aaabbccdee

d
1 ←

aaa = a

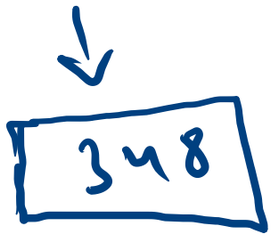
compression 1

a b c d e

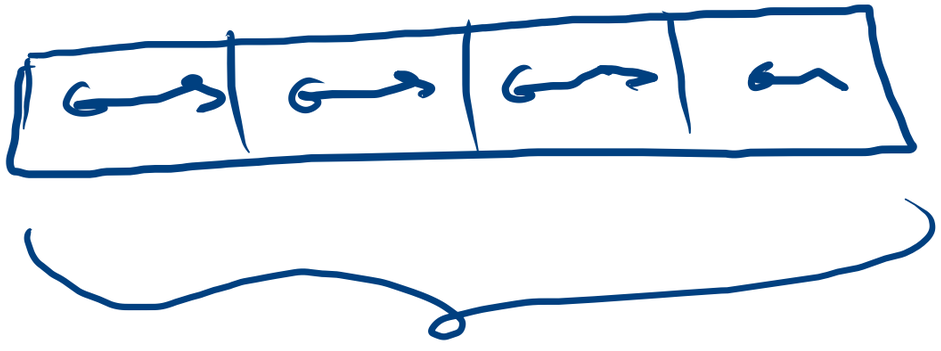
compression 2

a 3 b 2 c 2 d e 2

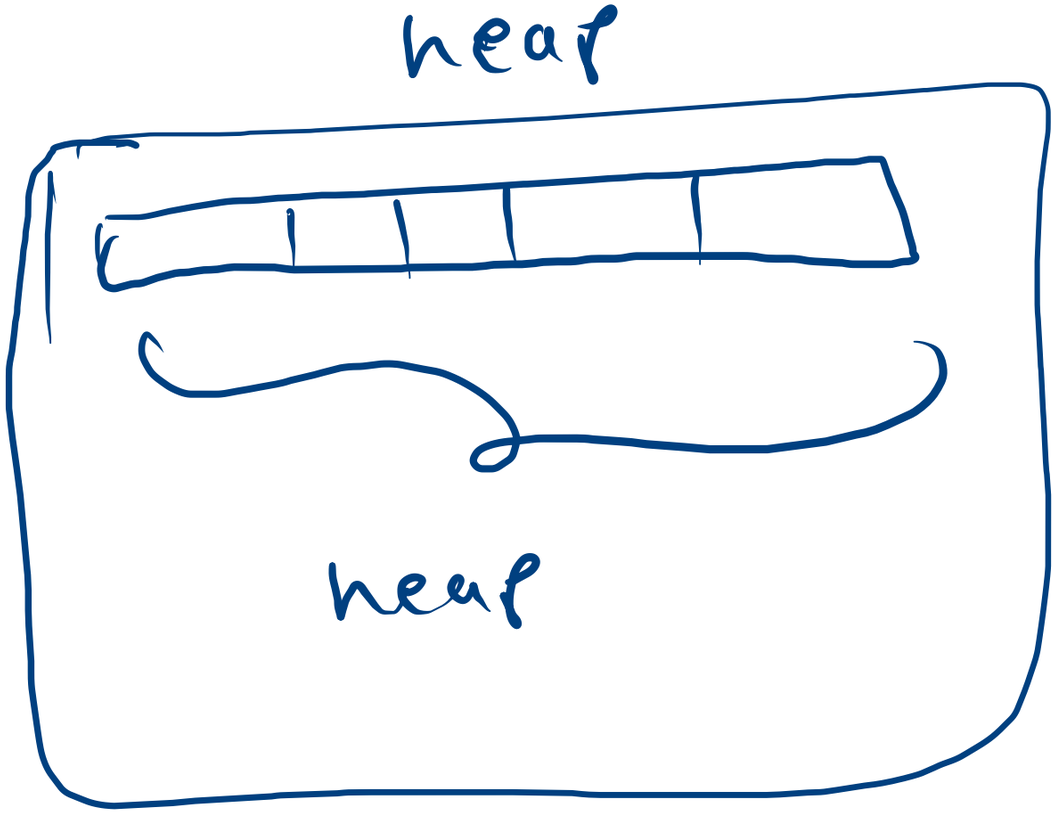
int



array



String



aaabbccdeeF

count 01

char F

ans a3b2c2deF

while(i < str.length())

→ int count = 0

→ char ch -> str.charAt(i);
i<str.length() &&
while(str.charAt(i)==ch)

→ i++

→ count++

}

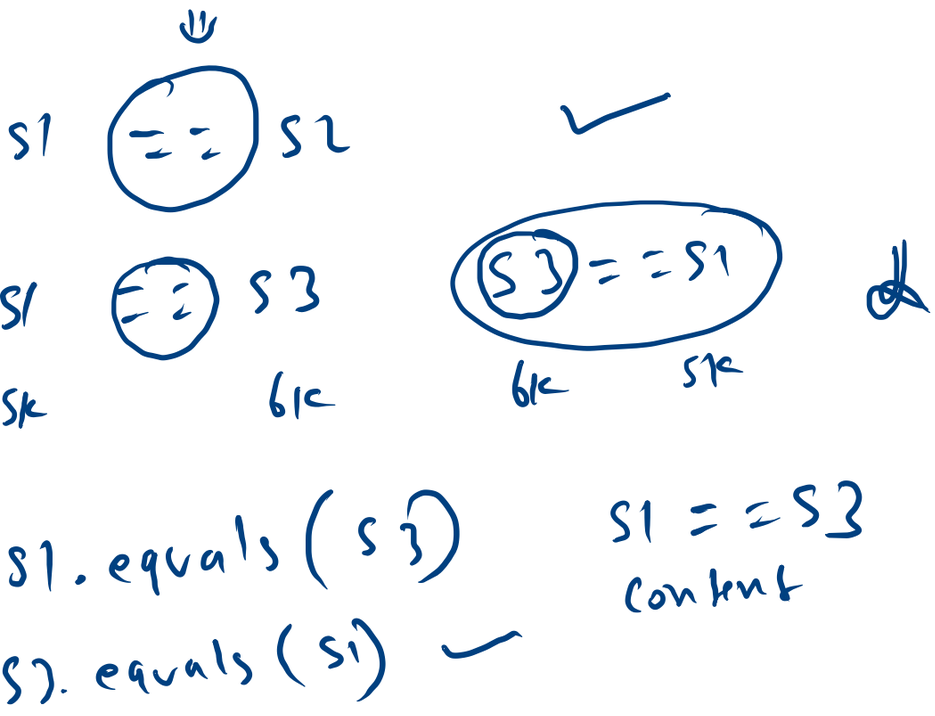
-> ans += ch;

if(count>1) ans += count;

