Exercise 6: Gene Finding

String client example: gene finding

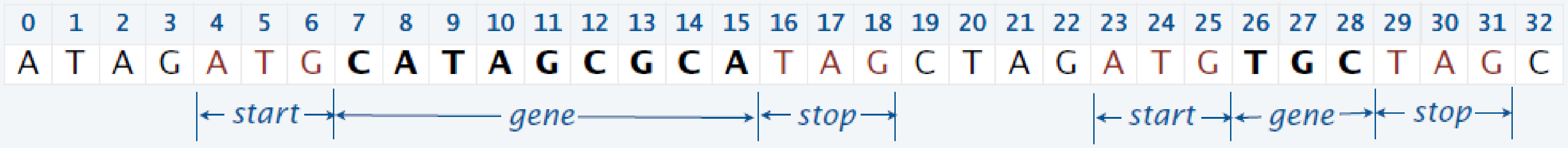
Pre-genomics era. Sequence a human genome.

Post-genomics era. Analyze the data and understand structure.

Genomics. Represent genome as a string over A C T G alphabet.

Gene. A substring of genome that represents a functional unit.

- Made of *codons* (three A C T G *nucleotides*).
- Preceded by ATG (*start* codon).
- Succeeded by TAG, TAA, or TGA (*stop* codon).



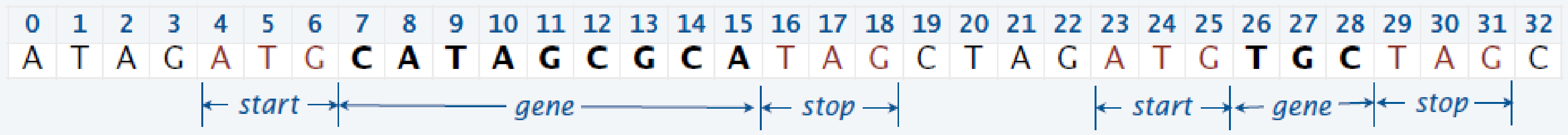
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
A	T	A	G	A	T	G	**C**	**A**	**T**	**A**	**G**	**C**	**G**	**C**	**A**	T	A	G	C	T	A	G	A	T	G	**T**	**G**	**C**	T	A	G	C

Goal. Write a Java program to find genes in a given genome.

Exercise Six (Oct. 23), Last submitting Date: Nov. 1.

String client exercise: Gene finding

Goal. Write a Java program to find genes in a given genome.



0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
A	T	A	G	A	T	G	**C**	**A**	**T**	**A**	**G**	**C**	**G**	**C**	**A**	T	A	G	C	T	A	G	A	T	G	**T**	**G**	**C**	T	A	G	C

Algorithm. Scan left-to-right through dna.

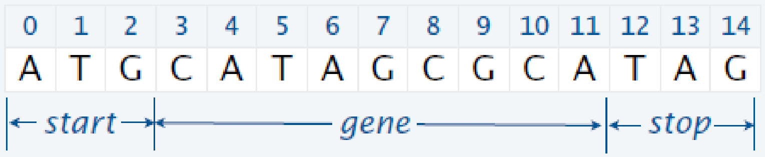
- If start codon ATG found, set beg to index i.
- If stop codon found and substring length is a multiple of 3, print gene and reset beg to -1.

i	codon		beg	*output*	*remainder of input string*
	start	*stop*			
0			-1		ATAGATGCATAGCGCATAGCTAGATGTGCTAGC
1		TAG	-1		TAGATGCATAGCGCATAGCTAGATGTGCTAGC
4	ATG		4		ATGCATAGCGCATAGCTAGATGTGCTAGC
9		TAG	4		TAGCGCATAGCTAGATGTGCTAGC
16		TAG	4	CATAGCGCA	TAGCTAGATGTGCTAGC
20		TAG	-1		TAGATGTGCTAGC
23	ATG		23		ATGTGCTAGC
29		TAG	23	TGC	TAGC

Implementation. Entertaining programming exercise!

