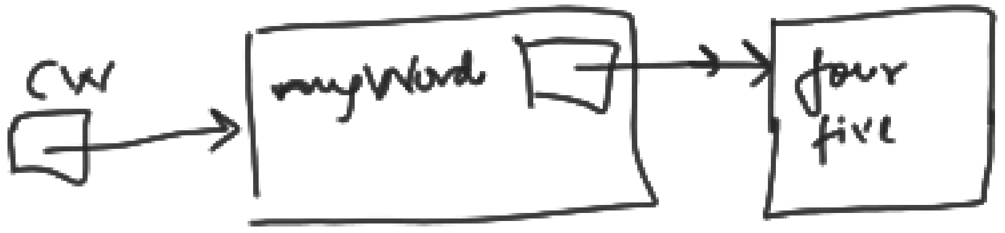
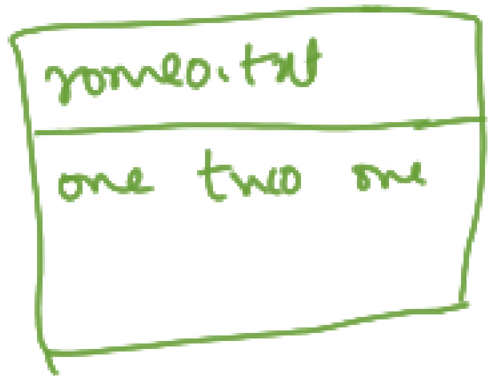
```
CountWords cw = new CountWords();
cw.readWords(filename);
int count = cw.getCount(); → 3

cw.readWords("test.txt");
count = cw.getCount(); → 2
```

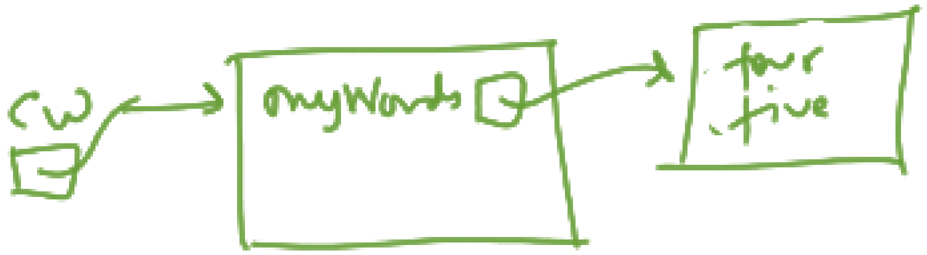


```
cw.readWords("https://nent. . . ");

cw.readWords("/Users/rohan/test.txt");
```

```
class CountWords {
    StorageResource myWords;

    public CountWords() {
        myWords = new SR();
    }

    public void readWords(String source) {
        myWords.clear();
        FR fr = new FR(source);
        for(String word : fr.words()){
            myWords.add(word.toLowerCase());
        }
    }

    public int getCount() {
        return myWords.size();
    }
}
```

```
CW cw = new CW();
println(cw.getCount()); //0
cw.readWord("romeo.txt");
print(cw.getCount()); // 3
print(cw.getCount()); // 3

cw.readWords("test.txt");
print(cw.getCount()); //2
```





```
for(String w : fr.words()){
    w = w.toLowerCase();
    if(!myWords.contains(w)){
        myWords.add(w);
    }
}
```

	0	1	2	3
mywords →	dog	cat	pig	goat
myFreqs →	3	1	4	1

dog cat pig dog goat pig dog pig pig

hv	0
ind	0
i	1
cv	1

```
HashMap<String,Integer> map
            = new HashMap<String,Integer>();

map.size();  // 0

map.put("rohan", 3);
map.put("rohan", map.get("rohan") + 2);

map.get("rohan");  // 5
map.size();  // 1

map.put("kumar", 7 + map.get("rohan"));
map.size();  // 2
map.get("kumar");  // 12
map.put(2, 4);  // Compile error
      (key, value)
```

rohan → 5
kumar → 12

↓A B C ↓D E F ↓G H I ↓J

```
for(i=start; i+3 ≤ dna.length(); i=i+3){
      codon = dna.substring(i, i+3);
```

