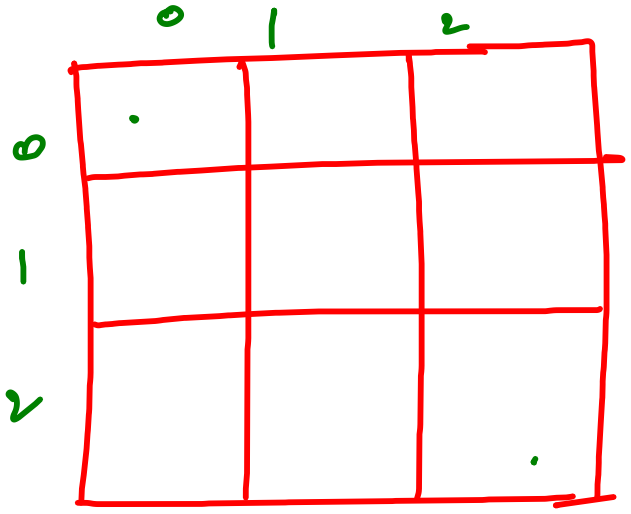


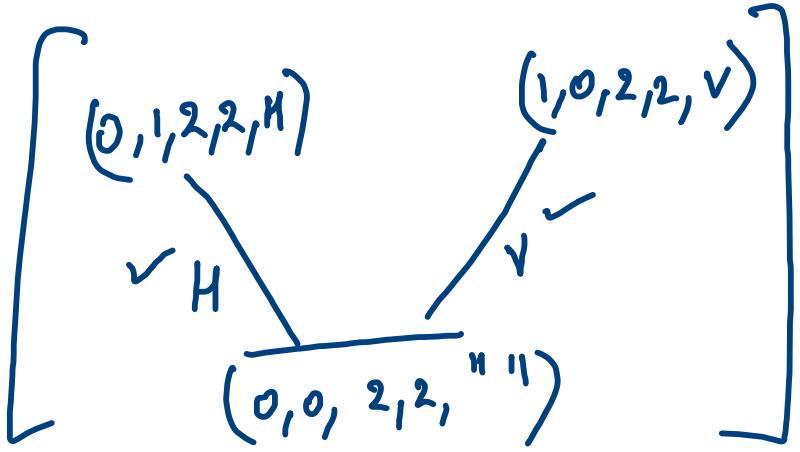
sr = nr
sc = nc
⇒ input

(x,y) H → (x, y+1)
V
(x+1, y)

src ⇒ (0,0)

dest ⇒ (2,2) ⇒ (nr-1), (nc-1)

```
if(sr > dr || sc > dc){
    // invalid
    return;
}
```

