{upper} = SoundCount\$  \$= \text{mod8_gua}(SoundCount)\$
 2.  \$Num_{lower} = T\$  \$= \text{mod8_gua}(T)\$
 3.  \$Dong = \text{mod6_yao}(SoundCount + T)\$
- 

1.4

-  F1  F2

1.4.1

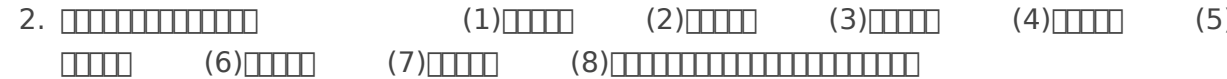
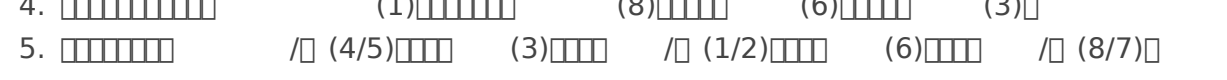
- 
- 
 -  /  / 
 -  \$P_{left}\$  \$P_{right}\$
 -  \$= \text{mod8_gua}(P_{left})\$
 -  \$= \text{mod8_gua}(P_{right})\$
 -  \$= \text{mod6_yao}(P_{left} + P_{right})\$  
- 
 -  “ ” “ ”  “ ” “ ” “ ” 

- $\square \square \square \square \square$
 - $\square \square \square \square \square \quad \$N\$ \square \quad \$N \setminus \text{ge } 11\$ \square \square$
 - $\square \quad \$N\$ \square \square \square \square \square \quad \$ = \text{\texttt{\text{mod8_gua}}}(N / 2)\$ \square \square \square \quad \$ = \text{\texttt{\text{mod8_gua}}}(N / 2)\$ \square \square \square \quad \$ = \text{\texttt{\text{mod6_yao}}}(N)\$ \square$
 - $\square \quad \$N\$ \square \square \square \square \square \quad \$ = \text{\texttt{\text{mod8_gua}}}((N - 1) / 2)\$ \square \square \square \quad \$ = \text{\texttt{\text{mod8_gua}}}((N + 1) / 2)\$ \square \square \square \quad \$ = \text{\texttt{\text{mod6_yao}}}(N)\$ \square$
 - $\square \square \square \square \square \quad \square \square \square \square \square \quad \square$

 1.5

- **F1** `\text{mod8_gua}` `F2` `\text{mod8_gua}`
- **1.5.1** `\text{mod8_gua}`
 - `\text{mod8_gua}` `\text{mod8_gua}` `\text{mod8_gua}`
 - `\text{mod8_gua}` `\text{mod8_gua}` `\text{mod8_gua}`
 - `\text{Dong}` `\text{mod6_yao}` `\text{mod6_yao}` `\text{mod6_yao}`
 - `\text{mod6_yao}` `\text{mod6_yao}` `\text{mod6_yao}`
- **1.5.2** `\text{mod8_gua}` `\text{mod8_gua}`
 - `\text{mod8_gua}` `\text{mod8_gua}` `\text{mod8_gua}`
 - `\text{mod8_gua}` `\text{mod8_gua}` `\text{mod8_gua}`
 - `\text{Dong}` `\text{mod6_yao}` `\text{mod6_yao}` `\text{mod6_yao}` `\text{mod6_yao}`
 - `\text{mod6_yao}` `\text{mod6_yao}` `\text{mod6_yao}` `\text{mod6_yao}`

[illegible]

- 
- 
1. 
2. 
3. 
4. 
5. 
- 

1.7

- [illegible]

- $$\begin{aligned} & \$Gua_{\{dir\}} \in [1, 8] \$ \\ & \begin{aligned} & \begin{aligned} & = 3 \end{aligned} \end{aligned} \end{aligned}$$
 - $$\begin{aligned} & = 6 \end{aligned}$$
 - $$\begin{aligned} & = 4 \end{aligned}$$
 - $$\begin{aligned} & = 2 \end{aligned}$$
 - $$\begin{aligned} & = 5 \end{aligned}$$
 - $$\begin{aligned} & = 1 \end{aligned}$$
 - $$\begin{aligned} & = 7 \end{aligned}$$
 - $$\begin{aligned} & = 8 \end{aligned}$$
 - $$\begin{aligned} & \$T \in [1, 12] \$ \end{aligned}$$
- $$\begin{aligned} & \$ = Gua_{\{obj\}} \$ \end{aligned}$$
 - $$\begin{aligned} & \$ = Gua_{\{dir\}} \$ \end{aligned}$$
 - $$\begin{aligned} & \$ = \text{mod6_yao}(Gua_{\{obj\}} + Gua_{\{dir\}} + T) \$ \end{aligned}$$

