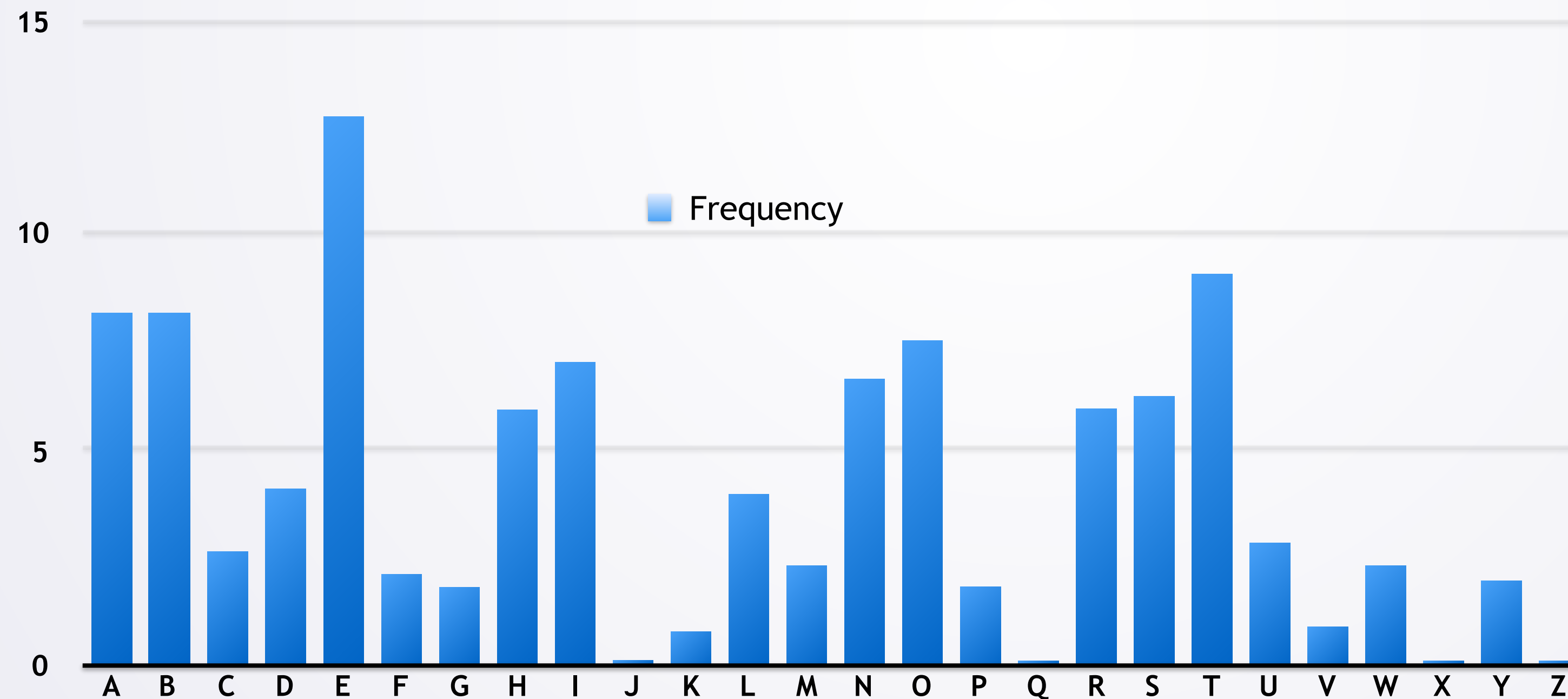


Breaking the Caesar Cipher

Developing an Algorithm

Decrypt Without Eyeball/Human

- Rely on letter frequencies in English
 - Use frequencies in other languages as needed
 - We will find the maximally occurring character
 - Assume this is 'e', find shift!



Counting Occurrences

Hi, do you want a lollipop today?
I own many good flavors,
but banana is outstanding.

- Code to scan text and increment counter

{0,0,0,0,0,0,0,0,0,0,0,0,0,0,0...,0,0,0}

0 1 2 3 4 5 6 7 8.....14

23 24 25

"A B C D E F G H I..... O

X Y Z"

Counting Occurrences

Hi, do you want a lollipop today?
I own many good flavors,
but banana is outstanding.

- Code to scan text and increment counter

{0,0,0,0,0,0,0,1,0,0,0,0,0,0...,0,0,0}

0 1 2 3 4 5 6 7 8.....14

23 24 25

"A B C D E F G H I..... O

X Y Z"

Counting Occurrences

Hi, do you want a lollipop today?
I own many good flavors,
but banana is outstanding.