String s1 = "abc";

String s2 = "abc";

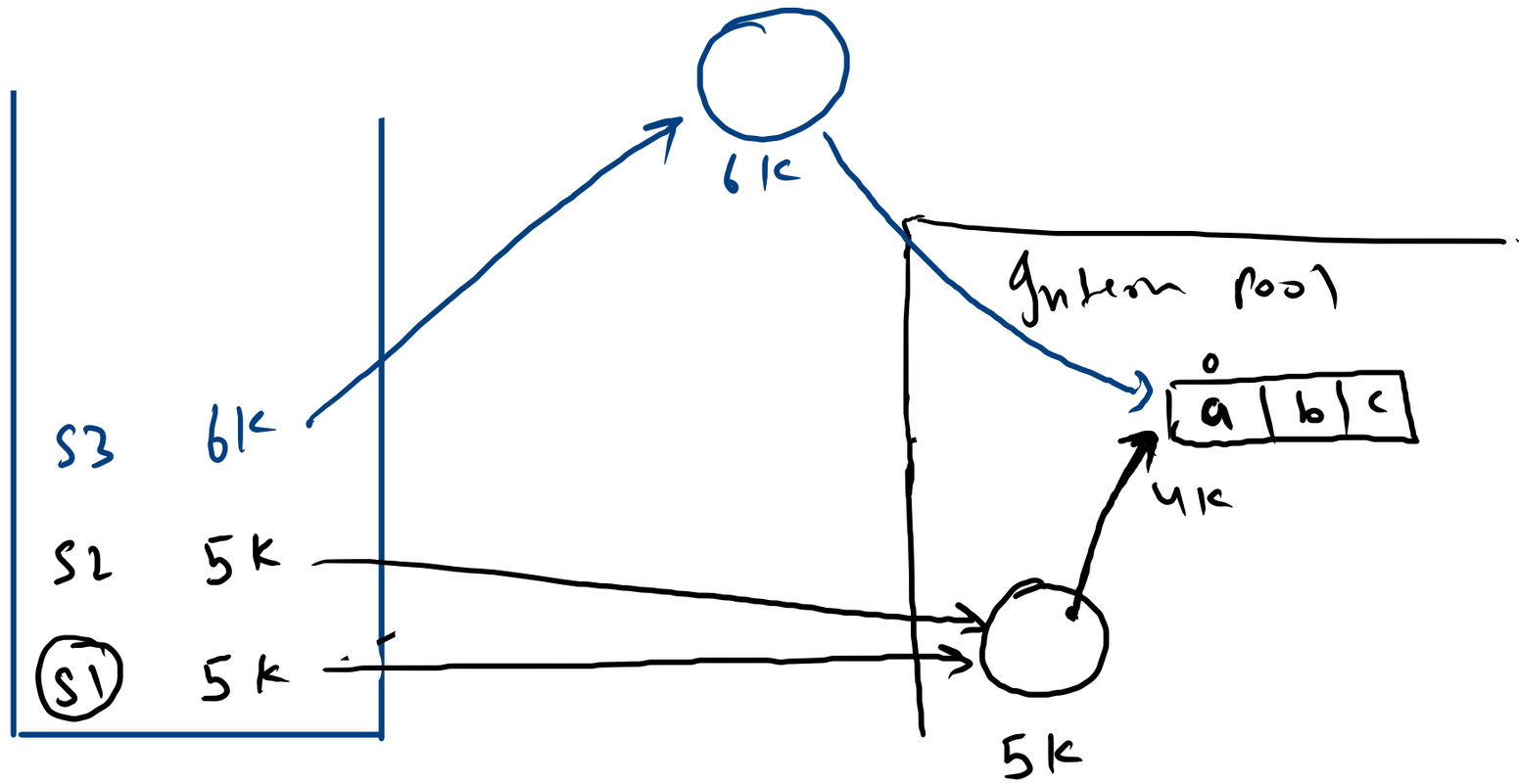
String s3 = new String("abc")



s1.equals(s3)

s3.equals(s1)

s1 == s3
content



```
String s1 = "abc";
String s2 = "abc";
String s3 = new String("abc")
```