1.8

- $$\begin{aligned} & F1 \end{aligned}$$
 - $$\begin{aligned} & F2 \end{aligned}$$
- $$\begin{aligned} & \end{aligned}$$
- $$\begin{aligned} & \$K \$ \end{aligned}$$
 - $$\begin{aligned} & ID \end{aligned}$$
 - $$\begin{aligned} & \$Num_{\{upper\}} = (Y + M + D + K) \$ \end{aligned}$$
 - $$\begin{aligned} & \$Num_{\{lower\}} = (Y + M + D + K + T) \$ \end{aligned}$$
 - $$\begin{aligned} & \$Dong = \text{mod6_yao}(Y + M + D + K + T) \$ \end{aligned}$$
- $$\begin{aligned} & (3) \end{aligned}$$
 - $$\begin{aligned} & (12) \end{aligned}$$
 - $$\begin{aligned} & (1) \end{aligned}$$
 - $$\begin{aligned} & (7) \end{aligned}$$
 - $$\begin{aligned} & \$3+12+1=16 \$ \end{aligned}$$
 - $$\begin{aligned} & 4 \end{aligned}$$
 - $$\begin{aligned} & \$ (16+4)=20 \pmod{8} = 4 \$ \end{aligned}$$
 - $$\begin{aligned} & \$ (20+7)=27 \pmod{8} = 3 \$ \end{aligned}$$
 - $$\begin{aligned} & 6 \end{aligned}$$
 - $$\begin{aligned} & \$ (16+6)=22 \pmod{8} = 6 \$ \end{aligned}$$
 - $$\begin{aligned} & \$ (22+7)=29 \pmod{8} = 5 \$ \end{aligned}$$
 - $$\begin{aligned} & 21 \end{aligned}$$
 - $$\begin{aligned} & \$ (16+21)=37 \pmod{8} = 5 \$ \end{aligned}$$
 - $$\begin{aligned} & \$ (37+7)=44 \pmod{8} = 4 \$ \end{aligned}$$
 - $$\begin{aligned} & \$44 \pmod{6} = 2 \$ \end{aligned}$$

2.1

6

1 6

2.2

	$$(\square + \square + \square + \square) \pmod 6$$		$\square \ 0 \square \ 6$
	$$(\square + \square) \pmod 6$$		$\square \ 0 \square \ 6$
	$$(\square + \square) \pmod 6$$		$\square \ 0 \square \ 6$
	$(\square \square \square \square \square \square \square) \pmod 6$$		$\square \ 0 \square \ 6$
	$\square \square \square \square \square \square \square \square \square \pmod 6$$		$\square \ 0 \square \ 6$
	$$(\square + \square) \pmod 6$$		$\square \ 0 \square \ 6$
	$$(\square + \square + \square) \pmod 6$$		$\square \ 0 \square \ 6$
	$$(\square \square \square + \square \square \square + \square \square) \pmod 6$$		$\square \ 0 \square \ 6$
	$$(\square + \square + \square + \square + \square) \pmod 6$$		$\square \ 0 \square \ 6$

2.3

- 1 /

0 /
- 0 /

1 /

3.1

- F1

F2
-

- o `\begin{cases} \text{Dong} \text{ in } \{1, 2, 3\} \& \text{Dong} \text{ in } \{4, 5, 6\} \& \end{cases}`
- o `\begin{cases} \text{Dong} \text{ in } \{1, 2, 3\} \& \text{Dong} \text{ in } \{4, 5, 6\} \& \end{cases}`

3.2

- `F1` `F2`
- `L_1, L_2, L_3, L_4, L_5, L_6`
- `L_1` `L_6` `L_2, L_3, L_4, L_5`
- `L_2, L_3, L_4` `L_3, L_4, L_5`
- `"` `"`
- `\implies` `**`

3.3 “ ”

- `F1` `F2`
- `111111` `2,3,4` `3,4,5` `111`
- `000000` `2,3,4` `3,4,5` `000`
- `if` `is` `or` `is`

```
if is is or is is:
    # " "
    = [2, 3, 4]
    = [3, 4, 5]
else:
    #
    = [2, 3, 4]
    = [3, 4, 5]
```

- `[ ]`

3.4

- 
 1. 
 2. 

 3. 

\$0 \to 1, 1 \to 0\$
- 
 - 
 - 
 - 

** **

[illegible]

1

4.1

[illegible]

1. $\$E_{\text{other}} \E_{ti}
2. $\$E_{\text{other}} == E_{\text{ti}}$
3. $\$E_{\text{ti}} \E_{other}
4. $\$E_{\text{ti}} \E_{other}
5. $\$E_{\text{other}} \E_{ti}

4.2

[illegible]

1

[illegible]

- [illegible]

- 

					
	 ( / )	 ()	 ()	 ( / )	 ( / )
	 ()	 ( / )	 ( / )	 ()	 ( / )
	 ( / )	 ()	 ( / )	 ()	 ( / )
 	 ( / )	 ( / )	 ()	 ( / )	 ()

4.3

-  
 - 
 - 
-  
 - 
 - 
-  
 - 
 -  /  $\implies$ 
 -  /  $\implies$ 

-   F1   F2 
- 

5.1

-   $\text{\$Sum_all} = \text{\text{\{array of 4 empty boxes\}}} + \text{\text{\{array of 4 empty boxes\}}} + \text{\text{\{array of 4 empty boxes\}}} + \text{\text{\{array of 4 empty boxes\}}} + \text{\text{\{array of 4 empty boxes\}}}$