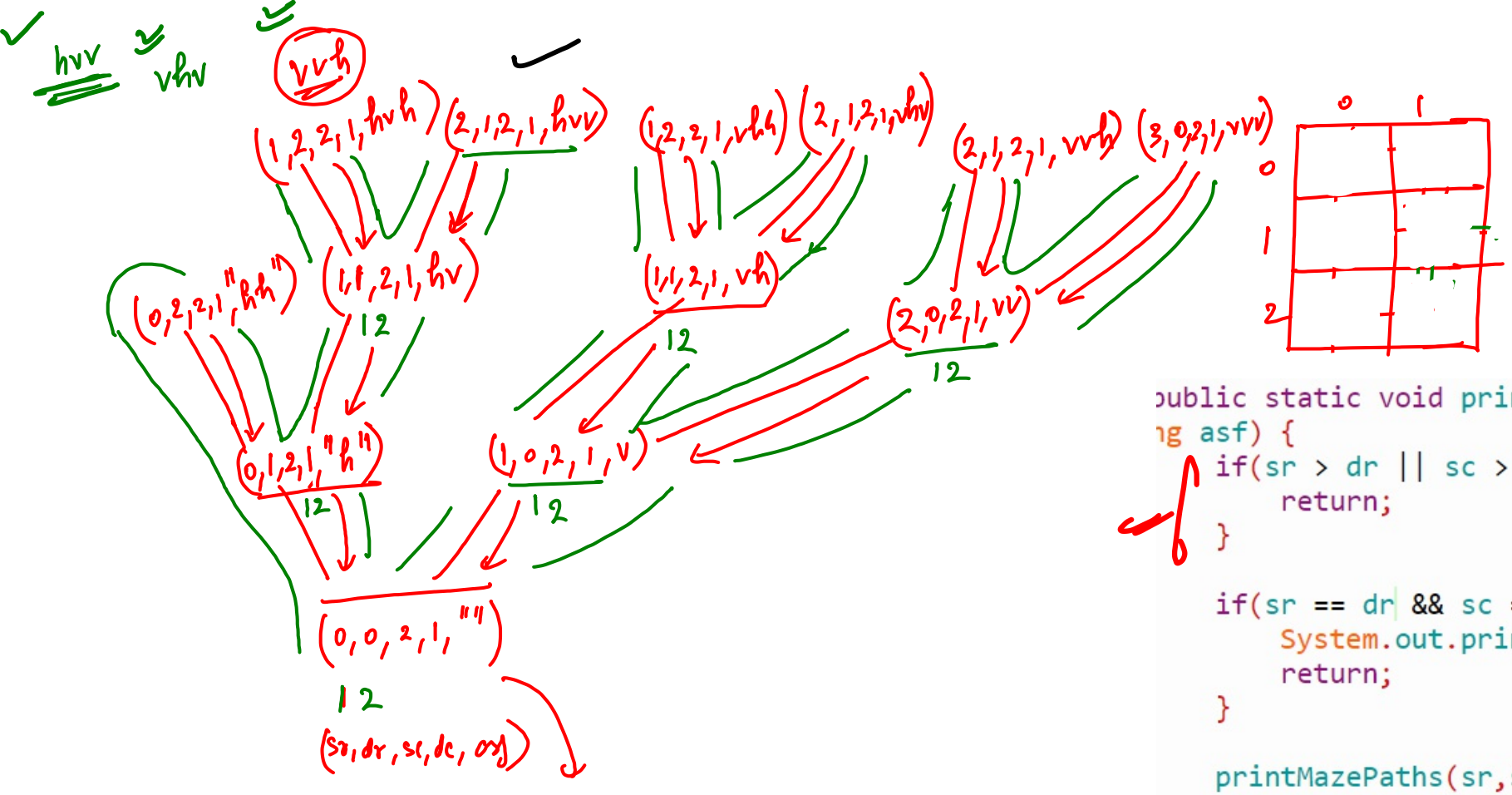



```
if(sr == dr && sc == dc){
    System.out.print(asf);
    return;
}
```

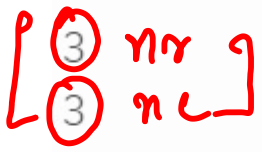
7 min

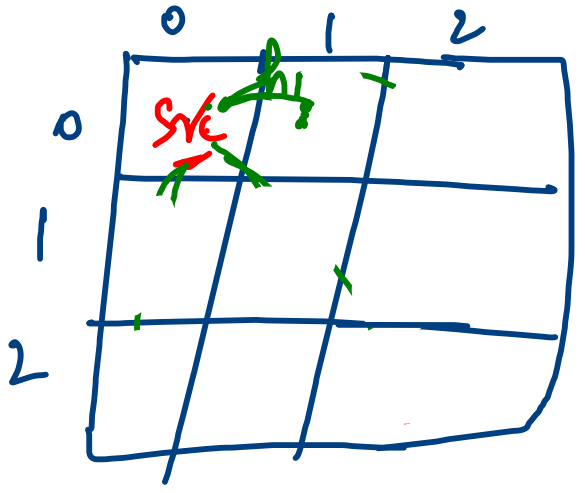
```java
public static void main(String[] args) throws Exception {
    Scanner scn = new Scanner(System.in);
    int nr = scn.nextInt();
    int nc = scn.nextInt();

    printMazePaths(0,0,nr-1,nc-1,"");
}

// sr - source row
// sc - source column
// dr - destination row
// dc - destination column
public static void printMazePaths(int sr, int sc, int dr, int dc, String asf) {
    if(sr > dr || sc > dc){
        return;
    }

    if(sr == dr && sc == dc){
        System.out.println(asf);
        return;
    }

    printMazePaths(sr,sc+1,dr,dc,asf+"h");
    printMazePaths(sr+1,sc,dr,dc,asf+"v");
}
```