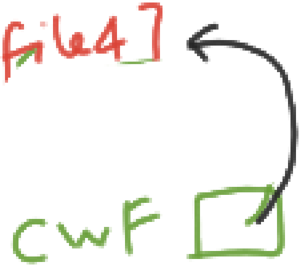
```
map.put("rohan", "dev");
map.put("rohan", "coder");
print(map.get("rohan"));
```

hello : 3

$$\text{map.put(}\underbrace{\text{"hello"}}\text{, }\underbrace{\text{map.get("hello")+1}}_{4}\text{));}$$

```
{
    "key1" : "value",
    "key2" : "value"
}
```

```
map = {
      cat : [file1, file2]
      dog : [file2, file3, file4]
}
```

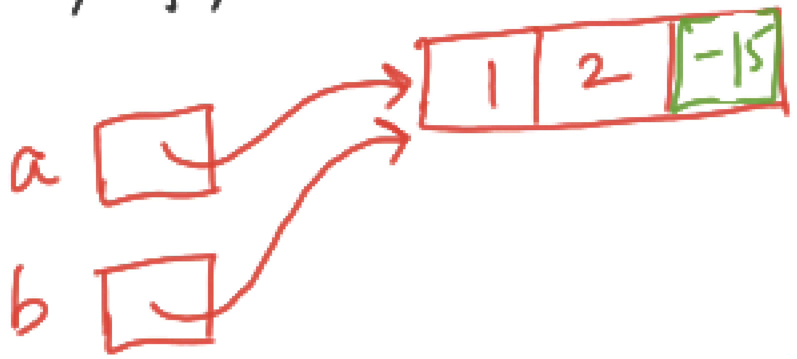


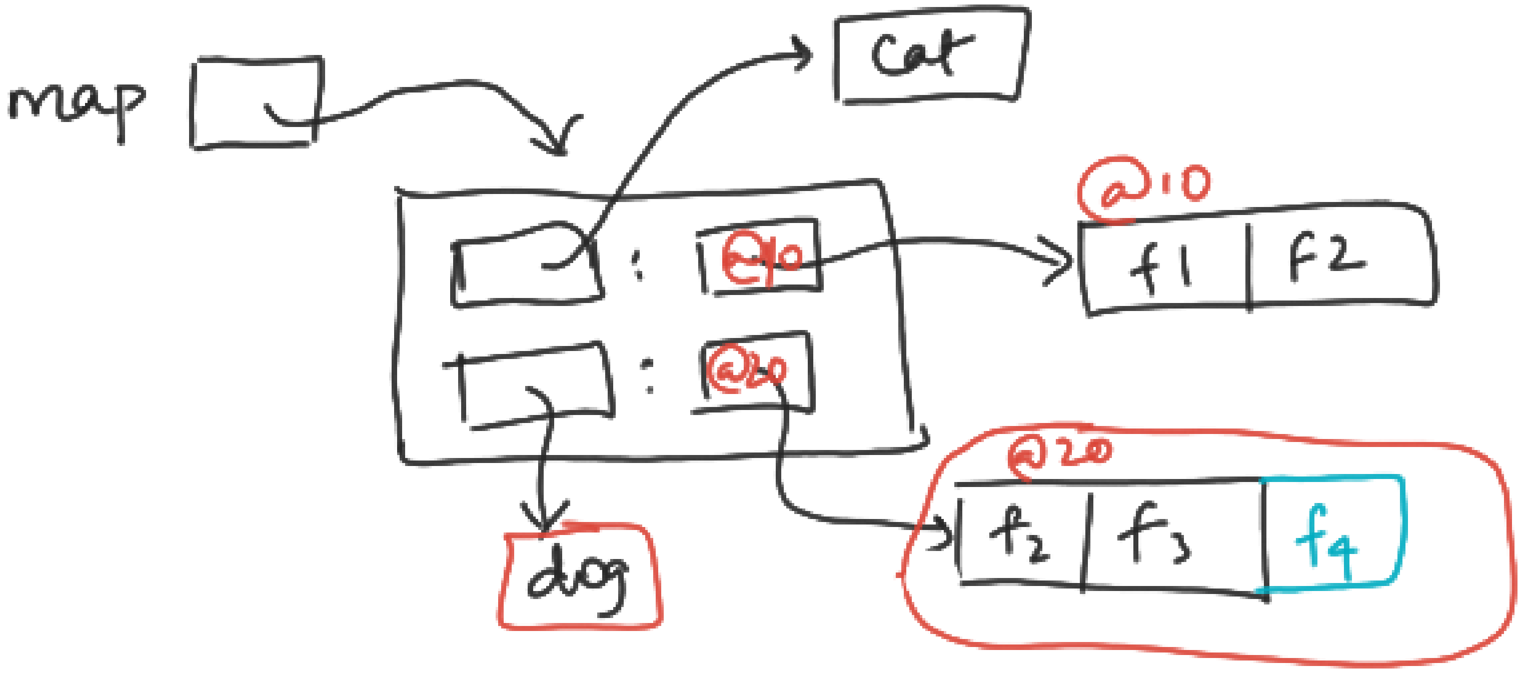
dog it file4

```
map.containkey("dog") → true
ArrayList<String> currWordFiles = map.get(word)
currWordFiles.add("file4");
```

```
horse
map.containKey (horse) → false
AL<String> currWF = new AL<>();
currWF.add("file4")
map.put (horse, currWF)
```

```
int[] a = new int[]{1, 2, 5};
int[] b = a;
b[2] = -15;
print(b[2])  //-15
print(a[2])  //-15
```





```
cwf.add("f4")
```

