

Credit
Extra?
Assignment
Security Suite

How has your program changed from planning to coding to now? Please explain?

```

1 package securitysuite;
2
3 import java.util.Scanner;
4
5 public class Main {
6
7     public static void main(String[] args) {
8
9         Scanner scanner = new Scanner(System.in);
10        CaesarCipher caesar = new CaesarCipher();
11        VigenerCipher vigenere = new VigenerCipher();
12        FileProcessor fileProcessor = new FileProcessor();
13
14        while (true) {
15            System.out.println("\n==== CipherGuard ===");
16            System.out.println("[1] Caesar Encrypt Message");
17            System.out.println("[2] Caesar Decrypt Message");
18            System.out.println("[3] Encrypt File (Caesar)");
19            System.out.println("[4] Brute Force Crack (Caesar)");
20            System.out.println("[5] Vigener Encrypt");
21            System.out.println("[6] Vigener Decrypt");
22            System.out.println("[7] Exit");
23            System.out.print("Choice: ");
24
25            int choice = scanner.nextInt();
26            scanner.nextLine(); // clear buffer
27
28        switch (choice) {
29
30            case 1: // Caesar Encrypt
31                System.out.print("Enter message: ");
32                String msg = scanner.nextLine();
33
34                System.out.print("Enter shift value: ");
35                int shift = scanner.nextInt();
36                scanner.nextLine();
37
38                String encrypted = caesar.encrypt(msg, shift);
39                System.out.println("Encrypted: " + encrypted);
40                break;
41
42            case 2: // Caesar Decrypt
43                System.out.print("Enter encrypted message: ");
44                String encMsg = scanner.nextLine();
45
46                System.out.print("Enter shift value used: ");
47                int dShift = scanner.nextInt();
48                scanner.nextLine();
49
50                String decrypted = caesar.decrypt(encMsg, dShift);
51                System.out.println("Decrypted: " + decrypted);
52                break;
53