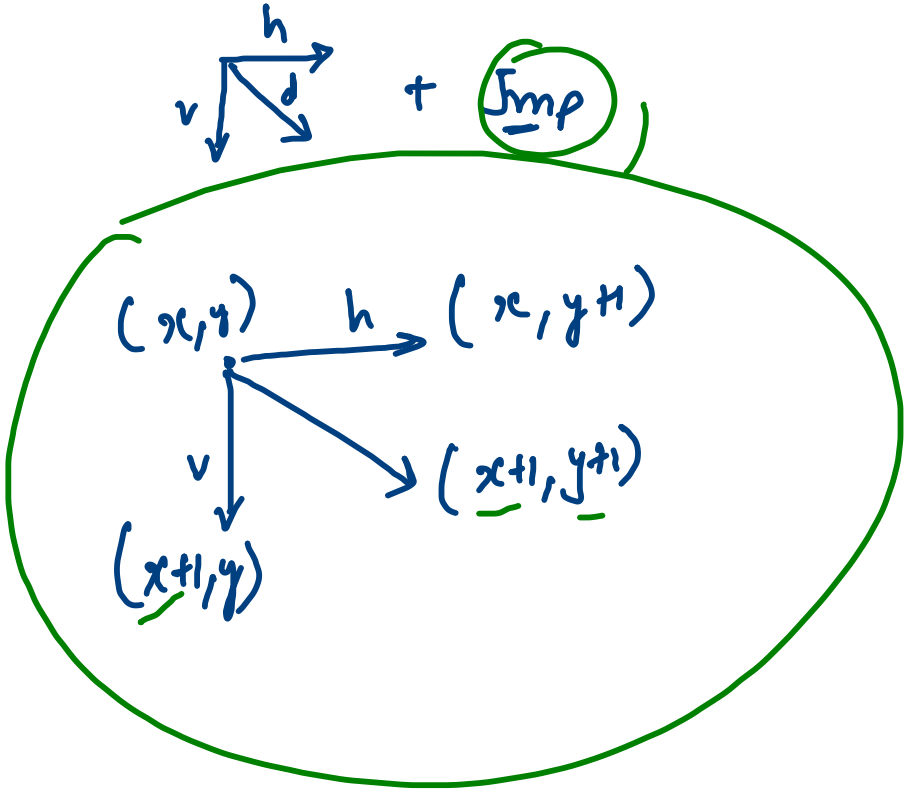
3
2
- 1
- 2

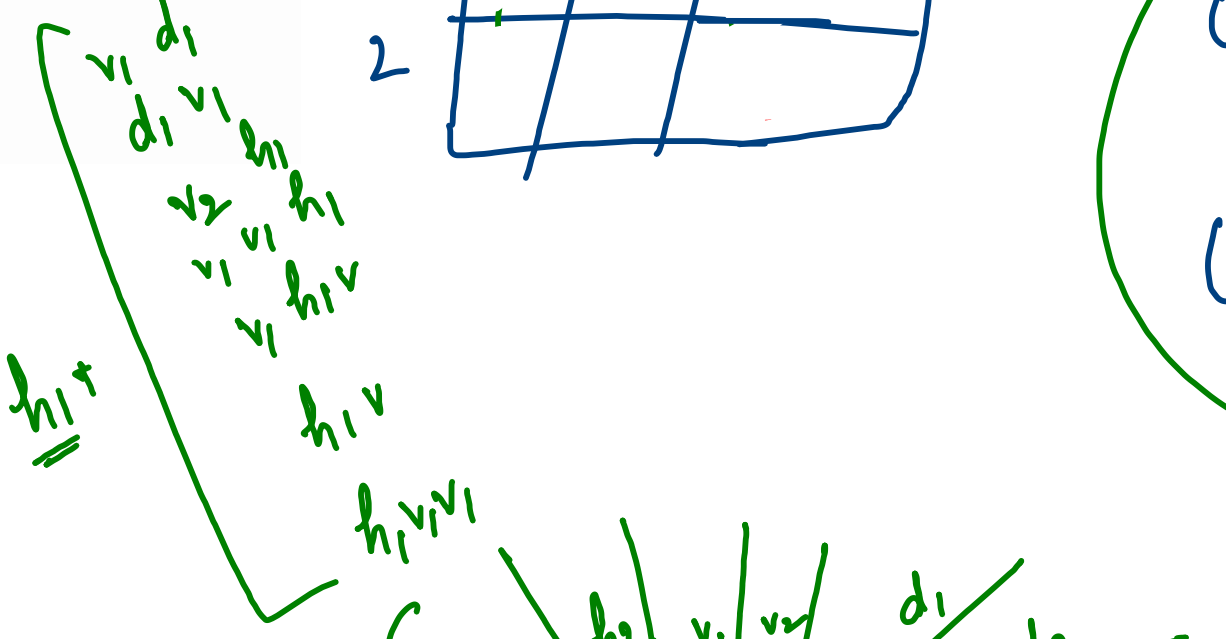


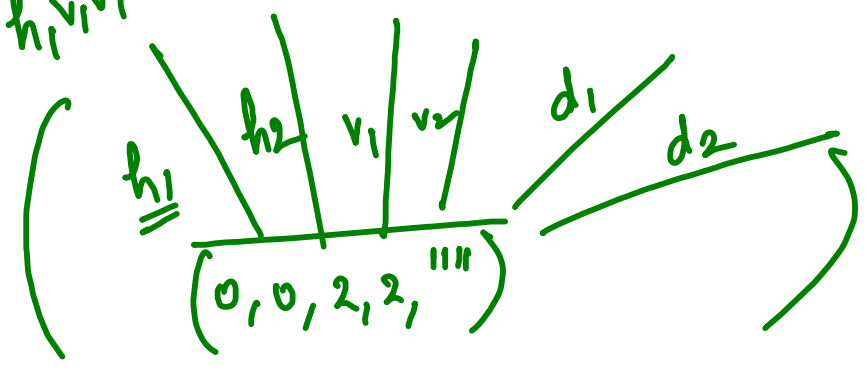

```
public static void printMazePaths(int sr, int s
ng asf) {
    if(sr > dr || sc > dc){
        return;
    }

    if(sr == dr && sc == dc){
        System.out.println(asf);