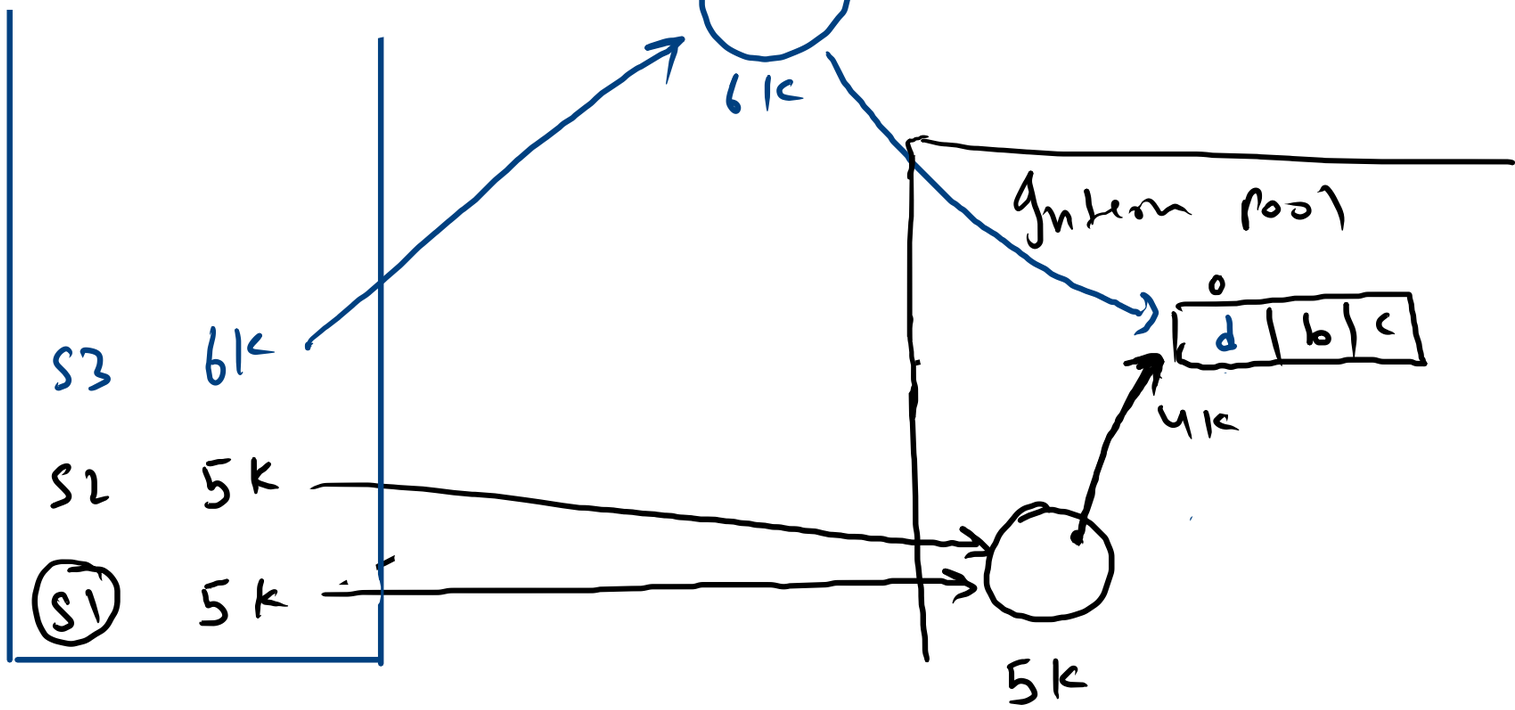
sharing → Interning

memory

print(s1) → abc
print(s2) → abc

s1.charAt(0) = 'd'

print(s1) → dbc
print(s2) → dbc



String s1 = ""

n

n

for (int i = 0 ; i < n ; i++) {

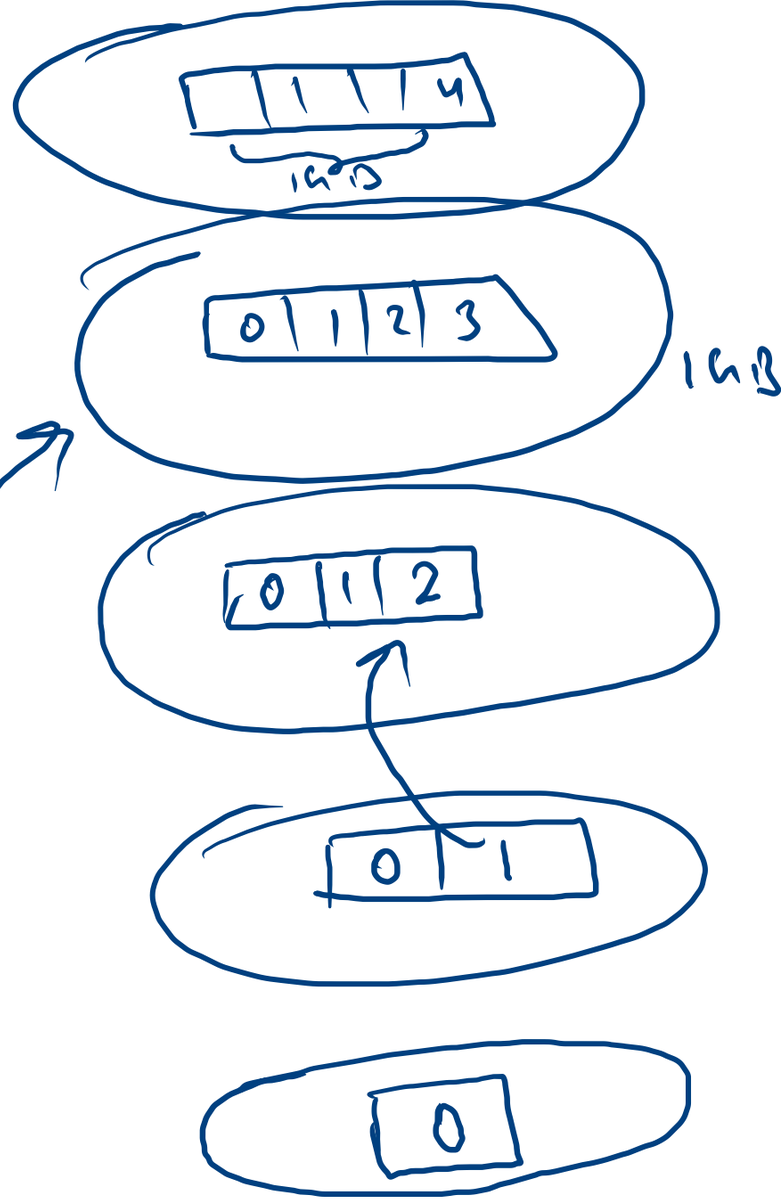
s1 += i;

}

ne ∈ performance

i = 0
i = 1
i = 2
i = 3
i = n

s1



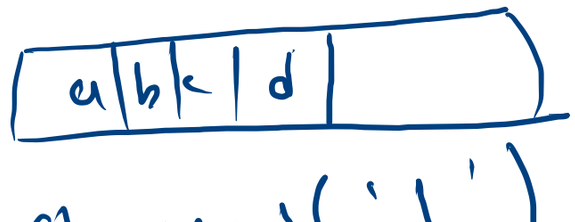
StringBuilder sb = new StringBuilder();

String s = new String("abc");

read charAt(i)

