

$$1: \begin{array}{c|c} 1 & 1 \\ \hline 1 & 1 \end{array}$$

$$1 \xrightarrow{x} 1 \quad \begin{array}{l} \text{uuu, uul, ulu} \\ \text{uuu, ulu, ulu} \end{array}$$

$$6: \begin{array}{c|c} 6 & 6 \\ \hline 6 & 6 \end{array}$$

$$2: \begin{array}{c|c} 2 & 5 \\ \hline 5 & 2 \end{array}$$

$$x: 2 \rightarrow 5$$

$$y: 2 \rightarrow 5$$

$$\begin{array}{l} \text{uuu, uul, ulu} \\ \text{uuu, ulu, ulu} \end{array}$$

$$3: \begin{array}{c|c} 3 & 3 \\ \hline 3 & 3 \end{array}$$

$$x: 3 \rightarrow 3$$

$$y: 3$$

$$4: \begin{array}{c|c} 4 & 4 \\ \hline 4 & 4 \end{array}$$

$$x: 4 \rightarrow 4$$

$$y: 4 \rightarrow 4$$

$$\begin{array}{l} \text{uuu, ulu, ulu} \\ \text{uuu, ulu, ulu} \end{array}$$

$$5: \begin{array}{c|c} 5 & 2 \\ \hline 2 & 5 \end{array}$$

$$5 \rightarrow 2$$

$$5 \rightarrow 2$$

-1-

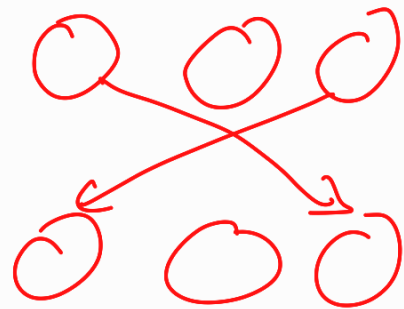
$$\text{uuu, ulu, ulu}$$

$$\xrightarrow{x} \text{uuu,}$$

$$\begin{array}{c} 1 \\ \hline 1 \end{array}$$

$$\xrightarrow{x} \begin{array}{c} \text{uuu} \\ \text{uuu} \\ \text{uuu} \end{array} - 1$$

$$\begin{array}{c} \text{uuu} \\ \text{uuu} \\ \text{uuu} \end{array}$$

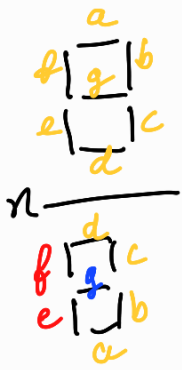


$$\xrightarrow{x} \begin{array}{c} \text{uuu} \\ \text{uuu} \\ \text{uuu} \end{array}$$

bleibt

Keine gute Idee

idea: 7 segments
0, 1

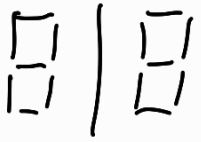


$u-v, 1-1, 1-1$

$u-v, 1-1, 1-1$

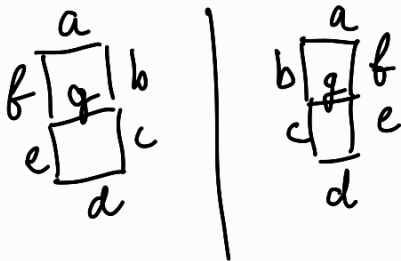
$u-v, 1-1, 1-1$

$u-v, 1-1, 1-1$

$$\begin{array}{l} a \rightarrow d \\ b \rightarrow c \\ c \rightarrow b \\ d \rightarrow a \\ e \rightarrow f \\ f \rightarrow e \\ g \rightarrow g \end{array}$$


4
H

test: 4: b, c, f, g
n: c, b, e, g ✓



$$\begin{array}{l} a \rightarrow a \\ b \rightarrow b \\ c \rightarrow c \\ d \rightarrow d \\ e \rightarrow e \\ f \rightarrow f \\ g \rightarrow g \end{array}$$

Gate Idea

pseudo code:

```
enum Segment [ A, B, C, D, E, F, G ]
```

```
class Digit  
    segment = Segment [ 7 ]
```

X

```
displayA()  
display FGB()  
display EDC()
```

```
class Number:  
    digits = Digit [ 4 ]  
    Number(int)
```