        return;
    }

    printMazePaths(sr,sc+1,dr,dc,asf+"h");
    printMazePaths(sr+1,sc,dr,dc,asf+"v");
}
```

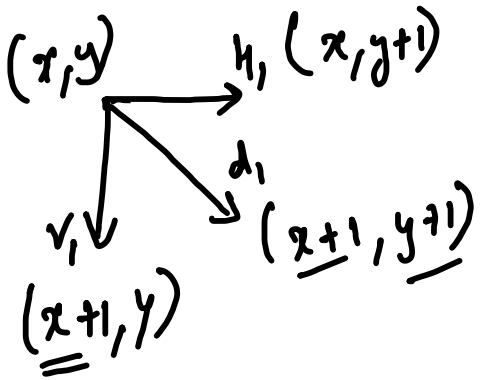

3 nr
3 nc

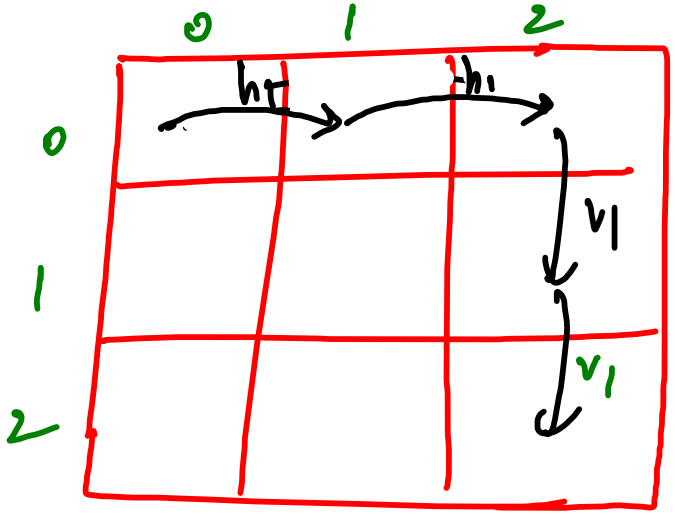

0
1
2
0
1
2
src
h1


h
v
d
+
fmp
(x,y)
h
(x, y+1)
v
(x+1, y+1)
(x+1, y)