-  
 -   $\text{\$}\text{\text{\{array of 2 empty boxes\}}} \text{TimeCount} = \lfloor \text{Sum_all} / 2 \rfloor \text{\text{\{array of 2 empty boxes\}}} \text{Sum_all} / 2$
 -   $\text{\$}\text{\text{\{array of 2 empty boxes\}}} \text{TimeCount} = \text{Sum_all} \times 2$

-   $\text{TimeCount} = \text{Sum}_{\{\text{all}\}}$
-  

5.2

 TimeCount 

- 
 -   
-  / 
 -   
- 
 -   

5.3





- 
 - 
 -  


- 
 - 
 -  

-  



-  $F1$  $F2$


6.1

-  

行	列	第1列	第2列	第3列	第4列
2	第1行	第1行第1列	第1行第2列	第1行第3列	第1行第4列
3	第2行	第2行第1列	第2行第2列	第2行第3列	第2行第4列
4	第3行	第3行第1列	第3行第2列	第3行第3列	第3行第4列
5	第4行	第4行第1列	第4行第2列	第4行第3列	第4行第4列
6	第5行	第5行第1列	第5行第2列	第5行第3列	第5行第4列
7	第6行	第6行第1列	第6行第2列	第6行第3列	第6行第4列
8	第7行	第7行第1列	第7行第2列	第7行第3列	第7行第4列

Python

Python

Python

```
# coding: utf-8
from dataclasses import dataclass
from typing import Tuple, List

# 数据
BAGUA_NAMES = ["", " ", " ", " ", " ", " ", " ", " ", " "]
```

```
BAGUA_ELEMENTS = ["", "☯", "☰", "☷", "☵", "☶", "☴", "☱", "☲"]
```

```
# 00000000: 0, 1: 000000000000[00, 00, 00]
```

```
BAGUA_YAO = {  
    1: [1, 1, 1], # ☰ (00)  
    2: [1, 1, 0], # ☶ (00, 00, 00)  
    3: [1, 0, 1], # ☱ (00, 00, 00)  
    4: [1, 0, 0], # ☷ (00, 00, 00)  
    5: [0, 1, 1], # ☲ (00, 00, 00)  
    6: [0, 1, 0], # ☵ (00, 00, 00)  
    7: [0, 0, 1], # ☴ (00, 00, 00)  
    8: [0, 0, 0], # ☳ (00)  
}
```

```
def mod8_gua(n: int) -> int:  
    r = n % 8  
    return 8 if r == 0 else r
```

```
def mod6_yao(n: int) -> int:  
    r = n % 6  
    return 6 if r == 0 else r
```

```
def get_gua_by_yao(yao: List[int]) -> int:  
    for g, y in BAGUA_YAO.items():  
        if y == yao:  
            return g  
    return 8
```

```
@dataclass  
class HexagramResult:  
    upper_gua: int          # 00000 (1-8)  
    lower_gua: int          # 00000 (1-8)  
    dong_yao: int          # 000 (1-6)  
    ti_gua: int            # 00000 (1-8)  
    yong_gua: int          # 00000 (1-8)  
    ti_is_upper: bool      # 00000000  
    hu_upper: int          # 000000 (1-8)  
    hu_lower: int          # 000000 (1-8)  
    bian_upper: int        # 00000000 (1-8)  
    bian_lower: int        # 00000000 (1-8)
```

```

def deduce_meihua(upper: int, lower: int, dong: int) -> HexagramResult:
    # 1. ䷀䷁䷂ (䷀䷁䷂, ䷀0䷂5)
    six_lines = BAGUA_YAO[lower] + BAGUA_YAO[upper]

    # 2. ䷀䷁ (䷀䷁䷂䷃䷄䷅䷆䷇䷈䷉䷊䷋䷌䷍䷎䷏)
    if dong in [1, 2, 3]:
        ti_gua = upper
        yong_gua = lower
        ti_is_upper = True
    else:
        ti_gua = lower
        yong_gua = upper
        ti_is_upper = False

    # 3. ䷀䷁ (䷀䷁䷂䷃䷄䷅䷆䷇䷈䷉䷊䷋䷌䷍䷎䷏)
    bian_lines = list(six_lines)
    target_idx = dong - 1
    bian_lines[target_idx] = 1 if bian_lines[target_idx] == 0 else 0
    bian_lower = get_gua_by_yao(bian_lines[0:3])
    bian_upper = get_gua_by_yao(bian_lines[3:6])

    # 4. ䷀䷁䷂䷃䷄䷅䷆䷇䷈䷉䷊䷋䷌䷍䷎䷏“䷀䷁䷂䷃䷄䷅䷆䷇䷈䷉䷊䷋䷌䷍䷎䷏”䷀䷁䷂
    if (upper == 1 and lower == 1) or (upper == 8 and lower == 8):
        # ䷀䷁䷂䷃䷄䷅䷆䷇䷈䷉䷊䷋䷌䷍䷎䷏
        hu_lower = get_gua_by_yao([bian_lines[1], bian_lines[2], bian_lines[3]])
        hu_upper = get_gua_by_yao([bian_lines[2], bian_lines[3], bian_lines[4]])
    else:
        # ䷀䷁ (䷀䷁䷂䷃䷄䷅䷆䷇䷈䷉䷊䷋䷌䷍䷎䷏)
        hu_lower = get_gua_by_yao([six_lines[1], six_lines[2], six_lines[3]])
        hu_upper = get_gua_by_yao([six_lines[2], six_lines[3], six_lines[4]])

    return HexagramResult(
        upper_gua=upper,
        lower_gua=lower,
        dong_yao=dong,
        ti_gua=ti_gua,
        yong_gua=yong_gua,
        ti_is_upper=ti_is_upper,
        hu_upper=hu_upper,