$n = \text{"and"}$

$h = \text{"}\overset{0}{r}\overset{1}{o}\overset{2}{h}\overset{3}{e}\overset{4}{n}\text{"}$ $i=0$ $i<\boxed{3}$

↑ ↑ ↓

$h.ss(0, 2) \rightarrow ro$

$h.ss(i, i+n.length())$

$\boxed{i=0}$

$j=0..3$

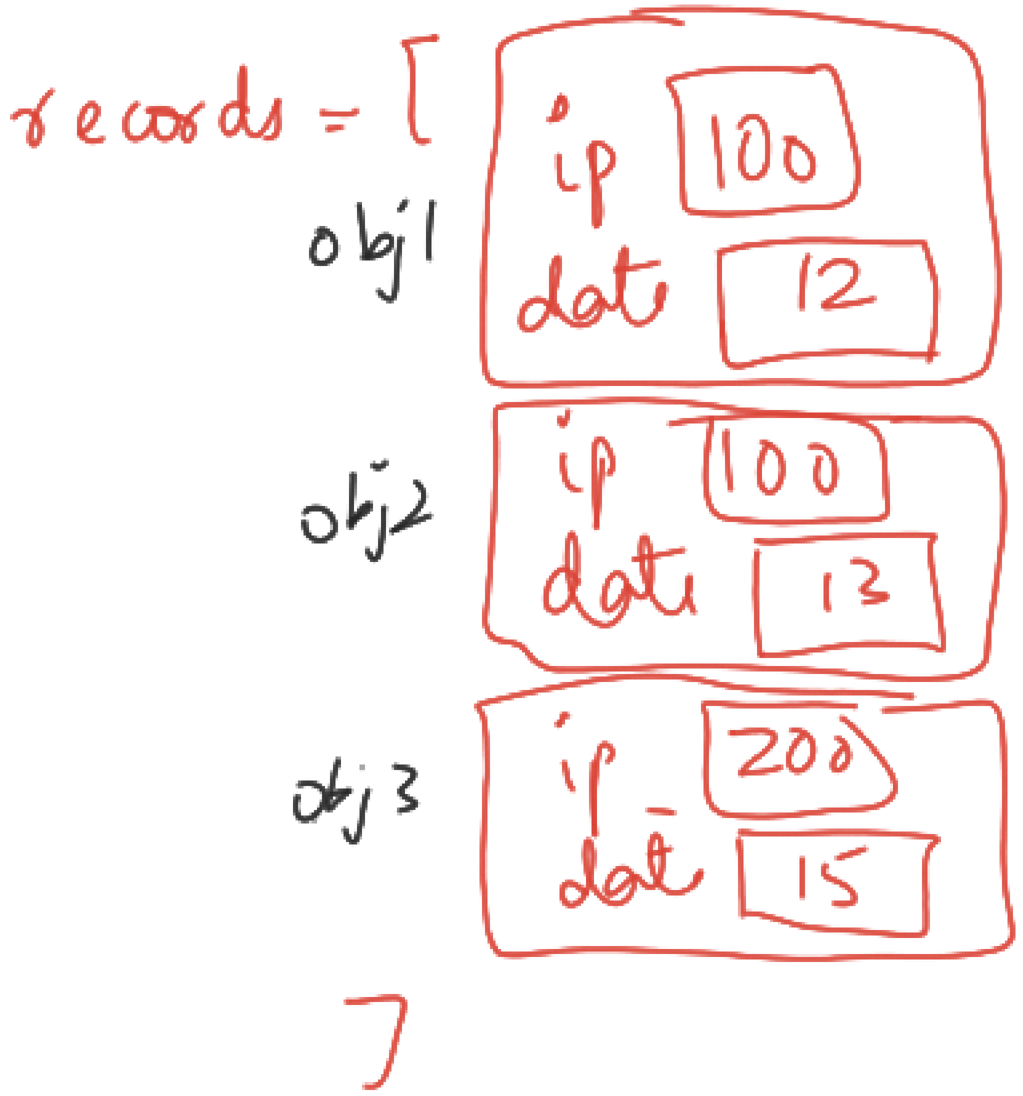
strs[j][i]