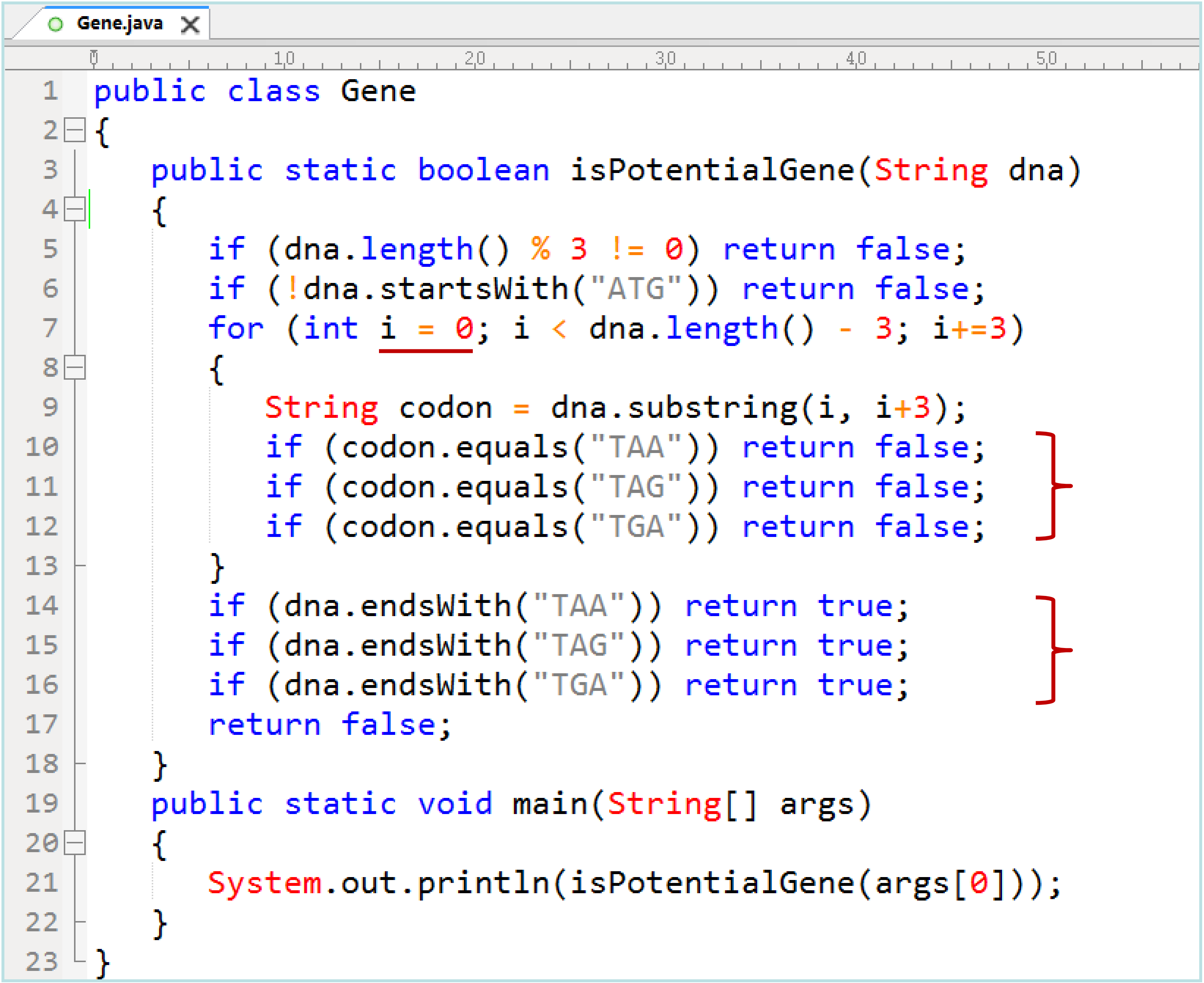
String client warmup: Identifying a potential gene

Goal. Write a Java program to determine whether a given string is a potential gene.