```

```
    hu_lower=hu_lower,  
    bian_upper=bian_upper,  
    bian_lower=bian_lower  
)
```

“

[illegible]

--	--	--	--	--

[illegible][illegible]

Software Architecture &

Mathematical Engine Specification

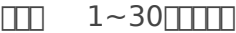
- `src/engine/math_engine.rs`
`src/engine/math_engine.rs`
`src/engine/math_engine.rs`
`src/engine/math_engine.rs`
- `src/engine/math_engine.rs`
`src/engine/math_engine.rs`
`src/engine/math_engine.rs`
`src/engine/math_engine.rs`
- `src/engine/math_engine.rs`
`src/engine/math_engine.rs`
`src/engine/math_engine.rs`
`src/engine/math_engine.rs`

[illegible]

--	--

**//

”**

- 
 - 
 -  1~30  (1)  (30) 
 -  31~60  (31)  (60) 
 -  61~90  (1)  (30) 
 -  91~120  (31)  (60) 
 - 

3.

-  $S_global \in [1, 4320]$ 
-  $s_local \in [1, 12]$

-  $S_global = (Y_global - 1) \times 12 + s_local$ 
-  
 -  30 
 -  **$S_global \% 2 == 1$**   **(1)**   **(30)**

 -  **$S_global \% 2 == 0$**   **(31)**  
(60) 
 -  60  360 

2.2

 $N_year \in [1, 129600]$ 

- $H = (N_year - 1) // 10800 + 1$ ( $[1, 12]$)
- $Y_global = (N_year - 1) // 360 + 1$ ( $[1, 360]$)
- $S_global = (N_year - 1) // 30 + 1$ ( $[1, 4320]$)
- $YearInShi = (N_year - 1) \% 30 + 1$ ( $[1, 30]$)
- $s_local = (S_global - 1) \% 12 + 1$ ( $\sim$ )
- $y_local = (Y_global - 1) \% 30 + 1$

 $GZ \in [0, 59]$ 

-  S_global  $GZ = YearInShi - 1$  0  29 
-  S_global  $GZ = YearInShi + 29$  30  59 

2.3

Anchor Points



└─ 0214900000 0215500000000000
└─ 02156000000000001100000000— 000000000002357000
|

└─ 0216100000 02172000000000
└─ 0182000000000000
└─ 0183000000000000
└─ 0184000000000000
└─ 022660000000000270000000— 0000000000960000

[illegible]

項目	年次	年次	N_year	年次
1980年	180年・2156年	11年	64,661	2357年 (BCE 2357)
1981年	180年・2158年	21年	64,731	2287年 (BCE 2287)
1982年	180年・2159年	24年	64,764	2254年 (BCE 2254)
1983年	182年・2176年	16年	65,266	1752年 (BCE 1752)
1984年	183年・2197年	16年	65,896	1122年 (BCE 1122)
1985年	184年・2266年	27年	67,977	960年 (CE 960)
1986年	184年・2267年	14年	68,094	1077年 (CE 1077)

[illegible]

Offset = 67017

 0

 1

 1

- `CE` `N_year`
 - `CE > 0` `N_year = CE + 67017`
 - `BCE > 0` `N_year = 67018 - BCE`
- `N_year`
 - `N_year > 67017` `CE = N_year - 67017`
 - `N_year <= 67017` `BCE = 67018 - N_year`



3.1

-  
-  
 -  “”
 -  
 -   
 - $60 \times 6 = 360$   



-   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. 
-   31.  32.  33.  34.  35.  36.  37. 
 38.  39.  40.  41.  42.  43.  44.  45.  46.  47.  48.  49. 
 50.  51.  52.  53.  54.  55.  56.  57.  58.  59.  60. 

