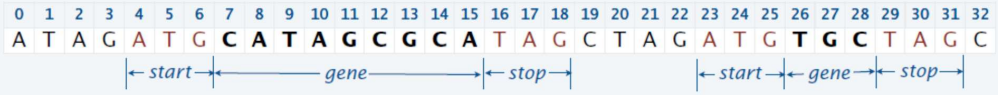
String client example: gene finding

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.

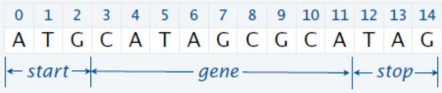
- Made of *codons* (three A C T G *nucleotides*).
- Preceded by ATG (*start* codon).
- Succeeded by TAG, TAA, or TGA (*stop* codon).



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

String client warmup: Identifying a potential gene

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

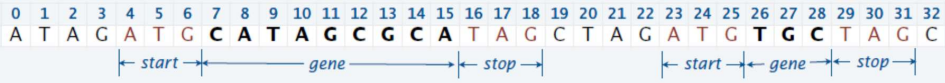



```
% java Gene ATGCATAGCGCATAG
true
% java Gene ATGCGCTGCGTCTGTACTAG
false
% java Gene ATGCCGTGACGTCTGTACTAG
false
```

```
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]));
    }
}
```

String client exercise: Gene finding

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



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.

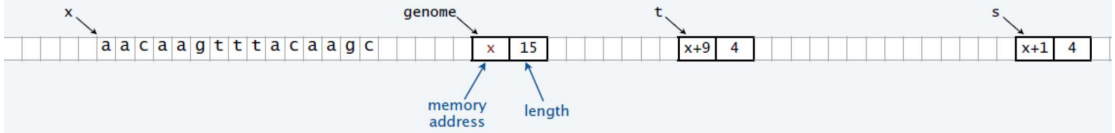
i	codon start	codon stop	beg	output	remainder of input string
0			-1		ATAGATGCATAGCGCATAGCTAGATGTGCTAGC
1		TAG	-1		TAGATGCATAGCGCATAGCTAGATGTGCTAGC
4	ATG		4		ATGCATAGCGCATAGCTAGATGTGCTAGC
9		TAG	4		TAGCGCATAGCTAGATGTGCTAGC
16		TAG	4	CATAGCGCA	TAGCTAGATGTGCTAGC
20		TAG	-1		TAGATGTGCTAGC
23	ATG		23		ATGTGCTAGC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

Implementation. Entertaining programming exercise!

OOP context for strings

Possible memory representation of

```
String genome = "aacaagtttacaagc";
String s = genome.substring(1, 5);
String t = genome.substring(9, 13);
```



Implications

- s and t are different strings that share the same value "acaa".
- (s == t) is false (because it compares addresses).
- (s.equals(t)) is true (because it compares character sequences).
- Java String interface is more complicated than the API.