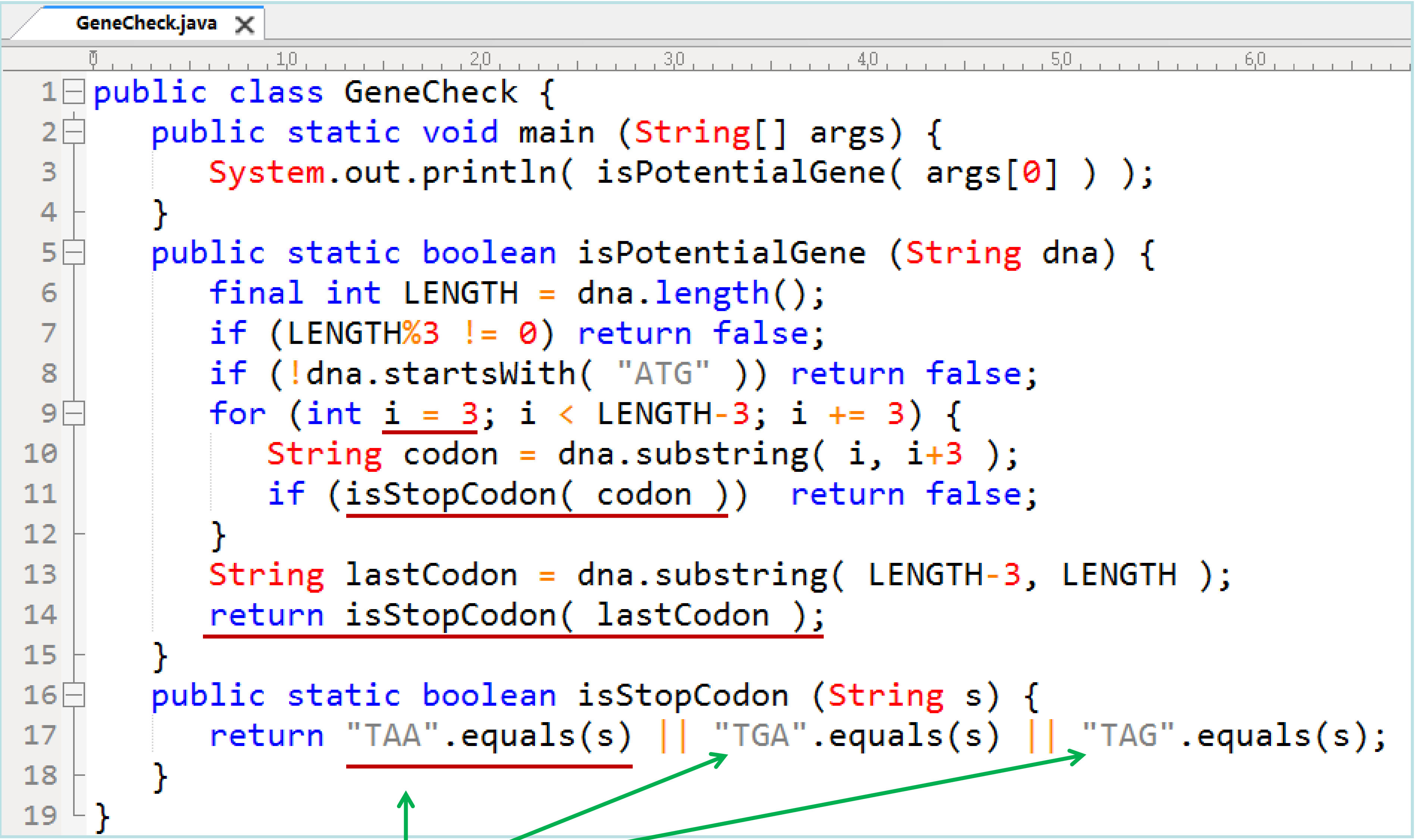


```
% java Gene ATGCATAGCGCATAG
true
% java Gene ATGCGCTGCGTCTGTACTAG
false
% java Gene ATGCCGTGACGTCTGTACTAG
false
```

```
public class Gene
{
    public static boolean isPotentialGene(String dna)
    {
        if (dna.length() % 3 != 0) return false;
        if (!dna.startsWith("ATG")) return false;
        for (int i = 0; i < dna.length() - 3; i+=3)
        {
            String codon = dna.substring(i, i+3);
            if (codon.equals("TAA")) return false;
            if (codon.equals("TAG")) return false;
            if (codon.equals("TGA")) return false;
        }
        if (dna.endsWith("TAA")) return true;
        if (dna.endsWith("TAG")) return true;
        if (dna.endsWith("TGA")) return true;
        return false;
    }
    public static void main(String[] args)
    {
        StdOut.println(isPotentialGene(args[0]));
    }
}
```