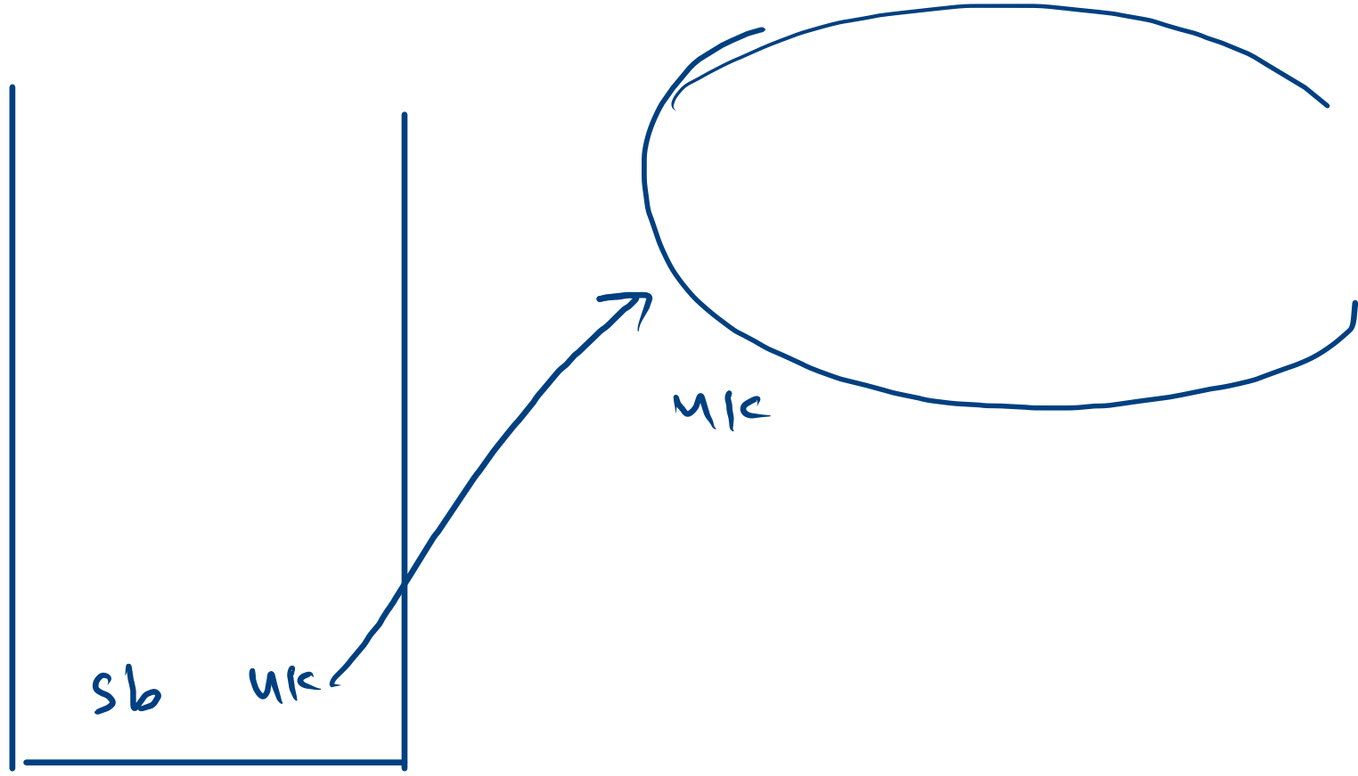
upsak setcharAt(i,'b')

sb

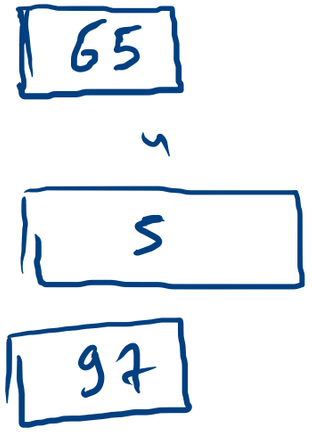


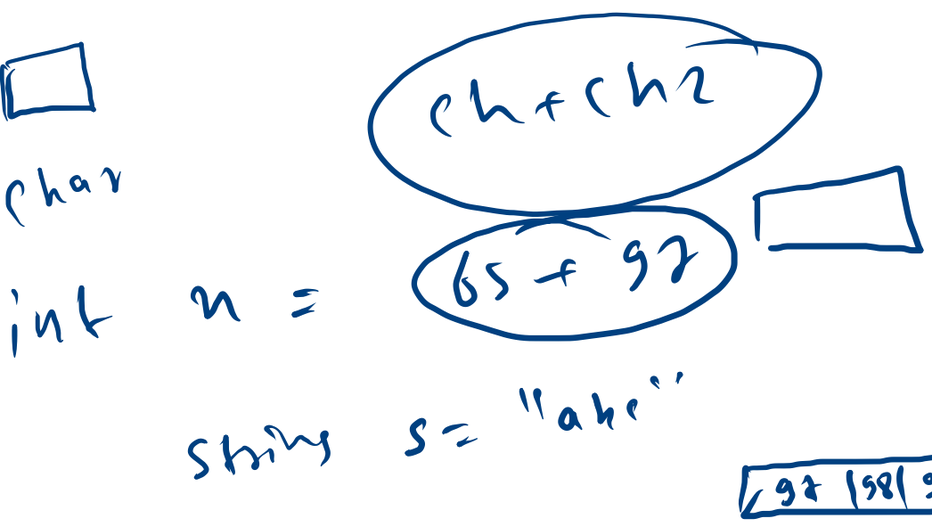
sb.append('d')