- Code to scan text and increment counter

{0,0,0,0,0,0,0,1,1,0,0,0,0,0...,0,0,0}

0 1 2 3 4 5 6 7 8.....14

23 24 25

"A B C D E F G H I..... O

X Y Z"

Counting Occurrences

Hi, do you want a lollipop today?
I own many good flavors,
but banana is outstanding.

- Code to scan text and increment counter

{0,0,0,0,0,0,0,1,1,0,0,0,0,0...,0,0,0}

0 1 2 3 4 5 6 7 8.....14

23 24 25

"A B C D E F G H I..... O

X Y Z"

Counting Occurrences

Hi, do you want a lollipop today?
I own many good flavors,
but banana is outstanding.

- Code to scan text and increment counter

{0,0,0,0,0,0,0,1,1,0,0,0,0,0...,0,0,0}

0 1 2 3 4 5 6 7 8.....14

23 24 25

"A B C D E F G H I..... O

X Y Z "

Counting Occurrences

Hi, **d**o you want a lollipop today?
I own many good flavors,
but banana is outstanding.

- Code to scan text and increment counter

{0,0,0,1,0,0,0,1,1,0,0,0,0,0...,0,0,0}

0 1 2 3 4 5 6 7 8.....14

23 24 25

"A B C D E F G H I..... O

X Y Z"

Counting Occurrences

Hi, do you want a lollipop today?
I own many good flavors,
but banana is outstanding.

- Code to scan text and increment counter

{0,0,0,1,0,0,0,1,1,0,0,1,0,0...,0,0,0}

0 1 2 3 4 5 6 7 8.....14

23 24 25

"A B C D E F G H I..... O

X Y Z"

Counting Occurrences

Hi, do you want a lollipop today?
I own many good flavors,
but banana is outstanding.

- Code to scan text and increment counter

{0,0,0,1,0,0,0,1,1,0,0,1,0,0...,0,0,0}

0 1 2 3 4 5 6 7 8.....14

23 24 25

"A B C D E F G H I..... O

X Y Z"

Counting Occurrences

Hi, do you want a lollipop today?
I own many good flavors,
but banana is outstanding.

- Code to scan text and increment counter

{0,0,0,1,0,0,0,1,1,0,0,1,0,0...,0,1,0}

0 1 2 3 4 5 6 7 8.....14

23 24 25

"A B C D E F G H I..... O

X Y Z"

Counting Occurrences

Hi, do you want a lollipop today?
I own many good flavors,
but banana is outstanding.

- Code to scan text and increment counter

{0,0,0,1,0,0,0,1,1,0,0,2,0,0...,0,1,0}

0 1 2 3 4 5 6 7 8.....14

23 24 25

"A B C D E F G H I..... O

X Y Z"

Counting Occurrences

Hi, do you want a lollipop today?
I own many good flavors,
but banana is outstanding.

- Code to scan text and increment counter

{9,2,0,4,0,1,2,1,5,0,0,4,1,7,10,2,0,1,3,5,3,1,2,0,3,0}

0 1 2 3 4 5 6 7 8.....14

23 24 25

"A B C D E F G H I..... O

X Y Z "

Counting Occurrences with Code

```
String alph = "abcdefghijklmnopqrstuvwxyz";  
int[] counts = new int[26];  
for(int k=0; k < message.length(); k++){  
    char ch = Character.toLowerCase(message.charAt(k));  
    int dex = alph.indexOf(ch);  
    if (dex != -1){  
        counts[dex] += 1;  
    }  
}
```

Counting Occurrences with Code

```
String alph = "abcdefghijklmnopqrstuvwxyz";  
int[] counts = new int[26];  
for(int k=0; k < message.length(); k++){  
    char ch = Character.toLowerCase(message.charAt(k));  
    int dex = alph.indexOf(ch);  
    if (dex != -1){  
        counts[dex] += 1;  
    }  
}
```

- Scan message, if character is alphabetic:

Counting Occurrences with Code

```
String alph = "abcdefghijklmnopqrstuvwxyz";  
int[] counts = new int[26];  
for(int k=0; k < message.length(); k++){  
    char ch = Character.toLowerCase(message.charAt(k));  
    int dex = alph.indexOf(ch);  
    if (dex != -1){  
        counts[dex] += 1;  
    }  
}
```

- Scan message, if character is alphabetic:
 - Find location in "abcdefghijklmnopqrstuvwxyz"