Gene.java

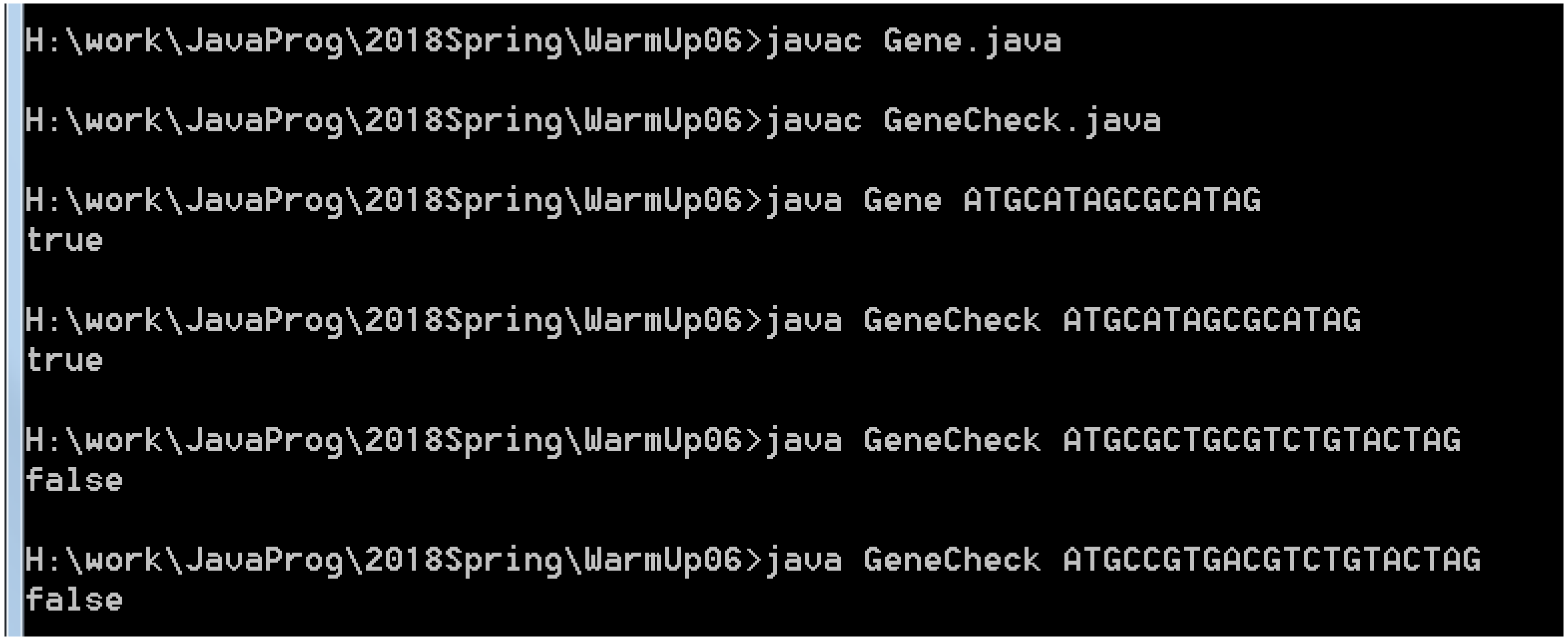
```java
public class Gene
{
    public static boolean isPotentialGene(String dna)
    {
        if (dna.length() % 3 != 0) return false;
        if (!dna.startsWith("ATG")) return false;
        for (int i = 0; i < dna.length() - 3; i+=3)
        {
            String codon = dna.substring(i, i+3);
            if (codon.equals("TAA")) return false;
            if (codon.equals("TAG")) return false;
            if (codon.equals("TGA")) return false;
        }
        if (dna.endsWith("TAA")) return true;
        if (dna.endsWith("TAG")) return true;
        if (dna.endsWith("TGA")) return true;
        return false;
    }
    public static void main(String[] args)
    {
        System.out.println(isPotentialGene(args[0]));
    }
}
```