X

```
displayA()  
for d in digits  
    d.displayA()  
"  
"
```

```
class Mirror
```

```
mirror-x (Number I, Iy) return Ix, Ixy  
mirror-y (Number I, Ix) return Iy, Ixy
```

```
class Printer:
```

```
int Number I  
          Ix  
          Iy  
Mirror m Ixy
```

X

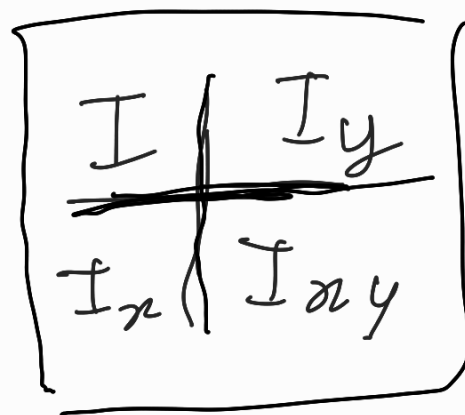
```
menu()  
while ( )  
    switch  
    case x: →  
    y:  
    xy:  
    q, exit, quit ...:
```

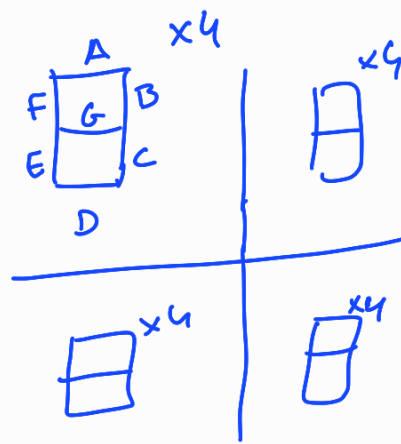
```
mirror.x-asc(I, Iy) return Ix, Ixy
```

X

```
displayA (Number)  
display FGB (Number)  
display EDC (Number)
```

The Mirror =





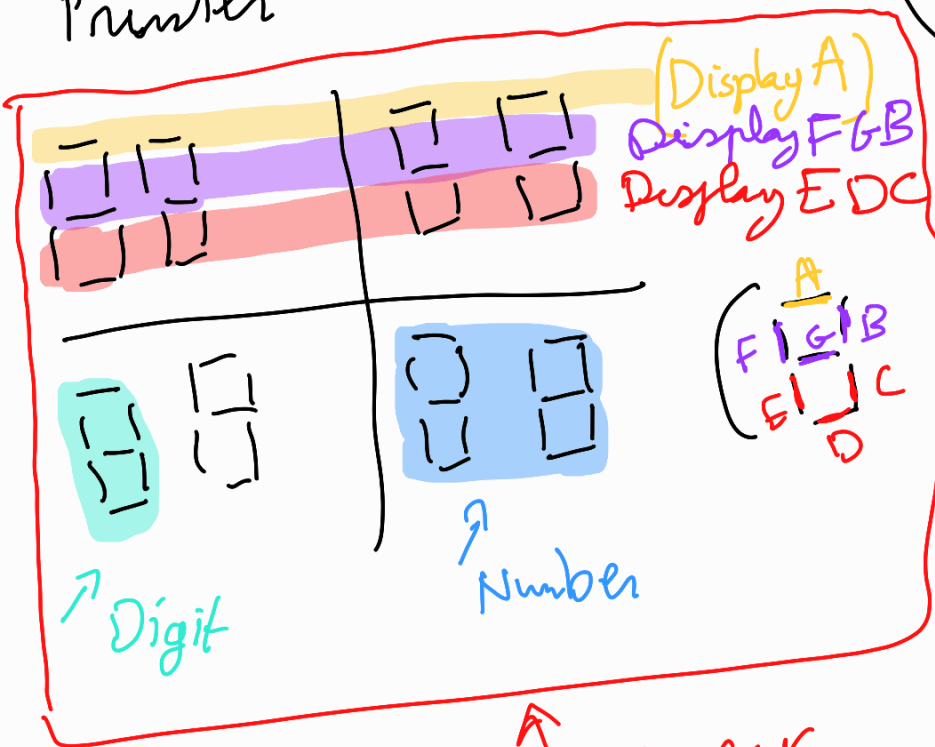
if defined
↓

display()
 1. displayA(I) + " | " + displayA(Ig)
 displayFGB(I) + " | " + displayFGB(Ig)
 displayEDC(I) + " | " + displayEDC(Ig) ∴ Ix, Iyg

displayA(Number):
 for displayA(Digit) + " | " ;

Number = Digit[4]

Printer



Mirror

all display's method
 should be in
 one class
 "Printer" + Menu
 in main
 → SOLID