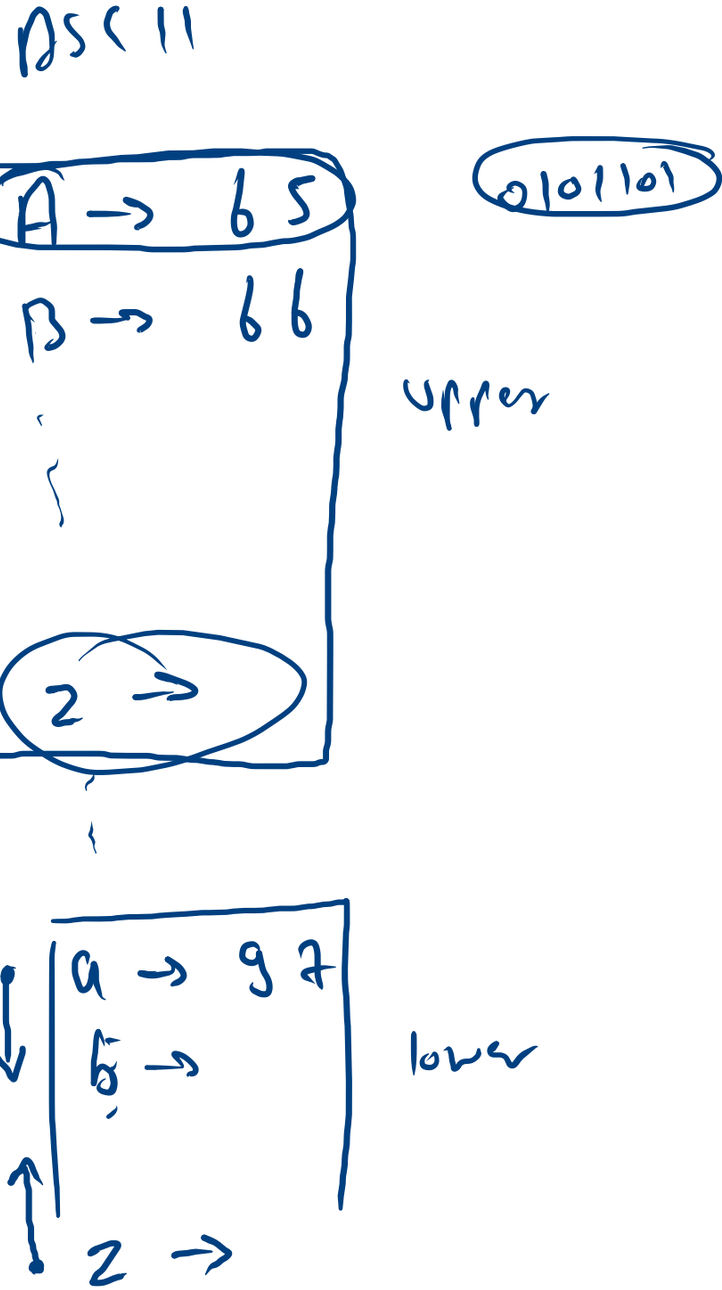
```
()->
append ( s )
length ()
charAt ( i )
sb.charAt( i , 'd' );
insert ( i , 'z' );
deleteCharAt( i );
```



```
char ch = 'A';
int a = 5;
char ch2 = 'a'
```







str -> pepCODinG

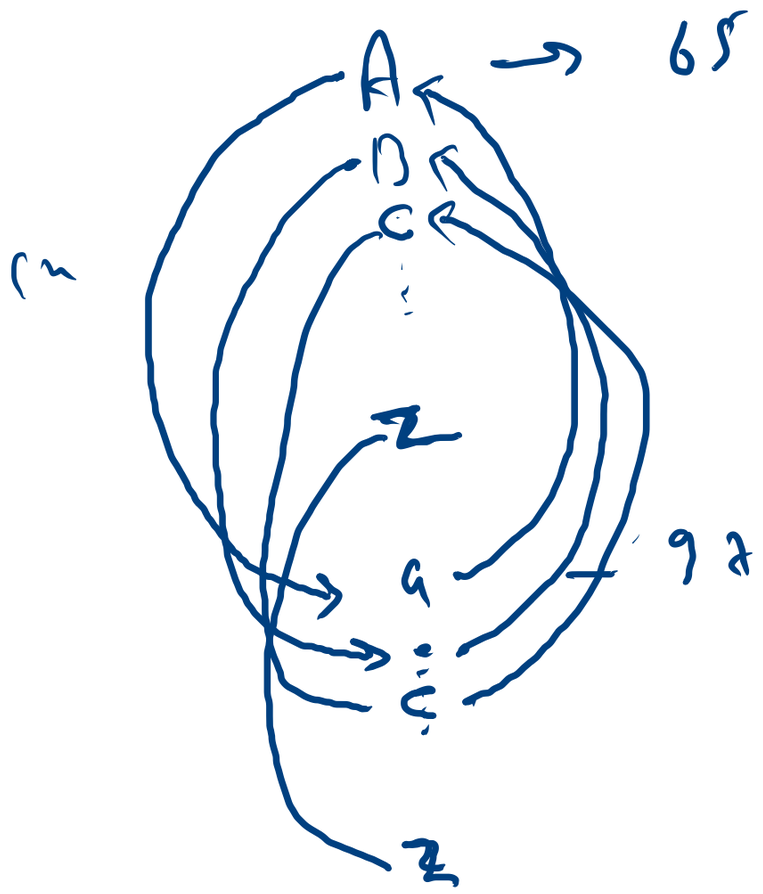
PEPcodING

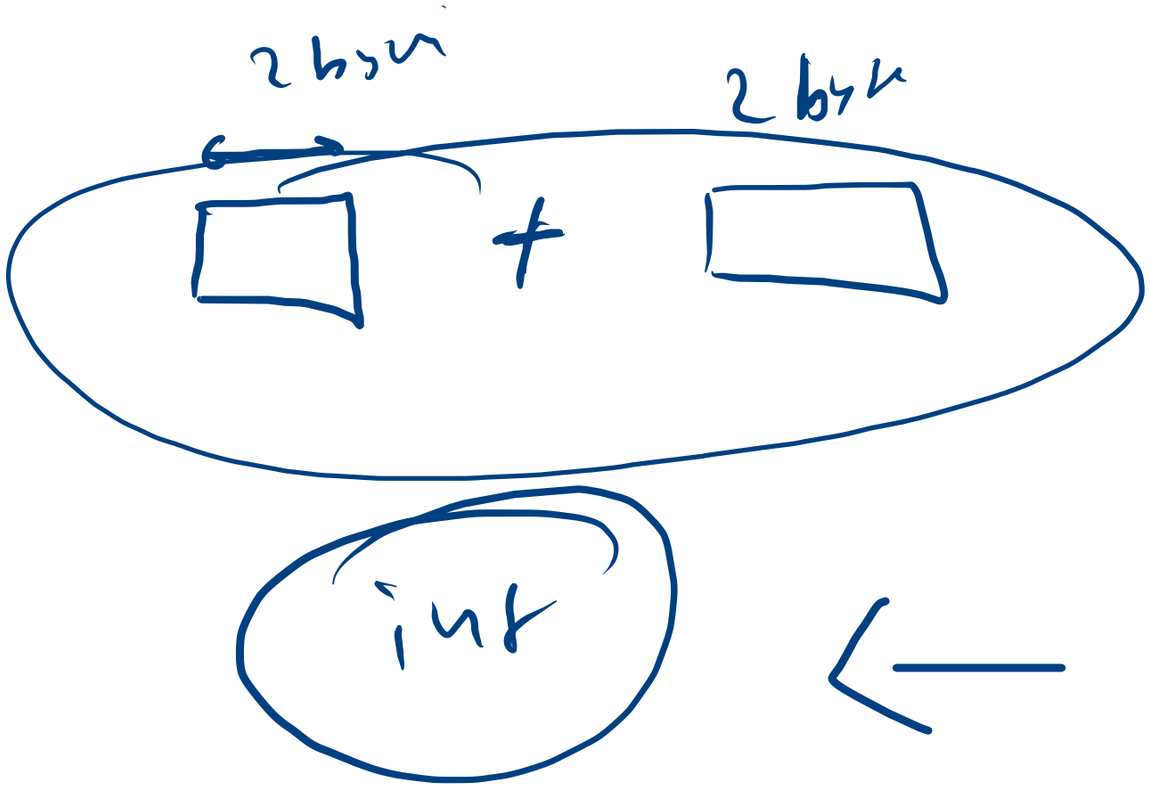
p => str.charAt(i)