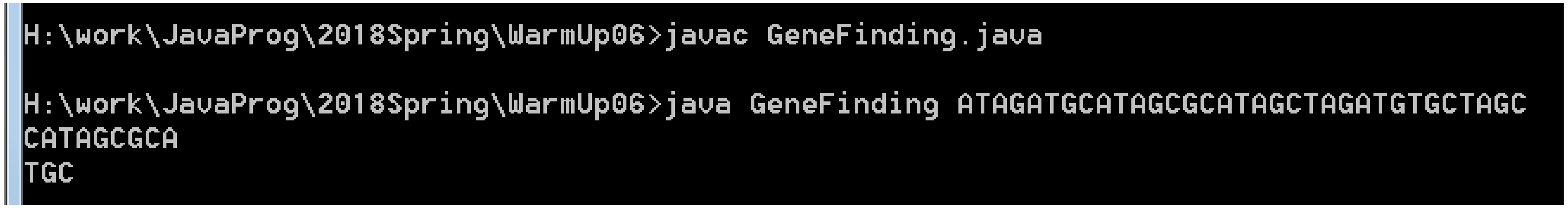
GeneCheck.java

```java
public class GeneCheck {
    public static void main (String[] args) {
        System.out.println( isPotentialGene( args[0] ) );
    }
    public static boolean isPotentialGene (String dna) {
        final int LENGTH = dna.length();
        if (LENGTH%3 != 0) return false;
        if (!dna.startsWith( "ATG" )) return false;
        for (int i = 3; i < LENGTH-3; i += 3) {
            String codon = dna.substring( i, i+3 );
            if (isStopCodon( codon ))  return false;
        }
        String lastCodon = dna.substring( LENGTH-3, LENGTH );
        return isStopCodon( lastCodon );
    }
    public static boolean isStopCodon (String s) {
        return "TAA".equals(s) || "TGA".equals(s) || "TAG".equals(s);
    }
}
```

