```
package codecompressor;
import java.util.HashMap;
import java.util.Map;
import java.util.regex.Pattern;

/**
 *
 * @author adari
 */
public class CodeCompressor {
    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        // TODO code application logic here
        String program =
                    " /*\n"
                + " *Function to chop a string in half.\n"
                + " */\n"
                + " public static string chop(string input) {\n"
                +     " if (input == null || input.isEmpty()) {\n"
                +         " return input;\n"
                +     " }\n"
                +     " if (input.length() % 2 == 1) {\n"
                +         " return \"cannot chop an odd-length string in half\";\n"
                +     " }\n"
                + " return input.substring(input.length() / 2);\n"
                + " }\n";
```

```
        System.out.println(" Example: "+ minimize("you say yes, I say no you say stop and I say go go go\n"));
        System.out.println(minimize(program));
    }
    public static String minimize(String code) {
        int value = 0;
        Map <String, Integer> temp = new HashMap <String, Integer>();
        String[] alphabet = code.split("(?=[\\P{Alpha}+])|(?<=[\\P{Alpha}+])");
        StringBuilder sb = new StringBuilder();
        Pattern alphabets = Pattern.compile("^[a-zA-Z]+");
        for (String str : alphabet) {
            if (alphabets.matcher(str).matches()) {
                if (temp.containsKey(str)) {
                    sb.append("$" + Integer.toString(temp.get(str)));
                }
                else {
                    temp.put(str, value);
                    sb.append(str);
                }
                value++;
            }
            else {
                sb.append(str);
            }
        }
        String minimizedString = sb.toString();
        return minimizedString;
    }
}
```