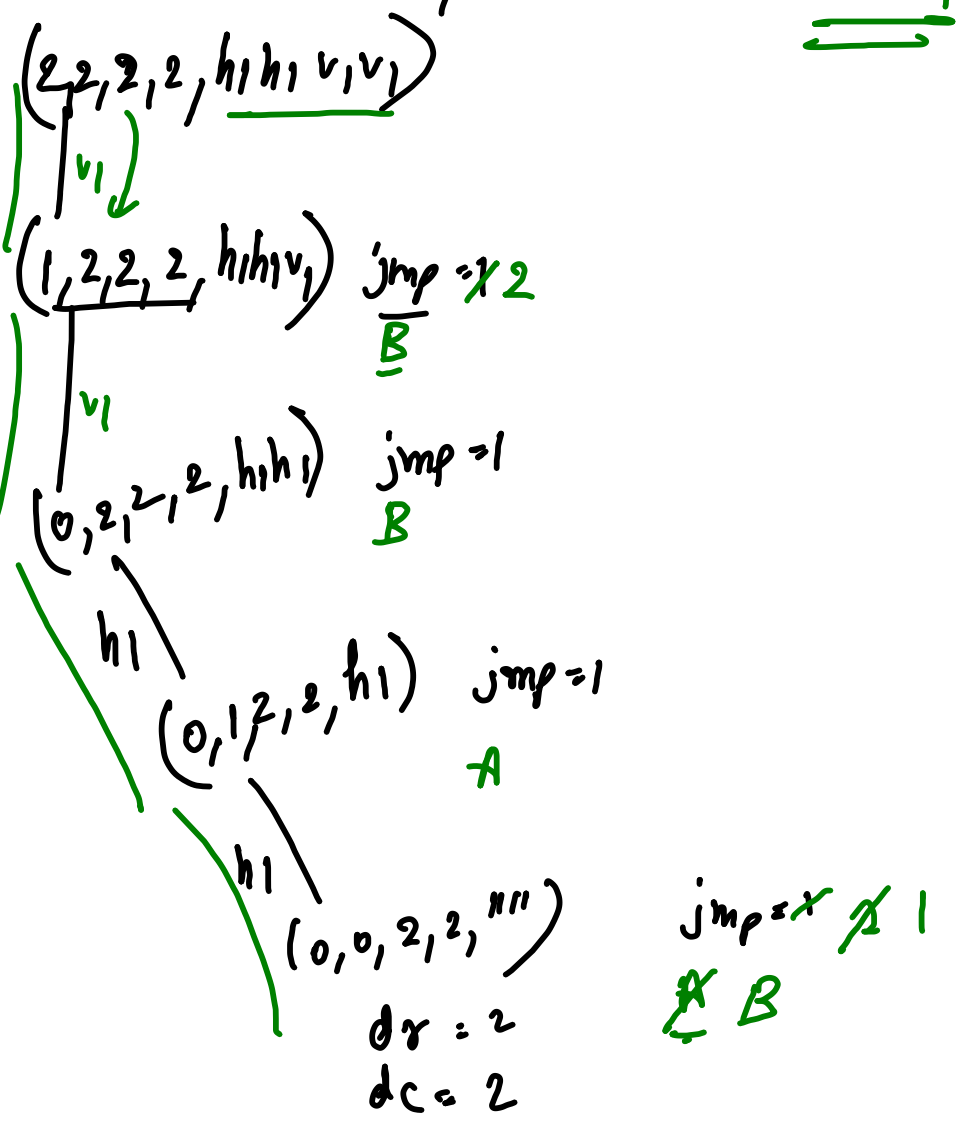

d1
v1
d1 v1
h1
v2
v1
h1
v1
h1 v
v1
h1 v
h1 v1 v1
h1


h1
h2
v1
v2
d1
d2
(0, 0, 2, 2, 1111)