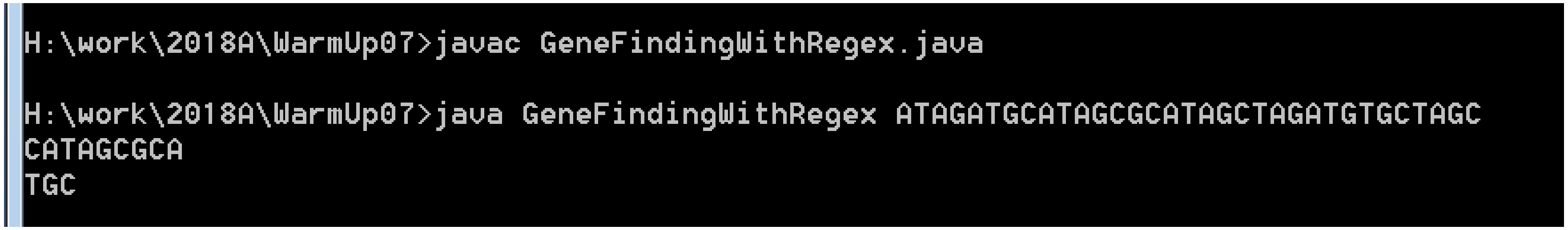
An Implementation Pattern (Idiom): When compare a variable and a literal Strings, always put the literal String as the first object, so that make the code work without runtime Exception even the variable String is null.



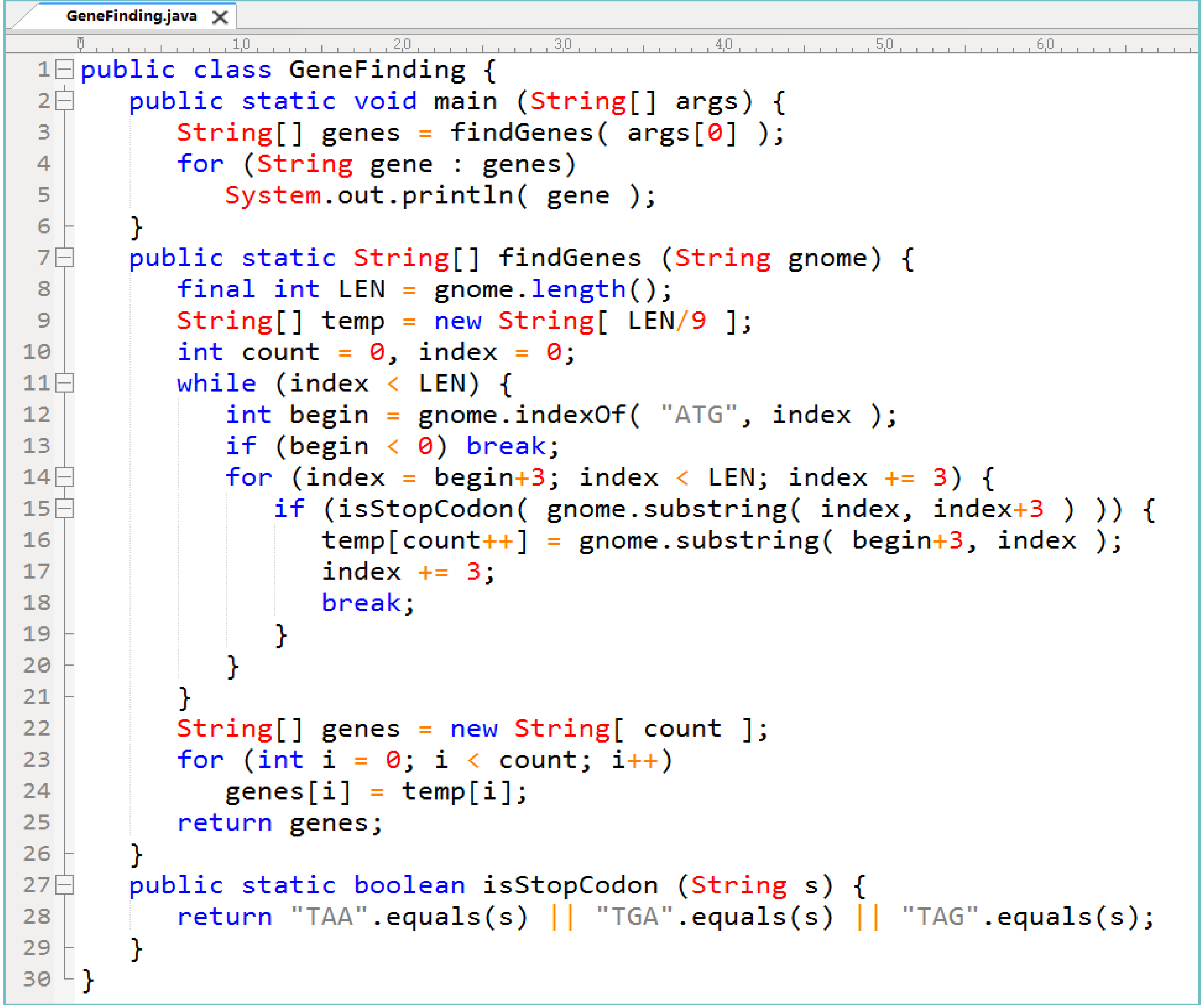
```
H:\work\JavaProg\2018Spring\WarmUp06>javac Gene.java

H:\work\JavaProg\2018Spring\WarmUp06>javac GeneCheck.java

H:\work\JavaProg\2018Spring\WarmUp06>java Gene ATGCATAGCGCATAG
true

H:\work\JavaProg\2018Spring\WarmUp06>java GeneCheck ATGCATAGCGCATAG
true

H:\work\JavaProg\2018Spring\WarmUp06>java GeneCheck ATGCGCTGCGTCTGTACTAG
false

H:\work\JavaProg\2018Spring\WarmUp06>java GeneCheck ATGCCGTGACGTCTGTACTAG
false
```