3.2 12

-  1
 -  **A** / 
 - 
 - 
 - 
 - 
 - 
 - 
 - 
 - 
 - 
 - 
 - 
 - 
 - 

- **B** XXXXXXXXXXXXXXXXXXXX □
 - XXXXXXXXXXXXXXXXXXXX $5 \times 12 \times 5 = 60$ □
- [XXXXXXXXXXXX] □ **A** XXXXXX
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX

3.3 XXXXXXXXXX 360□

- XXXXXXXXXXXXXXXXXXXXXXXXXXXX □
 - XXXXXX 360 □
 - XXXXXX **360**□ XXXXXXXXXX $6 \times 360 = 2,160$ □
 - XXXXXX $Y \in [1, 360]$ □
 - □□ $Gualdx = (Y - 1) // 6 \in [0, 59]$ □□□□ $Gualdx + 1$ □□
 - □□ $Yaoldx = (Y - 1) \% 6 + 1 \in [1, 6]$ □□□□□□
 - XXXXXX XXXXXX $Gualdx + 1$ □
 - XXXXXX XXXXXX $Yaoldx$ □□□□□□
- [□□□ 2□□□□□□□] □
 - XXXXXXXXXXXXXXXXXXXX “XXXXXXXXXX”
XXXXXXXXXXXXXXXXXXXXXXXXXXXX
 - [XXXXXXXXXXXX] □ □ “XXXXXXXXXX” ”XXXXXXXXXXXXXXXXXXXX”

3.4 XXXXXXXXXX 4320□

□□ 12 □ 360XXXXXXXXXX 12 □□□

- [□□□ 3□□□□□□□]
 - **A** XXXXXXXXXXXXXXXXXXXX □
 - XXXXXXXXXX
 - XXXXXXXXXX 6 □□□ 6 □ 1~6□□□
 - □□ 6 XXXXXXXXXXXXXXXXXXXX 6 □ 7~12□□□
 - **B** XXXXXX □
 - XXXXXXXXXX
 - **C** XXXXXX □
 - □□ 12 □ 30 □ 360 XXXXXXXXXXXX
 - [XXXXXXXXXXXX] □ □ **A** XXXXXX XXXXXXXXXXXXXXXXXXXX

3.5 XXXXXXXXXX 129600□

□□ 30 XXXXXX

- [□□□ 4□□□□□□□]
 - XXXXXXXXXXXXXXXXXXXXXXXXXXXX □
 - □□ 60 XXXXXXXXXX

-  $N_{\text{kaiwu}} = 27,000$
 -  $N_{\text{biwu}} = 113,400$
 -  $\Delta T_{\text{active}} = N_{\text{biwu}} - N_{\text{kaiwu}} = 113,400 - 27,000 = 86,400$
-  $8 \times 10,800$   

- $\Delta T_{\text{inactive}} = 129,600 - 86,400 = 43,200$ 4
- $4 \times 10,800$
- $N_{\text{year}} = 2026 + 67017 = 69,043$
- $H = (69043 - 1) // 10800 + 1 = 7$
- $Y = (69043 - 1) // 360 + 1 = 192$
- $S = (69043 - 1) // 30 + 1 = 2302$
- 13
- $69,043 - 27,000 = 42,043$
- $113,400 - 69,043 = 44,357$
-



1837-1865 1923-1935 2105-2113

5.1

“ 1857

		1-10	$4 \times 10 = 40$	
		1-12	$4 \times 12 = 48$	
$\times$	$\times$	10	$16 \times 10 = 160$	
$\times$	$\times$	12	$16 \times 12 = 192$	


 “” “” 
 160 
 48 
112 

192 
 40 
152 

$$\begin{aligned} \square\square\square\square &= 160 \times 192 = 30,720 \\ \square\square\square\square &= 160 - 48 = 112 \\ \square\square\square\square &= 192 - 40 = 152 \\ \square\square\square\square &= 112 \times 152 = 17,024 \\ \square\square\square\square &= 152 \times 112 = 17,024 \\ \square\square\square\square &= 17,024 \times 17,024 = 289,816,576 \end{aligned}$$

[illegible]

5.2

□	□□□	□□□	□□□□
□□	□□□□□	tone = 1□ 2□ 3□ 4	□□□□□□□□ □□ 4 × 4 □□□
□□□	□□□□□	opening = 1□ 2□ 3□ 4	□□□□□□□□ □□ 4 × 4 □□□
□□□	□□□□□□□□ □□□□	voice = 1...10	□□□□□□
□□□	□□□□□□□□ □□□□□□□	sound = 1...12	□□□□□□□
□□	□□ / □□	odd / even	□□□□□□□□ / □
□□	□ / □	open / closed	□□□□□□□□ □□□ /□□

 $4 \times \text{100}$ $\times \text{400}$ = 160  $4 \times \text{100}$ $\times \text{1000}$ =
 192  112  152  $10 \times 12 =$
 120  "100"  $12 \times 10 = 120$  "100"  $/ \text{100000000}$ $16 \times 240 = \mathbf{3,840}$ 
 151-153  1861-1863 

5.3

```

sky = (sky ∈ {0,1,2,3}, tone ∈ {0,1,2,3}, voice ∈ 1..10)
earth = (earth ∈ {0,1,2,3}, opening ∈ {0,1,2,3}, sound ∈ 1..12)

```

1. sky tone earth opening
2. voice sound
- 3.
- 4.
- 5.
6. use_voice_index use_sound_index usable=false
7. animal_number = use_voice_index * use_sound_index 1-17,024