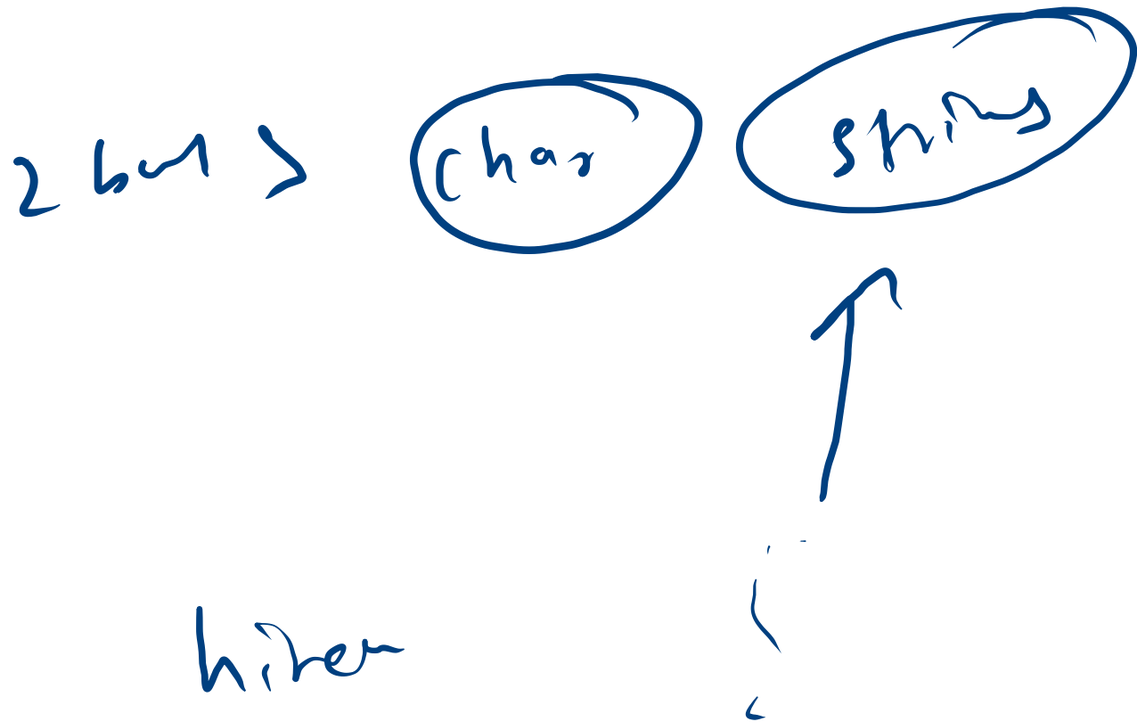
p

int diff = 'a' - 'A';

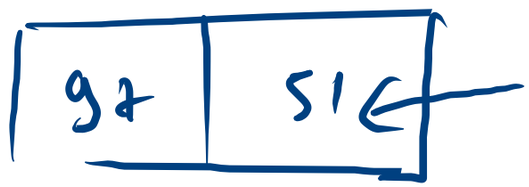
StringBuilder ans = new S.. B...();



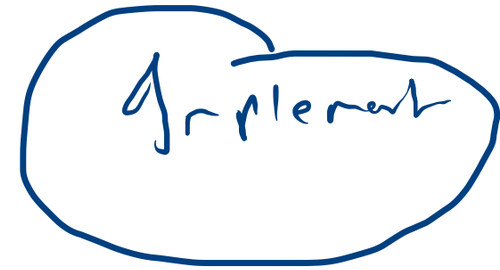


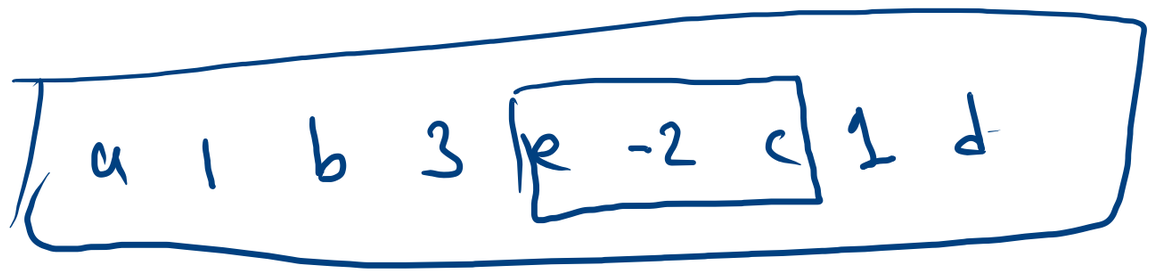


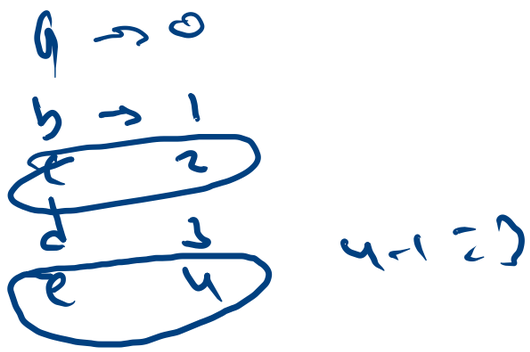
str = " a 2 "

