Lee

① 1.
```
add x15, x12, x11
NOP
NOP
ld x13, 4(x15)
ld x12, 0(x 2)
NOP
or x13, x15, 13
NOP
NOP
sd X13. 0 (X15)
```

2. No. Because the first one and second one have RAW
four and two have RAW
five and four have RAW

3. code can execute here even don't have hazard detection Unit. ~~Because~~ if we do forward. These code are availabe to keep run., Because they don't have RAW (~~three and two~~). , that meaning the one of instruction perform memory operation.

② T, NT, T, T, NT.

1. Always-taken.	Always-Not-taken.
T, NT, T, T, NT.	T, NT, T, T, NT
T T T T T.	NT, NT, NT, NT, NT.
$\frac{3}{5}$ = 60% = 0.6	$\frac{2}{5}$ = 40% = 0.4.

2. T, NT, T, T, NT.

Begin from NT. → NT

NT in get NT → NT.

T in get NT → NT.

T in get NT → NT.

T, NT, T, T $\frac{1}{4} = 25\% = 0.25$.

NT, NT, NT, NT

3.

		1st	2nd	3rd	4.
~~NT~~	~~T~~	NT	NT	T	T
~~NT~~	~~NT~~	NT	T	T	T
~~NT~~	~~T~~	NT	NT	T	T
~~T~~		NT	T	T	T
~~NT~~	~~T~~	T	T	T	T

T NT T T NT

T T T T T $\frac{3}{5} = 60\% = 0.6$

③ 0x03, 0xb4, 0x2b, 0x02, 0xbe, 0x58, 0xbf
0x0e, 0x1f, 0xb5, 0xbf, 0xba, 0x2e, 0xce

~~Binary Tag Set offset word.~~

~~0x03, 00000011, 0000, 001, 1, 3.~~

0x03	00000011	0000	001	1	3	
0x64	10110100	1011	010	0	~~[illegible]~~	180
0x2b	00101011	0010	101	1	43	
0x02	00000010	0000	001	0	2.	
0xbe	10111110.	1011	111	0	190	
0x58	01011000	0101	100	0	88	
0xbf	10111111	1011	111	1	191	
0x0e	00001110.	0000	111	0	14.	
0x1f	00011111	0001	111	1	31	
0xb5	10110101.	1011	010	1	181	
0xbf	10111111	1011	111	1	191	
0xba	10111010	1011	101	0	186	
0x2e	00101110.	0010	111	0	46	
0xce	11001110.	1100	111	0	206	

set offset.

0x03	00000011	0000	001	1	3	miss	1	0	
0xb4	10110100	1011	010	0	180	miss	2	0	
0x2b	00101011	0010	101	1	43	miss	5	0.	
0x02	00000010	0000	001	0	2	hit	1	0	
0xbe	10111110	1011	111	0	190	miss	7	0.	
0x58	01011000	0101	100	0	88	miss	4	0	
0xbf	10111111	1011	111	1	191	hit	7	0.	
0x0e	00001110.	0000	111	0	14.	miss	7	1.	
0x1f	00011111	0001	111	1	31	miss	7	2.	set 7 is full
0xb5	10110101	1011	010	1	181	hit	2	0	
0xbf	10111111	1011	111	1	191	hit	7	0.	
0xba	10111010	1011	101	0	186	miss	5	1.	
0x2e	00101110	0010	111	0	46	miss	7	1.	lease used=7.1
0xce	11001110.	1100	111	0	206	miss	7	2	lease used=7.2.

④. $\begin{cases} \text{Virtual Address Size} & \text{32 bits.} \\ \text{Page size} & \text{8KiB.} \\ \text{Page Table Entry Size} & \text{4 bytes} \end{cases}$

1.

8KiB=?. $\log_2(8 \cdot 2^{10}) = 13$ So the ~~Pag~~ page size = 2^{13}.

$$\text{number of page} = \frac{\text{Virtual Address Size}}{\text{page size}} = \frac{2^{32}}{2^{13}} = 2^{19}$$

page Table Entry Size = 4 byte

$2^{19} \cdot 4 = 2097152 = 2048$ KB = 2MB.

need running five processes $5 \cdot 2$MB = 10MB.

2. with up to 256 entires at the 1st level.

$256 = 2^8$ $2^8 \cdot 4 \times 2 = 2048 = 2^{11}$

$\frac{2^8}{2} = 2^7$ $128 \times 2048 = 2^7 \cdot 2^{11} = 2^{18}$

Min $\begin{cases} 2^{18} \cdot 4 = 2^{18} \cdot 2^2 = 2^{20} = \frac{2^{20}}{2^{10}} = 2^{10}\text{ KiB} = \frac{2^{10}}{2^{10}} = 1\text{ MiB} \\ 128 \cdot 6 = 768\text{ B} = 0.000732\text{ MiB} \end{cases}$

$1 + 0.000732 = 1.000732$ MiB

$1.000732 \cdot 5 = 5.003660$ MiB.

Max. $\begin{cases} 256 \times 2048 = 2^8 \cdot 2^{11} = 2^{19}. \\ 2^{19} \cdot 4 = 2^{19} \cdot 2^2 = 2^{21} = \frac{2^{21}}{2^{10}} = 2^{11}\text{ KiB} = \frac{2^{11}}{2^{10}} = 2\text{ MiB} \\ 256 \cdot 6 = 1536 = 0.001465\text{ MiB}. \end{cases}$

$2 + 0.001465 = 2.001465$ MiB.

$2.001465 \cdot 5 = 10.007325$ MiB.

So the Mimmum is 5.003660 MiB.
Maxmum is 10.007325 MiB.