Object Oriented Caesar Cipher

Rewriting with Encapsulation

CaesarCipher

```java
public class CaesarCipher {
    public String encrypt(String input, int key) {
        String alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
        String shiftedAlphabet = alphabet.substring(key) +
                                 alphabet.substring(0,key);
        StringBuilder sb = new StringBuilder(input);
        for (int i = 0; i < sb.length(); i++) {
            char c = sb.charAt(i);
            int idx = alphabet.indexOf(c);
            .... ... ....  ....
```

CaesarCipher

```java
public class CaesarCipher {

  public String encrypt(String input, int key) {

    String alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";

    String shiftedAlphabet = alphabet.substring(key) +
                             alphabet.substring(0,key);

    StringBuilder sb = new StringBuilder(input);

    for (int i = 0; i < sb.length(); i++) {

        char c = sb.charAt(i);

        int idx = alphabet.indexOf(c);

        …. … …. ….
```

CaesarCipher

```java
public class CaesarCipher {

    private String alphabet;

    private String shiftedAlphabet;

    public CaesarCipher(int key) {

        alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";

        shiftedAlphabet = alphabet.substring(key) +

                          alphabet.substring(0,key);

    }

    public String encrypt(String input) {

        StringBuilder sb = new StringBuilder(input);

        for (int i = 0; i < sb.length(); i++) {

            char c = sb.charAt(i);
```

CaesarCipher

```java
public class CaesarCipher {

    private String alphabet;

    private String shiftedAlphabet;

    public CaesarCipher(int key) {

        alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";

        shiftedAlphabet = alphabet.substring(key) +

                          alphabet.substring(0,key);

    }

    public String encrypt(String input) {

        StringBuilder sb = new StringBuilder(input);

        for (int i = 0; i < sb.length(); i++) {

            char c = sb.charAt(i);
```

CaesarCipher

```java
public class CaesarCipher {

    private String alphabet;

    private String shiftedAlphabet;

    public CaesarCipher(int key) {

        alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";

        shiftedAlphabet = alphabet.substring(key) +

                          alphabet.substring(0,key);

    }

    public String encrypt(String input) {

        StringBuilder sb = new StringBuilder(input);

        for (int i = 0; i < sb.length(); i++) {

            char c = sb.charAt(i);
```

CaesarCipher

```
    }
public String encrypt(String input) {

    StringBuilder sb = new StringBuilder(input);

    for (int i = 0; i < sb.length(); i++) {

        char c = sb.charAt(i);

        int idx = alphabet.indexOf(c);

        if (idx != -1) {

            c = shiftedAlphabet.charAt(idx);

            sb.setCharAt(i, c);

        }

    }

    return sb.toString();
}
```

Old Way: No Encapsulation

- Old way:
  - CaesarCipher held no data
  - Passed in message + key

```
message: FIRST LEGION ATTACK EAST FLANK

key:  23
```

CaesarCipher

.encrypt(message, key)

Old Way: No Encapsulation

- Old way:
  - CaesarCipher held no data
  - Passed in message + key

```
CFOPQ IBDFLK XQQXZH BXPQ CIXKH
```

CaesarCipher

.encrypt(message, key)

Duke
UNIVERSITY

New Way: Encapsulate Key

- New way:
  - CaesarCipher encapsulates key
  - Pass in message only (CaesarCipher has key)

```
CFOPQ IBDFLK XQQXZH BXPQ CIXKH
```

CaesarCipher

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
XYZABCDEFGHIJKLMNOPQRSTUVW
```
.encrypt(message)

What Is Benefit?

- Same functionality
  - What is benefit of OO approach?
  - Easier to think of Cipher as logical unit w/ key

Old	New

```
message: FIRST LEGION ATTACK EAST FLANK
    key: 23
```

↓

```
CaesarCipher
```

↓

```
CFOPQ IBDFLK XQQXZH BXPQ CIXKH
```

```
message: FIRST LEGION ATTACK EAST FLANK
```

↓

```
CaesarCipher
ABCDEFGHIJKLMNOPQRSTUVWXYZ
XYZABCDEFGHIJKLMNOPQRSTUVW
```

↓

```
CFOPQ IBDFLK XQQXZH BXPQ CIXKH
```

Duke
UNIVERSITY