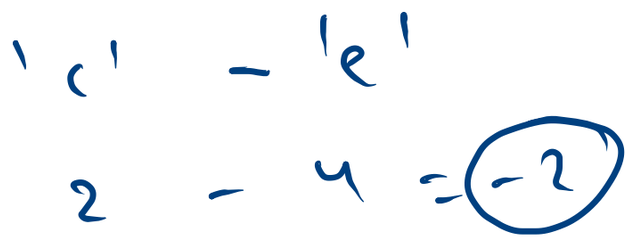


For "abecd", the answer should be "a1b3e-2c1d", as
'b'-'a' = 1
'e'-'b' = 3
'c'-'e' = -2
'd'-'c' = 1





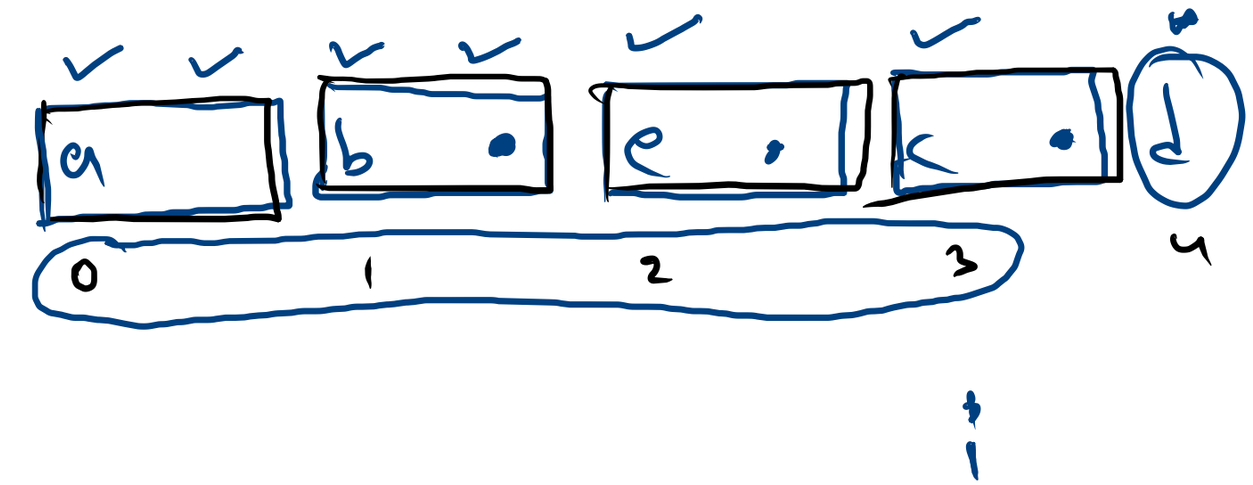


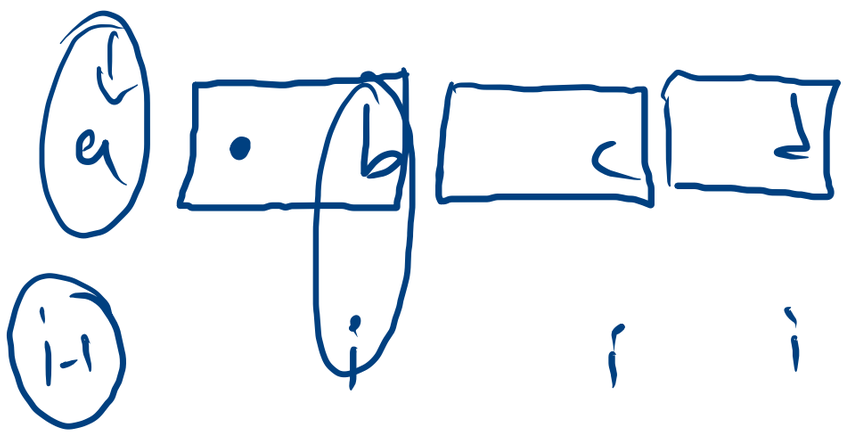


diff = charAt(i+1) − charAt(i)

d − c

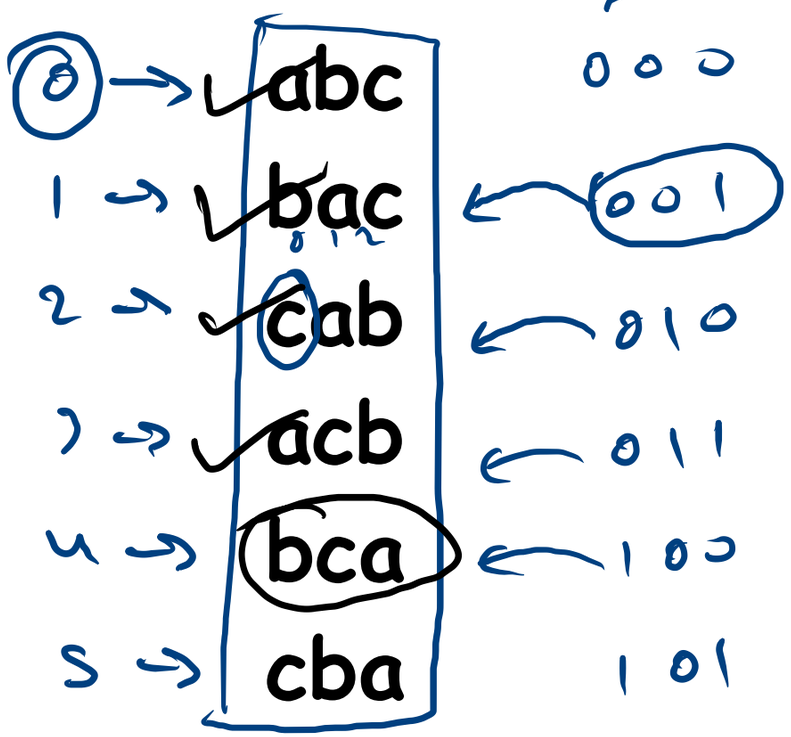
For "abecd", the answer should be "a1b3e-2c1d", as
'b'-'a' = 1
'e'-'b' = 3
'c'-'e' = -2
'd'-'c' = 1