[illegible]

/ / 1850-1851

5.4

[illegible][illegible]

”□ 2105-2107

ㄅ	ㄆ	ㄇ
ㄅ	ㄆ	yuan
ㄆ	ㄆ	hui
ㄇ	ㄆ	yun
ㄇ	ㄆ	shi

□□□□□	(yuan, hui, yun, shi)
-------	-----------------------

[illegible]

yun_gua(year) /

[illegible]5.5

--	--	--	--

-

- “ ” tone=1..4  “ ”opening=1..4
 /
-  /


-  1837-1859 1861-1867 1923-1935 2105-2113 1853-1855
 /



Python

```
# coding: utf-8

from dataclasses import dataclass
from typing import Tuple, Dict, Any

# 1. 字表
TIAN_GAN = ["", "干", "坤", "坎", "离", "震", "兑", "艮", "巽", "兑", "坎", "坤", "干"]
DI_ZHI = ["", "子", "丑", "寅", "卯", "辰", "巳", "午", "未", "申", "酉", "戌", "亥", "子"]

# 六十甲子
SIXTY_JIAZI = [
    "甲子", "乙丑", "丙寅", "丁卯", "戊辰", "己巳", "庚午", "辛未", "壬申", "癸酉",
    "甲戌", "乙亥", "丙子", "丁丑", "戊寅", "己卯", "庚辰", "辛巳", "壬午", "癸未",
    "甲申", "乙酉", "丙戌", "丁亥", "戊子", "己丑", "庚寅", "辛卯", "壬辰", "癸巳",
    "甲午", "乙未", "丙申", "丁酉", "戊戌", "己亥", "庚子", "辛丑", "壬寅", "癸卯",
    "甲辰", "乙巳", "丙午", "丁未", "戊申", "己酉", "庚戌", "辛亥", "壬子", "癸丑",
    "甲寅", "乙卯", "丙辰", "丁巳", "戊午", "己未", "庚申", "辛酉", "壬戌", "癸亥"
]

# 六十卦
HUI_GUA = ["", "干", "坎", "艮", "震", "巽", "离", "兑", "坤", "乾", "坎", "艮", "震", "巽", "离", "兑", "坤"]

# 六十卦
SIXTY_ROUND_GUA = [
    # 六十卦

```

```

"0", "0", "0", "0", "0", "00", "0", "00", "00", "0",
"00", "00", "0", "0", "00", "0", "0", "0", "00", "00",
"0", "0", "0", "0", "00", "0", "00", "00", "00", "0",
# 00000
"0", "00", "0", "0", "0", "0", "0", "0", "0", "0",
"00", "0", "0", "0", "0", "0", "0", "0", "00", "0",
"0", "0", "0", "0", "0", "0", "0", "0", "0", "0"
]

```

```

# 00000000000000000000
CE_OFFSET = 67017

```

```

@dataclass
class HuangjiCoordinate:
    absolute_year: int    # 000000 [1, 129600]
    hui_index: int        # 000 [1, 12]
    hui_name: str         # 000 (0~0)
    hui_gua: str          # 00 (0000)
    yun_global: int       # 00000 [1, 360]
    yun_local: int        # 00000 [1, 30]
    yun_gua: str          # 000
    yun_yao: int          # 00000 [1, 6]
    shi_global: int       # 00000 [1, 4320]
    shi_local: int        # 00000 [1, 12]
    shi_name: str         # 000 (0~0)
    year_in_shi: int      # 00000 [1, 30]
    ganzhi: str           # 0000
    year_gua: str         # 00000 (00000000)
    is_kaiwu: bool        # 0000000000
    ce_year: int          # 0000 (00000000)

```

```

class HuangjiEngine:
    @staticmethod
    def ce_to_absolute_year(ce: int) -> int:
        # 000000000000
        if ce > 0:
            return ce + CE_OFFSET
        elif ce < 0:
            return 67018 + ce
        else:

```

```
raise ValueError("0000000000000000 -1 000001001 0000100")
```

```
@staticmethod
```

```
def absolute_year_to_ce(ay: int) -> int:
```

```
    # 000000000000
```

```
    if ay > CE_OFFSET:
```

```
        return ay - CE_OFFSET
```

```
    else:
```

```
        return ay - 67018
```

```
@classmethod
```

```
def calculate(cls, absolute_year: int) -> HuangjiCoordinate:
```

```
    # 000000000000000000000000
```

```
    if not (1 <= absolute_year <= 129600):
```

```
        raise ValueError("00000000 [1, 129600] 000")
```

```
    ay = absolute_year
```

```
    # 1. 000
```

```
    h_idx = (ay - 1) // 10800 + 1
```

```
    h_name = DI_ZHI[h_idx]
```

```
    h_gua = HUI_GUA[h_idx]
```

```
    # 2. 000
```

```
    y_global = (ay - 1) // 360 + 1
```

```
    y_local = (y_global - 1) % 30 + 1
```

```
    # 0000000000600000000000600000
```

```
    gua_idx = (y_global - 1) // 6
```

```
    yao_idx = (y_global - 1) % 6 + 1
```

```
    y_gua = SIXTY_ROUND_GUA[gua_idx]
```

```
    # 3. 000
```

```
    s_global = (ay - 1) // 30 + 1
```

```
    s_local = (s_global - 1) % 12 + 1
```

```
    s_name = DI_ZHI[s_local]
```

```
    # 4. 000000
```

```
    year_in_shi = (ay - 1) % 30 + 1
```

```
    if s_global % 2 == 1:
```

```

        # 0~29
        gz_idx = year_in_shi - 1
    else:
        # 30~59
        gz_idx = year_in_shi + 29
    ganzhi_str = SIXTY_JIAZI[gz_idx]

    #
    year_gua_str = SIXTY_ROUND_GUA[gz_idx]

    # 5. (27000~113400)
    is_kaiwu_status = (27000 <= ay <= 113400)

    # 6.
    ce = cls.absolute_year_to_ce(ay)

    return HuangjiCoordinate(
        absolute_year=ay,
        hui_index=h_idx,
        hui_name=h_name,
        hui_gua=h_gua,
        yun_global=y_global,
        yun_local=y_local,
        yun_gua=y_gua,
        yun_yao=yao_idx,
        shi_global=s_global,
        shi_local=s_local,
        shi_name=s_name,
        year_in_shi=year_in_shi,
        ganzhi=ganzhi_str,
        year_gua=year_gua_str,
        is_kaiwu=is_kaiwu_status,
        ce_year=ce
    )

#
if __name__ == "__main__":
    engine = HuangjiEngine()

    # (2357)

```

```
yao_ay = 64661
res_yao = engine.calculate(yao_ay)
print(f"aaaa(64661): a={res_yao.hui_name}, a={res_yao.yun_global}, a
={res_yao.shi_global}({res_yao.shi_name}), aa={res_yao.year_in_shi}, aa={res_yao.ganzhi}")
assert res_yao.ganzhi == "aa", "aaaaaaaaaaaa"
assert res_yao.shi_global == 2156, "aaaaaaaaaaaa2156aa"
assert res_yao.yun_global == 180, "aaaaaaaaaaaa180aa"
assert res_yao.ce_year == -2357, "aaaaaaaaaaaa2357aa"

# aaaaaaaa (aa960a)
song_ay = 67977
res_song = engine.calculate(song_ay)
print(f"aaaa(67977): a={res_song.hui_name}, a={res_song.yun_global}, a
={res_song.shi_global}({res_song.shi_name}), aa={res_song.year_in_shi}, aa={res_song.ganzhi}")
assert res_song.ganzhi == "aa", "aaaaaaaaaaaa"
assert res_song.shi_global == 2266, "aaaaaaaaaaaa2266aa"
assert res_song.ce_year == 960, "aaaaaaaaaaaa960aa"

# aaaa2026a (aaaa)
ay_2026 = 69043
res_2026 = engine.calculate(ay_2026)
print(f"aa2026a(69043): a={res_2026.hui_name}, a={res_2026.yun_global}, a
={res_2026.shi_global}({res_2026.shi_name}), aa={res_2026.year_in_shi}, aa={res_2026.ganzhi}")
assert res_2026.ganzhi == "aa", "aa2026aaaaaaaa"
assert res_2026.ce_year == 2026, "aaaaaa2026aa"

print("aaaaaaaaaaaaaaaaaaaa")
```



aa	aaaaaaaa	aaaaaaaaaaaaaa
aaaa	12a 30aaaaaa 12aa 30aa 12aa 30aa	aaaaaa aaaaaaaaaaaaaa
aaaa	aaaaaaaaaaaa	aaaaaa aaaaaaaaaaaaaaaaaa

項目	計算式	計算結果
項目	計算式 /項目 vs	計算式 /項目 計算式 項目
項目	計算式 360 vs	計算式 Gualdx = (Y-1)//6 Yaoldx = (Y-1)%6 + 1
項目	計算式 12 vs	計算式 12
項目	計算式 vs	計算式
項目	計算式 27000 計算式 113400 項目	計算式 項目 8 項目 86,400 項目
項目	計算式	計算式 計算式

3.
-

960

3,840

1.2

“”

“”

““”

“

2.1

				/		...
				/		...
				/		...

数据	数据	数据	数据	数据	数据	数据 数据
数据	数据	数据	数据	数据 / 数据	数据	数据 数据 ...