```

```
46         System.out.print(" Enter shift value used: ");
47         int dShift = scanner.nextInt();
48         scanner.nextLine();
49
50         String decrypted = caesar.decrypt(encMsg, dShift);
51         System.out.println("Decrypted: " + decrypted);
52         break;
53
54     case 3: // File Encrypt
55         System.out.print("Enter input file (.txt): ");
56         String input = scanner.nextLine();
57
58         System.out.print("Enter output file name: ");
59         String output = scanner.nextLine();
60
61         System.out.print("Enter shift value: ");
62         int fileShift = scanner.nextInt();
63         scanner.nextLine();
64
65         fileProcessor.encryptFile(input, output, fileShift);
66         break;
67
68     case 4: // Brute force
69         System.out.print("Enter encrypted message: ");
70         String brute = scanner.nextLine();
71         fileProcessor.bruteForceCrack(brute);
72         break;
73
74     case 5: // Vigenere Encrypt
75         System.out.print("Enter message: ");
76         String vMsg = scanner.nextLine();
77
78         System.out.print("Enter keyword: ");
79         String key = scanner.nextLine();
80
81         String vEncrypted = vigenere.encrypt(vMsg, key);
82         System.out.println("Encrypted: " + vEncrypted);
83         break;
84
85     case 6: // Vigenere Decrypt
86         System.out.print("Enter encrypted message: ");
87         String vEnc = scanner.nextLine();
88
89         System.out.print("Enter keyword: ");
90         String vKey = scanner.nextLine();
91
92         String vDecrypted = vigenere.decrypt(vEnc, vKey);
93         System.out.println("Decrypted: " + vDecrypted);
94         break;
95
96     case 7:
97         System.out.println("Exiting CipherGuard.");
98         System.out.println("-----");
99
100        String vDecrypted = vigenere.decrypt(vEnc, vKey);
101        System.out.println("Decrypted: " + vDecrypted);
102        break;
103
104    case 7:
105        System.out.println("Exiting CipherGuard.");
106        scanner.close();
107        return;
108
109    default:
110        System.out.println("Invalid choice.");
111    }
112}
113}
```

```
1 package securitysuite;
2
3 import java.io.*;
4
5 public class FileProcessor {
6
7     private CaesarCipher cipher = new CaesarCipher();
8
9     // Encrypt file from securitysuite package folder
10    public void encryptFile(String inputFile, String outputFile, int shift) {
11
12        // Force path to src/securitysuite folder
13        String basePath = "src/securitysuite/";
14        String inputPath = basePath + inputFile;
15        String outputPath = basePath + outputFile;
16
17        try {
18            BufferedReader reader = new BufferedReader(new FileReader(inputPath));
19            BufferedWriter writer = new BufferedWriter(new FileWriter(outputPath))
20        } {
21            String line;
22
23            while ((line = reader.readLine()) != null) {
24                writer.write(cipher.encrypt(line, shift));
25                writer.newLine();
26            }
27
28            System.out.println("File encrypted successfully > saved in securitysuite package");
29
30        } catch (IOException e) {
31            System.out.println("File error: " + e.getMessage());
32            System.out.println("Make sure file is inside src/securitysuite/");
33        }
34    }
35
36    // Brute force Caesar crack
37    public void bruteForceCrack(String encryptedText) {
38        long start = System.currentTimeMillis();
39
40        System.out.println("\nBrute force results:\n");
41
42        for (int shift = 0; shift < 26; shift++) {
43            String attempt = cipher.decrypt(encryptedText, shift);
44            System.out.println("Shift " + shift + ": " + attempt);
45        }
46
47        long duration = System.currentTimeMillis() - start;
48        System.out.println("\nCompleted in: " + duration + " ms");
49    }
50}
```

```

1 package securitysuite;
2
3 public class VigenereCipher {
4
5     public String encrypt(String text, String keyword) {
6         StringBuilder result = new StringBuilder();
7         keyword = keyword.toLowerCase();
8
9         int keyIndex = 0;
10
11        for (char c : text.toCharArray()) {
12            if (Character.isLetter(c)) {
13                int shift = keyword.charAt(keyIndex % keyword.length()) - 'a';
14                char base = Character.isUpperCase(c) ? 'A' : 'a';
15
16                int newChar = (c - base + shift) % 26;
17                c = (char) (newChar + base);
18
19                keyIndex++;
20            }
21            result.append(c);
22        }
23        return result.toString();
24    }
25
26    public String decrypt(String text, String keyword) {
27        StringBuilder result = new StringBuilder();
28        keyword = keyword.toLowerCase();
29
30        int keyIndex = 0;
31
32        for (char c : text.toCharArray()) {
33            if (Character.isLetter(c)) {
34                int shift = keyword.charAt(keyIndex % keyword.length()) - 'a';
35                char base = Character.isUpperCase(c) ? 'A' : 'a';
36
37                int newChar = (c - base - shift) % 26;
38                if (newChar < 0) {
39                    newChar += 26;
40                }
41
42                c = (char) (newChar + base);
43                keyIndex++;
44            }
45            result.append(c);
46        }
47        return result.toString();
48    }
49 }
50

```

```

1 package securitysuite;
2
3 public class CaesarCipher {
4
5     public String encrypt(String text, int shift) {
6         StringBuilder result = new StringBuilder();
7
8         for (char c : text.toCharArray()) {
9             if (Character.isLetter(c)) {
10                 char base = Character.isUpperCase(c) ? 'A' : 'a';
11
12                 int newChar = (c - base + shift) % 26;
13                 if (newChar < 0) {
14                     newChar += 26;
15                 }
16
17                 c = (char) (newChar + base);
18             }
19             result.append(c);
20         }
21         return result.toString();
22     }
23
24     public String decrypt(String text, int shift) {
25         return encrypt(text, -shift);
26     }
27 }
28

```

Reflection and test file

The screenshot shows a Java code editor interface with several tabs at the top: CaesarCipher.java, VigenereCipher.java, FileProcessor.java, Main.java, and test.txt. The test.txt tab is currently selected. The code in the editor consists of two numbered comments:

```
1 Programmers must understand encryption to protect personal data, secure systems, and prevent cybercrime.  
2 Understanding encryption helps developers defend against hackers, safeguard privacy, and build trustworthy technology.
```