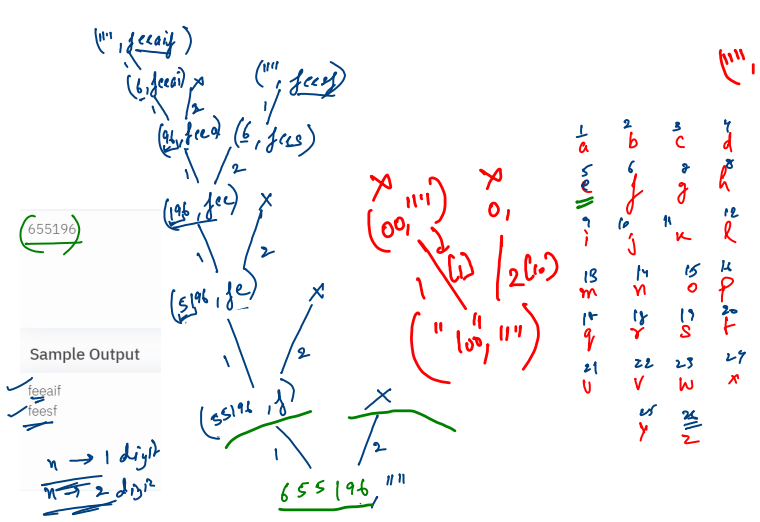


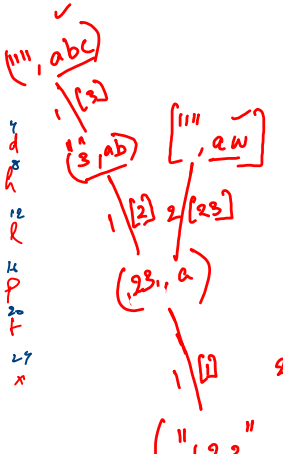
nr=3 nc=3 h1h1v1v1

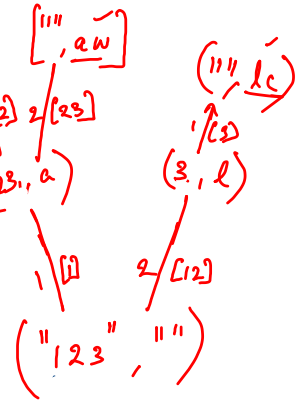


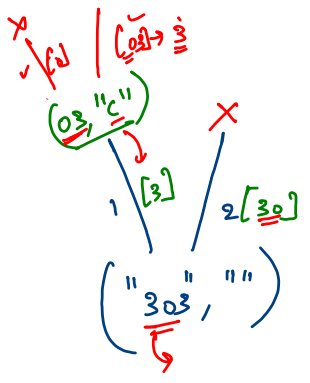
Proactive

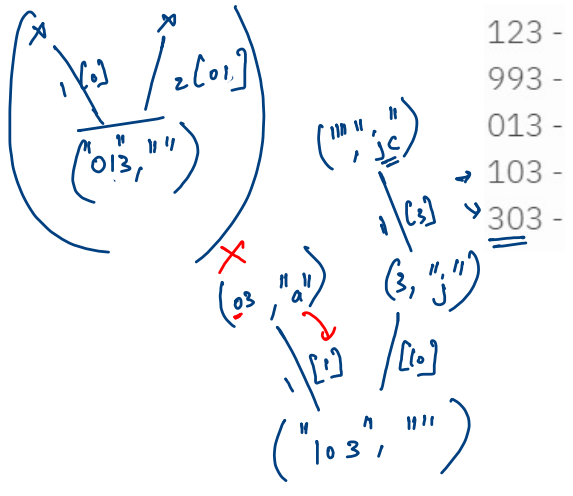
```
public static void printMazePaths(int sr, int sc, int dr, int dc, String psf) {
    if(sr == dr && sc == dc){
        System.out.println(psf);
        return;
    }

A [ for(int jmp = 1 ; sc + jmp <= dc ; jmp++){
        printMazePaths(sr,sc+jmp,dr,dc,psf+"h"+jmp);
    } ] h

B [ for(int jmp = 1 ; sr + jmp <= dr ; jmp++){
        printMazePaths(sr+jmp,sc,dr,dc,psf+"v"+jmp);
    } ] v

C [ for(int jmp = 1 ; sr + jmp <= dr && sc + jmp <= dc; jmp++){
        printMazePaths(sr+jmp,sc+jmp,dr,dc,psf+"d"+jmp);
    } ] d
}
```