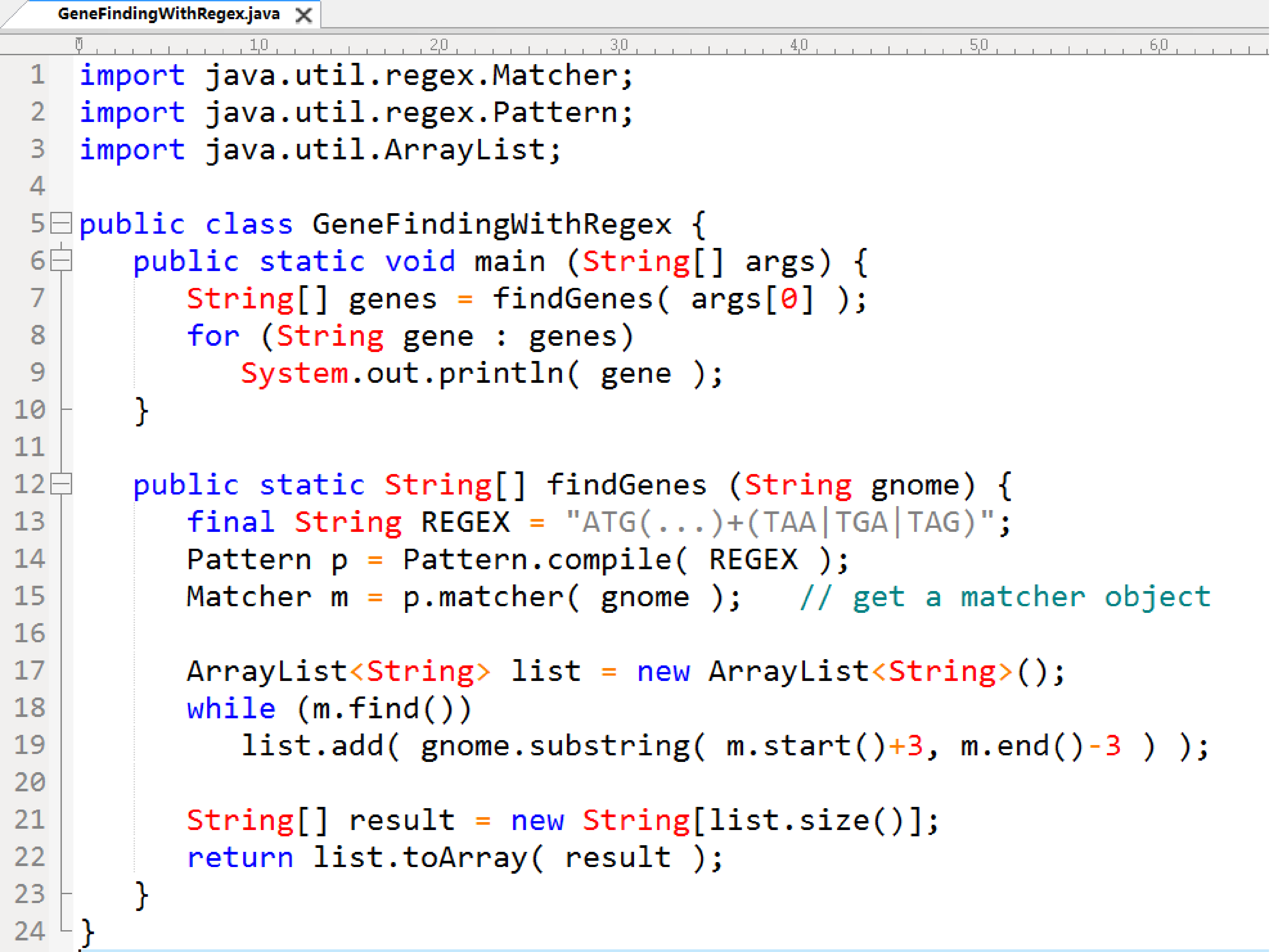
```
H:\work\JavaProg\2018Spring\WarmUp06>javac GeneFinding.java

H:\work\JavaProg\2018Spring\WarmUp06>java GeneFinding ATAGATGCATAGCGCATAGCTAGATGTGCTAGC
CATAGCGCA
TGC
```



```
H:\work\2018A\WarmUp07>javac GeneFindingWithRegex.java

H:\work\2018A\WarmUp07>java GeneFindingWithRegex ATAGATGCATAGCGCATAGCTAGATGTGCTAGC
CATAGCGCA
TGC
```


