1. Use Case

Description	A set of unit test that call different Bioinformatics Algorithms which are provided by a set of exercises
Main Path	1. The Unit Test are executed a. Each Unit Test will test a different algorithm with different inputs to an existing algorithm including the following: • Pretest provide by the set of exercises • Posttest provides by the set of exercises • Edge cases such as empty string and null for the input parameters b. Each unit test will verify that the output is correct c. The output will be display to the user on the console
Actors	A person or another program • (Person mainly) → understand how to code an algorithms • Using the library to create a new algorithm • Analyzing a dataset
Prerequisites	None
Post Condition	All the test have executed and the result have been verified and presented to the user

2. Assumptions

All input will be done by reading files. However, this could later change by implementing Restful Service or Microservices.

3. High Level View

Compiling the program

1. mvn install – Create the class files and run the test programs that test the utility code and run the bioinformatic algorithms
2. Configuration Management command line programs
   - mvn javadoc:javadoc – Creates the html files generated from the javadoc.
   - mvn fmt;format – Formats the files to the Google Standard
   - tidy -xml -i pom.xml > .tmp ; mv .tmp pom.xml

Function Libraries

A library that contains a set of classes to make it easier to program.

Object	Description
MyLogger	A class that creates and configures a logger that the developer can write messages to. In order to keep the coupling low the classes that use this object must convert the object into a String.
MyInputFile	A class to read the text from a text file. Built my own to experiment with Streams
ReturnPacket	An object that be returned from any function and/ or tier. Contains a variable for an error code, error message and object for a class that is returned.
JsonJacksonWrapper	A wrapper around the Faster XML Jackson Library to Read JSON Files.
JsonWrapper	An interface to convert JSON Files into Java Classes. Allows for the user to use mock objects or the simple integration of other JSON Files into java Classes into the environment

Types

Object	Description
NucleicAcidPolymerType	The base class or class that describes any Nucleic AcidPolymer. Contains functions that can operate on any nucleic acid without know anything specific about the nucleic acid Example of children classes are: • Deoxyribonucleic Acid, • RiboNucleicAcids
DeoxyribenucleicAcidType	The class that models DNA. Contains functions that operatr on DNA. Examples are • convertToRNA • complement
RibonucleicAcidType	The class that models RNA. Currently there are no functions, but this type is created by other functions found in the Deoxyribonucleic Acid
NucleicAcidInformation	A class that contains fact about a specific nucleic acid such as Deoxyribonucleic Acid or Ribonucleic Acid. Example of information that they can contain are the following : • Names of base pairs that can exist in the polymer • Opposites of each of the bases Usually each NucleicAcidInformation will have an “has” relationship with a class that inherits from NucleicAcidPolymer

Overall Structure

The TestIntroductionBioinformatics Class will contain a method with the @Test for each algorithm. Currently we have the following algorithms

- Introductory Algorithms
  - CountNucleotides
  - TranscribeDNA
  - reverseComplement

The algorihtm will have the following structure

- A call to the member function data in TestIntroductionBioinformatics which rereives the preTest and postTest (name and data) and the default test functions (when the polymer is either null or empty. The call will produce the following map: LinkedHashMap<Stirng, NucleicAcidPolymertestCase>

- The map is iterated through

  - During the iteration the first function called will execute is the function that performs the algorithm.

    Problem : Attempt to make the funcitons generic, but the type erase was causing issues (two functions turned out to have the same signature). The solution devised was to use the Stratergy Pattern

Strategy Pattern	Description
ConvertDnaToRnaStrategy	The input is DNA and the output is RNA
Count Strategy	Start with a Polymer such as DNA or RNA and get a map of the nucleotides and the count of each nucleictides
OperateOnDnaFunctionStrategy	The input is a DNA and the output is a DNA

  - During the iteration the second function will validate the data. There is a collection of functions that will help the developer valide the algorithm is correct.

Validation Function	Description
CountValidate	Verifies the count returned in a Map are correct
PolymerValidate	Verifies the polymer craeated (DNA, RNA etc…) matches the correct solution polymer

Lessons Learned

Infrastructure

Running Junit Test

Surefire Test was not working when made a dependency in the POM File. However, when it was designated a plugin in the POM File then it worked. A plugin provies a set of goals that can be executed.

The syntax usuaally is mvn [plugin-name]:[goal-name]

The type of plugins are:

Build Plugins which are executed during the Build Phase

Reporting Plugins that are executed during the reporting phase

Each plugin should have the Maven Coordinates (group id, artifact id, version)

Maven

Formatting the Code

The project will use com.ooveo.fmt-maven-plugin to format the code. To execute the plugin the following command must be executed:

mvn fmt:fmt

Creating the Javadoc

mvn javadoc:java

Plugins

An advantage of maven is the ability to use plugins. The plugin can be used from project to project, so acquiring a collection of plugins can help to improve the quality for this project and future projects. My set of plugins include the following.

Plugin	Description
maven-pmd-plugin	Examines the code for common programming issues by analyzing the source code
spotbug-maven-plugin	Uses static analysis to look for bugs
Maven-checkstyle-plugin-standard l	Help programmers to adhere to coding standard

Maven – Find the Goals of a plugin

The goals of a plugin can be displayed.

mvn help:describe

-DgroupId=com.coveo

-DartifactId=fmt-maven-plugin

-Dversion=2.8

The gorupId, artifactId, version will be different for each plugin