655196
Sample Output
feeaif
feesf



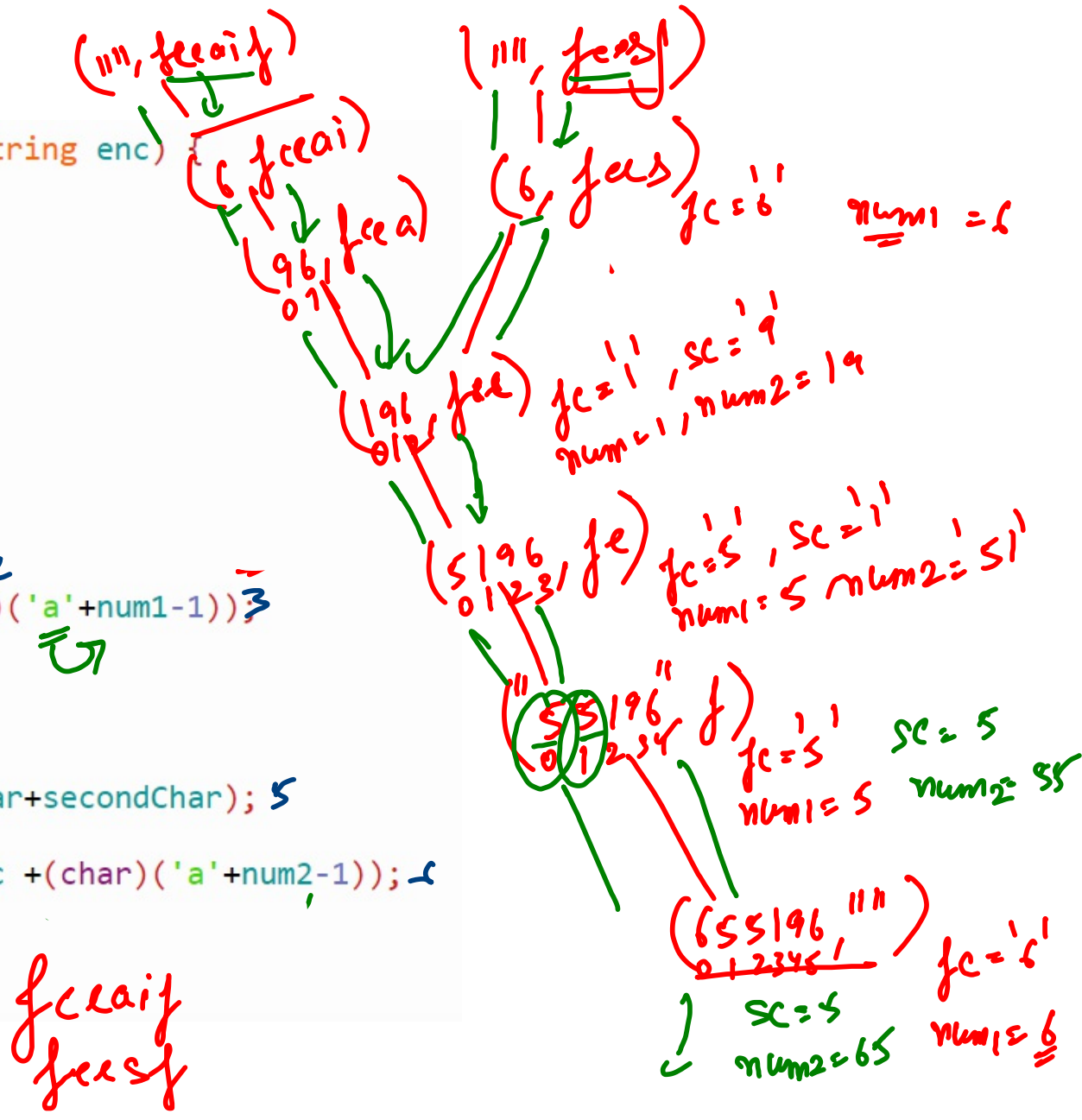




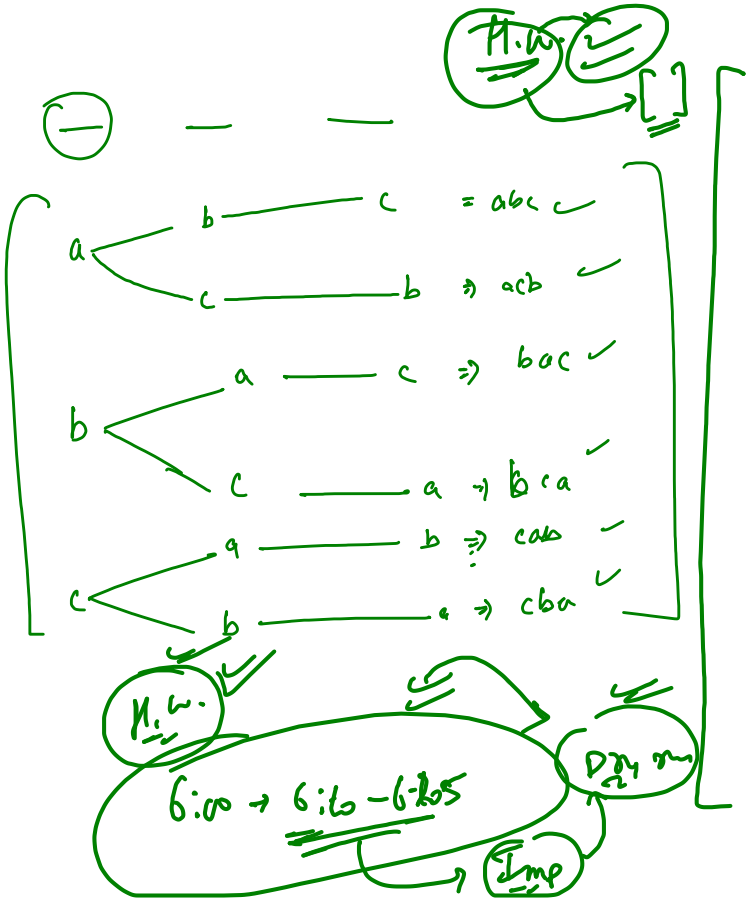

123 -
993 -
013 -
103 -
303 -

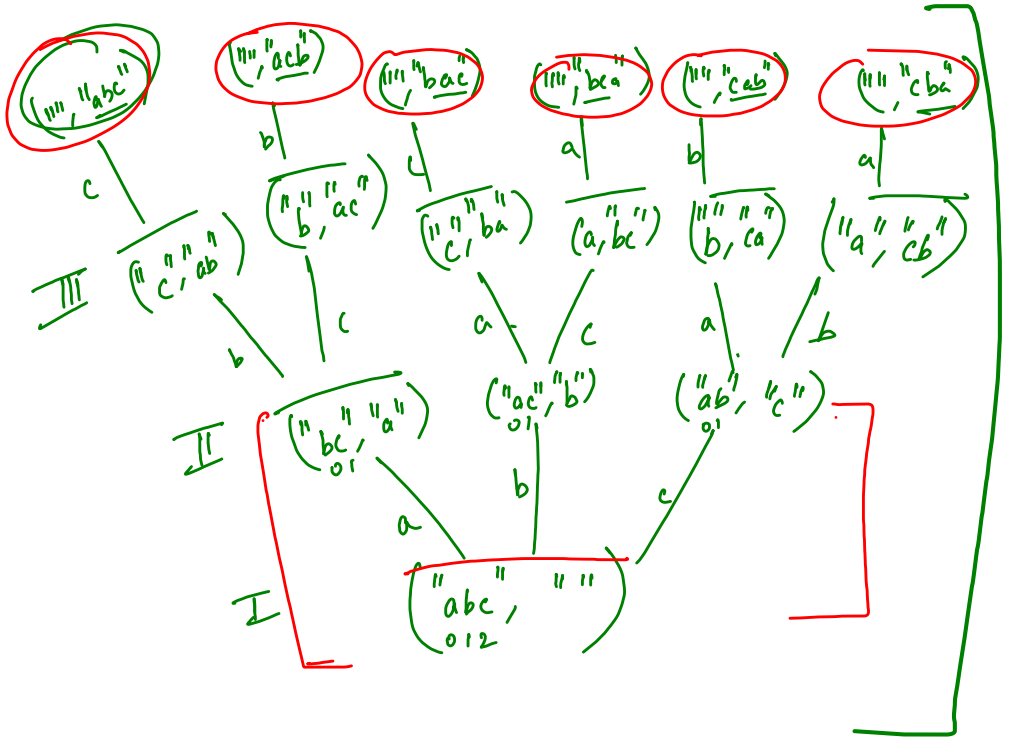
```java
public static void printEncodings(String str,String enc) {
    if(str.length() == 0){
        System.out.println(enc);
        return;
    }
    if(str.charAt(0) == '0'){
        return;
    }

    // single digit num
    char firstChar = str.charAt(0);
    int num1 = Integer.parseInt(firstChar+"");
    printEncodings(str.substring(1),enc +(char)('a'+num1-1));

    if(str.length() > 1){
        // double digit num
        char secondChar = str.charAt(1);
        int num2 = Integer.parseInt(""+firstChar+secondChar);
        if(num2 >= 10 && num2 <= 26){
            printEncodings(str.substring(2),enc +(char)('a'+num2-1));
        }
    }
}
```

655196



abc





abc acb bac bca
cab cba

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com

num1 = 1 → (a)

char → 'a' + num1 - 1

⇒ 97 + 1 - 1 ⇒ (97) → a

num1 ⇒ 5

char ⇒ 'a' + 5 - 1

⇒ 97 + 5 - 1

⇒ 101 → e

5 →

97 + 5 - 1 ⇒ 101 (e)