Counting Occurrences with Code

```
String alph = "abcdefghijklmnopqrstuvwxyz";  
int[] counts = new int[26];  
for(int k=0; k < message.length(); k++){  
    char ch = Character.toLowerCase(message.charAt(k));  
    int dex = alph.indexOf(ch);  
    if (dex != -1){  
        counts[dex] += 1;  
    }  
}
```

- Scan message, if character is alphabetic:
 - Find location in "abcdefghijklmnopqrstuvwxyz"
 - Use index to increment one of 26 counters

From Algorithm to Code

```
public String decrypt(String encrypted) {  
    CaesarCipher cc = new CaesarCipher();  
    int[] freqs = countLetters(encrypted);  
    int maxDex = maxIndex(freqs);  
    int dkey = maxDex - 4;  
    if (maxDex < 4) {  
        dkey = 26 - (4-maxDex);  
    }  
    return cc.encrypt(encrypted, 26-dkey);  
}
```

- Count 26 frequencies of 'a'-'z', 'A'-'Z'

From Algorithm to Code

```
public String decrypt(String encrypted) {  
    CaesarCipher cc = new CaesarCipher();  
    int[] freqs = countLetters(encrypted);  
    int maxDex = maxIndex(freqs);  
    int dkey = maxDex - 4;  
    if (maxDex < 4) {  
        dkey = 26 - (4-maxDex);  
    }  
    return cc.encrypt(encrypted, 26-dkey);  
}
```

- Count 26 frequencies of 'a'-'z', 'A'-'Z'
 - Find largest value, assume it's 'e'

From Algorithm to Code

```
public String decrypt(String encrypted) {  
    CaesarCipher cc = new CaesarCipher();  
    int[] freqs = countLetters(encrypted);  
    int maxDex = maxIndex(freqs);  
    int dkey = maxDex - 4;  
    if (maxDex < 4) {  
        dkey = 26 - (4-maxDex);  
    }  
    return cc.encrypt(encrypted, 26-dkey);  
}
```

- Count 26 frequencies of 'a'-'z', 'A'-'Z'
 - Find largest value, assume it's 'e'
 - Find distance from 'e' which has index 4

From Algorithm to Code

```
public String decrypt(String encrypted) {  
    CaesarCipher cc = new CaesarCipher();  
    int[] freqs = countLetters(encrypted);  
    int maxDex = maxIndex(freqs);  
    int dkey = maxDex - 4;  
    if (maxDex < 4) {  
        dkey = 26 - (4 - maxDex);  
    }  
    return cc.encrypt(encrypted, 26 - dkey);  
}
```

- Count 26 frequencies of 'a'-'z', 'A'-'Z'
 - Find largest value, assume it's 'e'
 - Find distance from 'e' which has index 4

From Algorithm to Code

```
public String decrypt(String encrypted) {  
    CaesarCipher cc = new CaesarCipher();  
    int[] freqs = countLetters(encrypted);  
    int maxDex = maxIndex(freqs);  
    int dkey = maxDex - 4;  
    if (maxDex < 4) {  
        dkey = 26 - (4-maxDex);  
    }  
    return cc.encrypt(encrypted, 26-dkey);  
}
```

- Count 26 frequencies of 'a'-'z', 'A'-'Z'
 - Find largest value, assume it's 'e'
 - Find distance from 'e' which has index 4
 - Use 26-distance to decrypt using encrypt!

Values, Indexes, Details

- Array **freqs** has relationship between index and value, **freqs[8]** is how many times 'i' occurs
 - When looking for maximal value, return **index**, use to find distance from 'e' or 4 as shift

```
public int maxIndex(int[] vals) {  
    int maxDex = 0;  
    for(int k=0; k < vals.length; k++) {  
        if (vals[k] > vals[maxDex]) {  
            maxDex = k;  
        }  
    }  
    return maxDex;  
}
```


Values, Indexes, Details

- Array **freqs** has relationship between index and value, **freqs[8]** is how many times 'i' occurs
 - When looking for maximal value, return **index**, use to find distance from 'e' or 4 as shift
- Using the CaesarCipher class makes writing CaesarBreaker easier

Values, Indexes, Details

- Array **freqs** has relationship between index and value, **freqs[8]** is how many times 'i' occurs
 - When looking for maximal value, return **index**, use to find distance from 'e' or 4 as shift
- Using the CaesarCipher class makes writing CaesarBreaker easier
 - Call methods that work rather than rewriting