strs= hello 0
help 1
helium 2
hemp

prefix = "hel"

```
{
    0 : [h, h, h, h]
    1 : [e, e, e, e]
    2 : [l, l, l, m]
    3 : [l, p, i, p]
}
```

```
i = 2       0   1   2
cur = L     H   E   L   L   O
pre = HE    H   E   L   P
            H   E   M   P
            H   E   L   I   U   M
```



AL<String> uniqueIps

[100, 200]

AL<LogEntry> uniqIPs2

[obj1, obj2, obj3]

```
AL<String> al = new
String a = "hello" ;
Stringb = "hello";
al.contains(a); false
al.add(a);
al.contains(a); true
al.contains(b); true

a==b; false
a.equals(b); true

al ["hello", "world", "abc"]

al.contains("world"); → True
```

Sep14 ip1
Sep14 ip2
Sep14 ip1
Sep 7 ip1
Sep 6 ip3
Sep 7 ip4

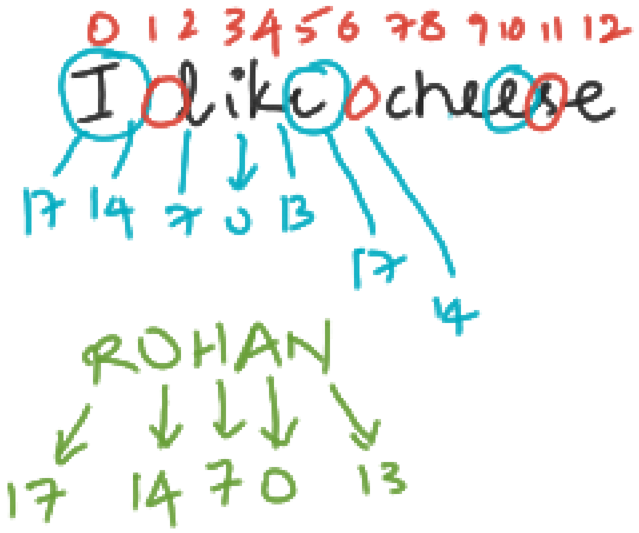
```
{
Aug30: [ip6],
Sep14: [ip1, ip2, ip1],
Sep7: [ip1, ip4),
Sep6: [ip3]
}
```