GeneFinding.java

```java
public class GeneFinding {
   public static void main (String[] args) {
      String[] genes = findGenes( args[0] );
      for (String gene : genes)
         System.out.println( gene );
   }
   public static String[] findGenes (String gnome) {
      final int LEN = gnome.length();
      String[] temp = new String[ LEN/9 ];
      int count = 0, index = 0;
      while (index < LEN) {
         int begin = gnome.indexOf( "ATG", index );
         if (begin < 0) break;
         for (index = begin+3; index < LEN; index += 3) {
            if (isStopCodon( gnome.substring( index, index+3 ) )) {
               temp[count++] = gnome.substring( begin+3, index );
               index += 3;
               break;
            }
         }
      }
      String[] genes = new String[ count ];
      for (int i = 0; i < count; i++)
         genes[i] = temp[i];
      return genes;
   }
   public static boolean isStopCodon (String s) {
      return "TAA".equals(s) || "TGA".equals(s) || "TAG".equals(s);
   }
}
```



```java
import java.util.regex.Matcher;
import java.util.regex.Pattern;
import java.util.ArrayList;

public class GeneFindingWithRegex {
   public static void main (String[] args) {
      String[] genes = findGenes( args[0] );
      for (String gene : genes)
         System.out.println( gene );
   }

   public static String[] findGenes (String gnome) {
      final String REGEX = "ATG(...)+(TAA|TGA|TAG)";
      Pattern p = Pattern.compile( REGEX );
      Matcher m = p.matcher( gnome );   // get a matcher object

      ArrayList<String> list = new ArrayList<String>();
      while (m.find())
         list.add( gnome.substring( m.start()+3, m.end()-3 ) );

      String[] result = new String[list.size()];
      return list.toArray( result );
   }
}
```