- 数据
 - 数据 7 数据 3 数据
 - 数据 数据 6 数据
 - 数据 10数据 160数据

2.2 数据 . 数据 . 数据

数据

数据	数据	数据	数据	数据	数据	数据 数据
数据	数据	数据	数据 数据	数据 / 数据	数据	数据 数据 ...
数据	数据	数据	数据 数据	数据 / 数据	数据	数据 数据 ...
数据	数据	数据	数据 数据	数据 / 数据	数据	数据 数据 ...
数据	数据	数据	数据 数据	数据 / 数据	数据	数据 数据

- 数据
 - 数据 9 数据 3
 - 数据 数据 数据 4 数据
 - 数据 12数据 192数据

数据

数据

数据

"数据"*

3.1 数据

-  10  10  10  10 
 ○   = $(10 + 10 + 10 + 10) * 4 = 160$
-  12  12  12 
 ○   = $(12 + 12 + 12 + 12) * 4 = 192$
-   = $160 * 192 = 30,720$


3.2

112 152

 “

”

-  
 ○  160  12  $12 * 4 = 48$ 
 ○  = $160 - 48 = 112$ 
-  
 ○  192  10  $10 * 4 = 40$ 
 ○  = $192 - 40 = 152$ 
-  
 ○  $160 + 192 = 352$ 
 ○  $112 + 152 = 264$ 



4.1



1.  •  
 •  112  152   () = $112 * 152 = 17,024$
 • 
2.  •  
 •  152  112   () = $152 * 112 = 17,024$
 • 

289,816,576□

$$17,024 = 289,816,576 \square\square \quad \square\square\square\square\square\square\square\square\square\square\square\square$$
$$= 17,024 *$$
[illegible]

“

[illegible][illegible][illegible]

24

4.3

-   $7 \times 152 = 1,064$ 
-   $9 \times 112 = 1,008$ 
-  $3,840 \times 10 = 38,400$ 

--	--	--

[illegible]

5.1 “ ” “ ” ”

--	--	--	--	--

	 	 	
---	--	---	---

項目	数量 個	数量 個	数量 個
数量	数量 27,000 個 数量	数量 数量	数量
数量	数量 数量 8個 数量 86,400 個	数量 9個 / 70個 数量 264 個 7 数量 9 個	数量
数量	数量 113,400 数量 数量	数量 数量	数量
数量	数量 数量 43,200 個	数量 数量 3 数量	“数量”

数量 “

数量

数量 ”

数量

数量

数量 2/3 数量 1/3 数量

5.2 数量

数量 数量

1数量 2数量 3数量 4数量 数量

1. 数量 1数量

2. 数量 2数量

3. 数量 3数量

4. 数量 4数量

数量

数量 数量

数量

6.1 数量 Python / Pydantic 数量

```
# coding: utf-8
from dataclasses import dataclass
```

```

from typing import List, Dict, Optional
from enum import Enum

class TianShengTone(Enum):
    PING = "平" # 阴平 · 一
    SHANG = "上" # 阴上 · 二
    QU = "曲" # 阴去 · 三
    RU = "入" # 阴入 · 四

class DiYinMouth(Enum):
    KAI = "开" # 阴开 · 一
    FA = "发" # 阴发 · 二
    SHOU = "收" # 阴收 · 三
    BI = "闭" # 阴闭 · 四

@dataclass(frozen=True)
class TianShengEntity:
    index: int # 1 ~ 16
    char: str # 普通话四声, 阴, 阳, 阴
    tone: TianShengTone # 阴 / 阳 / 阴 / 阳
    pishi: str # 阴 / 阳
    is_usable: bool # 是否可用 (True, False)

@dataclass(frozen=True)
class DiYinEntity:
    index: int # 1 ~ 16
    char: str # 普通话四声, 阴, 阳, 阴
    mouth: DiYinMouth # 阴 / 阳 / 阴 / 阳
    qingzhuo: str # 阴 / 阳
    is_usable: bool # 是否可用 (True, False)

@dataclass(frozen=True)
class HuangjiSoundHarmonicResult:
    tian_sheng: TianShengEntity
    di_yin: DiYinEntity
    is_harmonic: bool # 是否和谐 (True)
    tian_chang_bian: int # 天常变 (7 * 152 = 1064 个)
    di_he_hua: int # 地合化 (9 * 112 = 1008 个)
    tong_shu: int # 总数 (289,816,576)

```

6.2 断言

Invariant Assertions

测试用例

```
def test_huangji_sound_invariants():  
    # 1. 断言  
    TI_YANG = (10 + 10 + 10 + 10) * 4 # 160  
    TI_YIN = (12 + 12 + 12 + 12) * 4 # 192  
    assert TI_YANG == 160  
    assert TI_YIN == 192  
  
    YONG_YANG = TI_YANG - (12 * 4) # 160 - 48 = 112  
    YONG_YIN = TI_YIN - (10 * 4) # 192 - 40 = 152  
    assert YONG_YANG == 112  
    assert YONG_YIN == 152  
  
    # 2. 断言  
    ALL_NUMBER = TI_YANG * TI_YIN  
    assert ALL_NUMBER == 30720  
  
    # 3. 断言  
    DONG_SHU = YONG_YANG * YONG_YIN  
    ZHI_SHU = YONG_YIN * YONG_YANG  
    assert DONG_SHU == 17024  
    assert ZHI_SHU == 17024  
  
    # 4. 断言  
    TONG_SHU = DONG_SHU * ZHI_SHU  
    assert TONG_SHU == 289816576 # 断言
```

测试用例

测试用例

1. 断言
2. 断言

3. “ ” “ ”

4.

“ ”