MS = 0
MD = Aug30

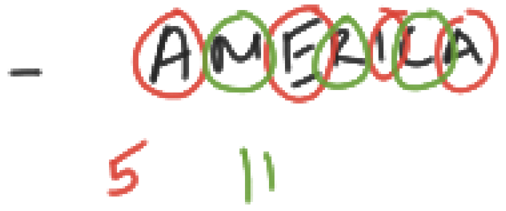
[ip1, ip2, ip1]

↓

{ ip1: 2, ip2: 1 }



```
CC cc1 = CC(17)
   cc2 = CC(14)
   cc3 = CC (7)
   cc4 = CC(0)
   cc5 =
         CC(13)

cc1.encrypt("Ice")
cc2.encrypt("_ _s")
```



```
CC cc1 = new CC(5)
CC cc2 = new CC (11)
cc1.encrypt("AEIA")
cc2.encrypt("MRC")
```

```
public class VigenereCipher {
        CC[] caesarCiphers;


    public VigenereCipher(String key){
        caesarCiphers = new CC[key.length()];
```

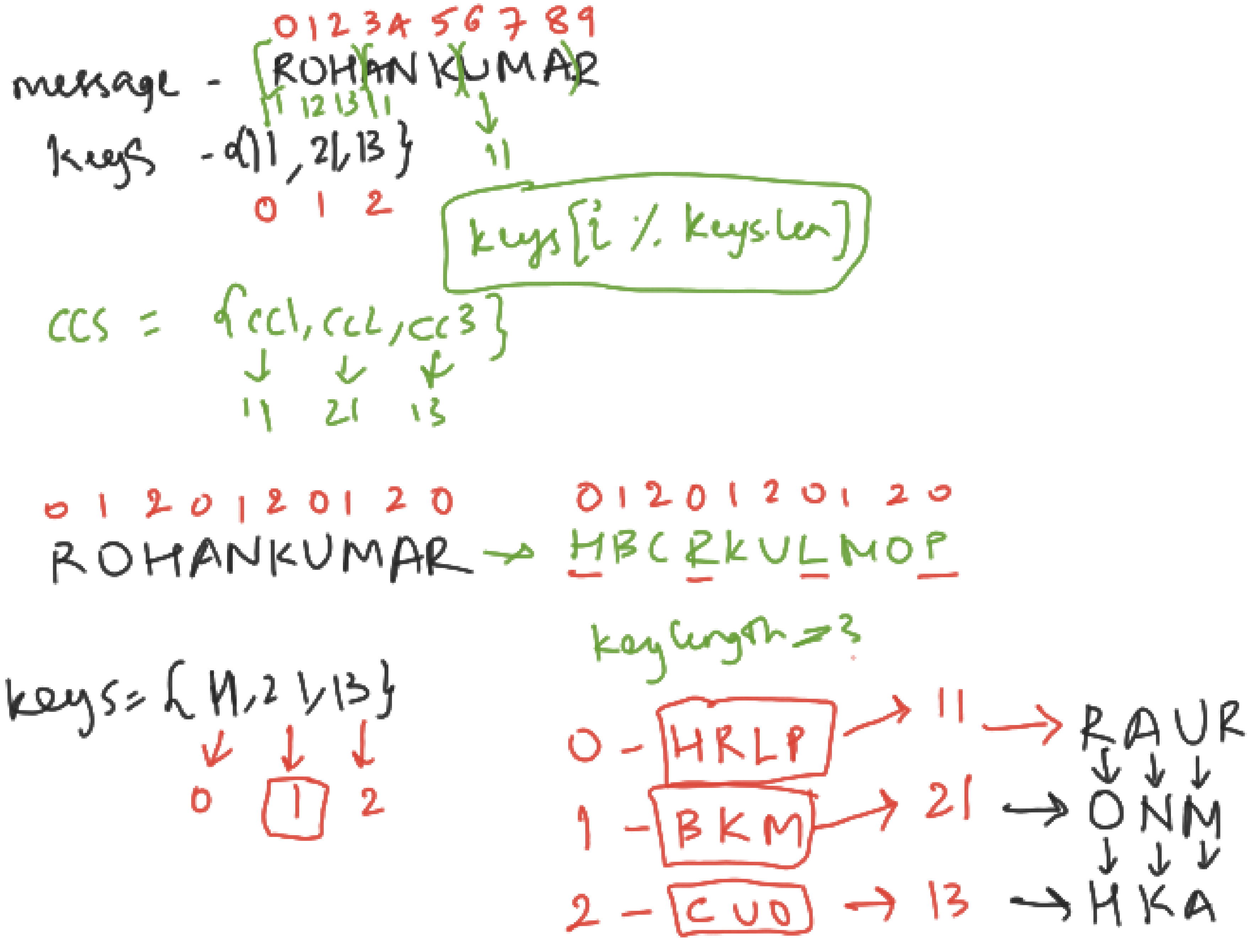
A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D