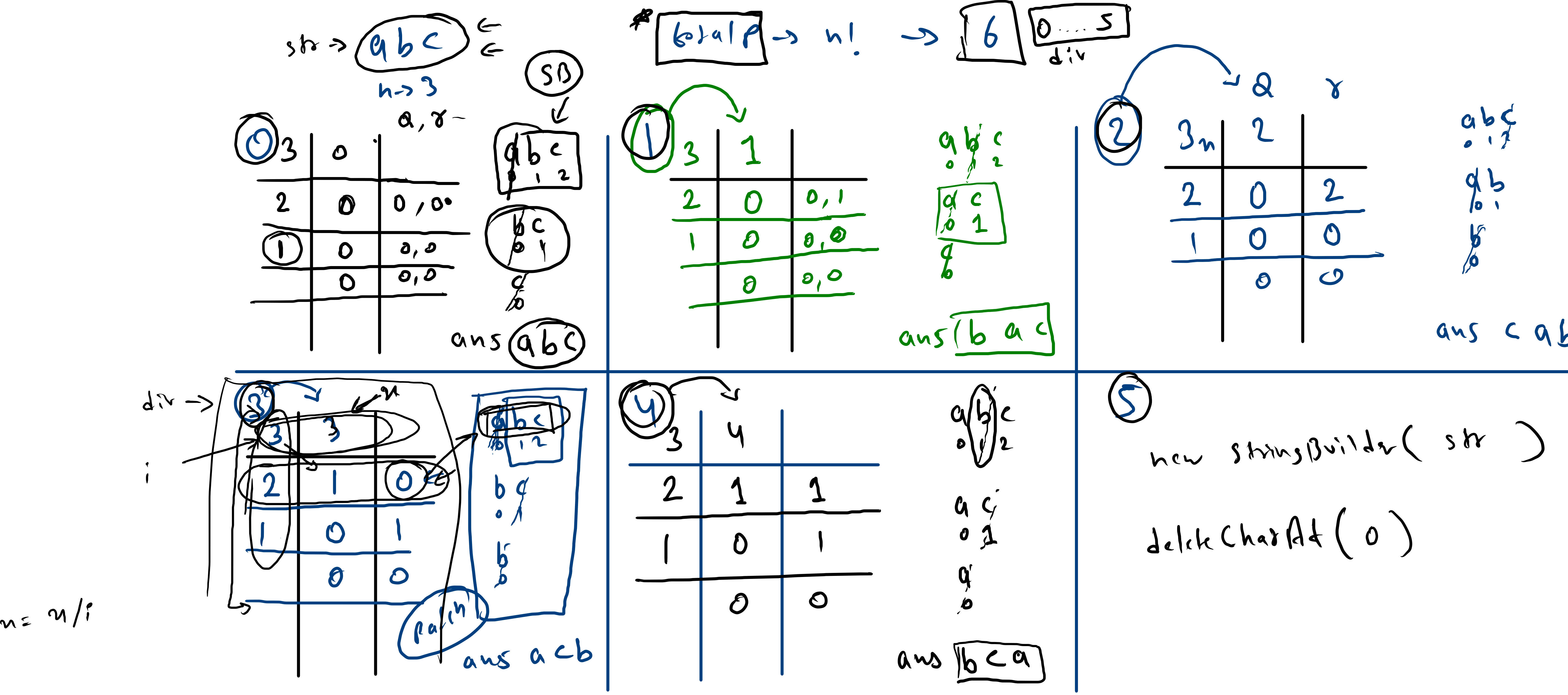
Sb -> a1b3e-2c1d

abc → 3! → 6



0 5

abc
acb
bac
bca
cab
cba


str -> abc
n->3
q, r
SB
totalP -> n! -> 6
0 5
div
ans abc
ans bac
ans cab
div ->
i
n = n/i
path
ans acb
ans bca
new StringBuilder(str)
deleteCharAt (0)

Array

Searching →

size fixed → ArrayList

insert →

delete →

same type

prog

index — access time

continuous memory

int ar[] = new int[size];

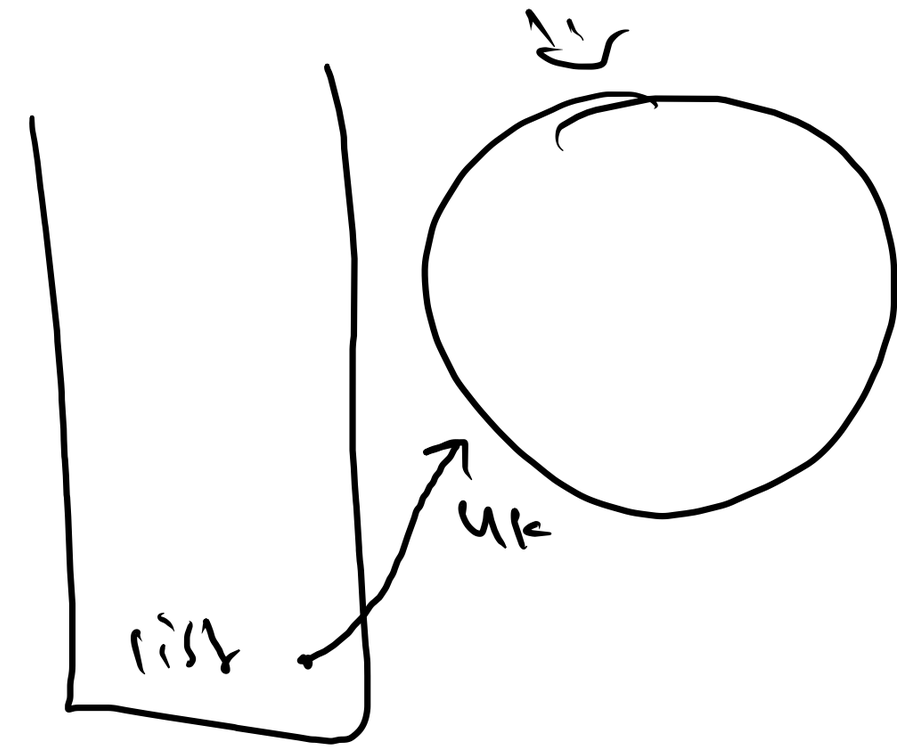
ArrayList<Integer> list = new ArrayList<>();

size

add(val) // last

add(i, val) // insert at i

remove(i) //



10 3 10 3

10 3 10 3 remove prints

10 10 3

insert → rearrange

remove → "

al.remove(i)

i=0

3 11 13 15 16

11 13 15 16

13 15 16

15 16

15 16

al → [3 12 13 15]

[12, 15]

0 1 2 3

3 12 13 15

↓

3 12 15

[10^14]