CHEESE

↓

GLIIWI

I → A ⇒ key = I - A


0 1 2 3 4 5 6 7 8 9
message - ROHANKUMAR
11 12 13 11
11
keys - {11, 21, 13}
0 1 2
keys[i % keys.len]
ccs = {cc1, cc2, cc3}
11 21 13
0 1 2 0 1 2 0 1 2 0
ROHANKUMAR
0 1 2 0 1 2 0 1 2 0
HBCRKULMOP
keylength = 3
keys = {11, 21, 13}
0 1 2
0 - HRLP
11
RAUR
1 - BKM
21
ONM
2 - CUD
13
HKA

	Eng	Spa	German
1	1	5	1
2	0	0	2
3	0	1758	0

Duke spec

DC_1

✓ DC_2

✓ DC_3

DC_4

✓ DC_5

UCSD speci

UC_1

UC_2

UC_3

UC_4

UC_5

Combined

✓ DC_2

✓ DC_3

UC_1

UC_2

Caesar Cipher

ABCDE
DEABC
key=3

DEA
BCD

```
CC cc = new CC(3);
e = cc.encrypt("DEA"); // BCD
d = cc.decrypt("BCD"); // DEA
```

"DEEBE"

↓ key=1

"EFFCF"

↳ most repeated : F

⇒ key = F-E
= 1

```
CC cc = new CC(1);
ans = cc.decrypt("EFFCF");
```

Vigenere Ciphere

MENSURATION
1 4 3 1 4 3 1 4 3 1 4

key = "B E D"
↓ ↓ ↓
1 4 3

MSAO $\xrightarrow{enc(1)}$ NTBP

EUTN $\xrightarrow{enc(4)}$ IXWR

NRI $\xrightarrow{enc(3)}$ QUL

"NIQTXUBWLPR"

MENSURATION

↓ enc (key - BED)

NIQTXUBWLPR

```
VC vc = new VC({1,4,3});
e = vc.encrypt("MENSURATION")
d = vc.decrypt(e);
```

NIQTXUBWLPR

key length = 3

lang = English

slice 0: NTBP → key = 1 given by Caesar cracker

slice 1: IXWR → key = 4

slice 2: QUL → key = 3

keys = {1, 4, 3}

VC vc = new VC(keys)

vc.decrypt(enc);

encrypted

lang - eng

<u>keylen = 1</u>

slice 0: → key = 5 from CC

keys = {5}

VC. decr (keys) → decrypted

↓

count #english words

↓

5

<u>key len = 2</u>

slice 0 → 3

slice 1 → 7

keys = {3, 7}

vc. decrypt (keys) → decrypt

↓

eng words count = 2

<u>key len = 3</u> eng words count = 21750

<u>keylen =3 4 ... 100</u>

#eng words

	Eng	Span	German
1	7	1	3
2	2	3	0
3	21770	2	1
4	3	5	0
⋮	⋮		
100	0	0	5

→ {1,4,3}

{1,4,3}

HELLO

↓ key len = 5

slice 0 : H →

slice 1 : E →

slice 2 : L →

slice 3 : L →

slice 4 : O →