Name – Bablu Kumar

Course – M. Tech Computational Biology (1st Year)

Register No- 20310004

Department- Centre for Bioinformatics, Pondicherry University.

JAVA ASSIGNMENT

Submitted to – Mr. Pranavathiyani G

Date of submission – 06-06-2021

- **com.sun.jarsigner**
  This package comprises the interfaces and classes used to define the signing mechanism used by the jarsigner tool.
- **com.sun.java.accessibility.util**
  Provides a collection of interfaces and classes that compose the Java Accessibility Utilities.
- **com.sun.jdi**
  This is the core package of the Java Debug Interface (JDI), it defines mirrors for values, types, and the target VirtualMachine itself - as well bootstrapping facilities.
- **com.sun.jdi.connect**
  This package defines connections between the virtual machine using the JDI and the target virtual machine.
- **com.sun.jdi.connect.spi**
  This package comprises the interfaces and classes used to develop new TransportService implementations.
- **com.sun.jdi.event**
  This package defines JDI events and event processing.
- **com.sun.jdi.request**
  This package is used to request that a JDI event be sent under specified conditions.
- **com.sun.management**
  This package contains the JDK's extension to the standard implementation of the java.lang.management API and also defines the management interface for some other components of the platform.
- **com.sun.net.httpserver**
  Provides a simple high-level Http server API, which can be used to build embedded HTTP servers.
- **com.sun.net.httpserver.spi**
  Provides a pluggable service provider interface, which allows the HTTP server implementation to be replaced with other implementations.
- **com.sun.nio.sctp**
  A Java API for Stream Control Transport Protocol.
- **com.sun.security.auth**
  Provides implementations of Principal.
- **com.sun.security.auth.callback**

Provides an implementation of CallbackHandler.

- **com.sun.security.auth.login**

  Provides an implementation of Configuration.
- **com.sun.security.auth.module**

  Provides implementations of LoginModule.
- **com.sun.security.jgss**

  This package defines classes and interfaces for the JDK extensions to the GSS-API.
- **com.sun.source.doctree**

  Provides interfaces to represent documentation comments as abstract syntax trees (AST).
- **com.sun.source.tree**

  Provides interfaces to represent source code as abstract syntax trees (AST).
- **com.sun.source.util**

  Provides utilities for operations on abstract syntax trees (AST).
- **com.sun.tools.attach**

  Provides the API to attach to a Java™ virtual machine.
- **com.sun.tools.attach.spi**

  Only developers who are defining new attach providers should need to make direct use of this package.
- **com.sun.tools.javac**

  This package provides a legacy entry point for the javac tool.
- **com.sun.tools.jconsole**

  This package contains the JConsole API.
- **java.applet**

  Provides the classes necessary to create an applet and the classes an applet uses to communicate with its applet context.
- **java.awt**

  Contains all of the classes for creating user interfaces and for painting graphics and images.
- **java.awt.color**

  Provides classes for color spaces.
- **java.awt.datatransfer**

  Provides interfaces and classes for transferring data between and within applications.

- **java.awt.desktop**
  Provides interfaces and classes for interaction with various desktop capabilities.
- **java.awt.dnd**
  Drag and Drop is a direct manipulation gesture found in many Graphical User Interface systems that provides a mechanism to transfer information between two entities logically associated with presentation elements in the GUI.
- **java.awt.event**
  Provides interfaces and classes for dealing with different types of events fired by AWT components.
- **java.awt.font**
  Provides classes and interface relating to fonts.
- **java.awt.geom**
  Provides the Java 2D classes for defining and performing operations on objects related to two-dimensional geometry.
- **java.awt.im**
  Provides classes and interfaces for the input method framework.
- **java.awt.im.spi**
  Provides interfaces that enable the development of input methods that can be used with any Java runtime environment.
- **java.awt.image**
  Provides classes for creating and modifying images.
- **java.awt.image.renderable**
  Provides classes and interfaces for producing rendering-independent images.
- **java.awt.print**
  Provides classes and interfaces for a general printing API.
- **java.beans**
  Contains classes related to developing beans -- components based on the JavaBeans™ architecture.
- **java.beans.beancontext**
  Provides classes and interfaces relating to bean context.
- **java.io**
  Provides for system input and output through data streams, serialization and the file system.

- **java.lang**
  Provides classes that are fundamental to the design of the Java programming language.
- **java.lang.annotation**
  Provides library support for the Java programming language annotation facility.
- **java.lang.constant**
  Classes and interfaces to represent nominal descriptors for run-time entities such as classes or method handles, and classfile entities such as constant pool entries or invokedynamic call sites.
- **java.lang.instrument**
  Provides services that allow Java programming language agents to instrument programs running on the JVM.
- **java.lang.invoke**
  The java.lang.invoke package provides low-level primitives for interacting with the Java Virtual Machine.
- **java.lang.management**
  Provides the management interfaces for monitoring and management of the Java virtual machine and other components in the Java runtime.
- **java.lang.module**
  Classes to support module descriptors and creating configurations of modules by means of resolution and service binding.
- **java.lang.ref**
  Provides reference-object classes, which support a limited degree of interaction with the garbage collector.
- **java.lang.reflect**
  Provides classes and interfaces for obtaining reflective information about classes and objects.
- **java.math**
  Provides classes for performing arbitrary-precision integer arithmetic (BigInteger) and arbitrary-precision decimal arithmetic (BigDecimal).
- **java.net**
  Provides the classes for implementing networking applications.
- **java.net.http**
  HTTP Client and WebSocket APIs
- **java.net.spi**

Service-provider classes for the java.net package.

- **java.nio**

  Defines buffers, which are containers for data, and provides an overview of the other NIO packages.
- **java.nio.channels**

  Defines channels, which represent connections to entities that are capable of performing I/O operations, such as files and sockets; defines selectors, for multiplexed, non-blocking I/O operations.
- **java.nio.channels.spi**

  Service-provider classes for the java.nio.channels package.
- **java.nio.charset**

  Defines charsets, decoders, and encoders, for translating between bytes and Unicode characters.
- **java.nio.charset.spi**

  Service-provider classes for the java.nio.charset package.
- **java.nio.file**

  Defines interfaces and classes for the Java virtual machine to access files, file attributes, and file systems.
- **java.nio.file.attribute**

  Interfaces and classes providing access to file and file system attributes.
- **java.nio.file.spi**

  Service-provider classes for the java.nio.file package.
- **java.rmi**

  Provides the RMI package.
- **java.rmi.activation**

  Provides support for RMI Object Activation.
- **java.rmi.dgc**

  Provides classes and interface for RMI distributed garbage-collection (DGC).
- **java.rmi.registry**

  Provides a class and two interfaces for the RMI registry.
- **java.rmi.server**

  Provides classes and interfaces for supporting the server side of RMI.
- **java.security**

  Provides the classes and interfaces for the security framework.

- **java.security.acl**
  The classes and interfaces in this package have been deprecated.
- **java.security.cert**
  Provides classes and interfaces for parsing and managing certificates, certificate revocation lists (CRLs), and certification paths.
- **java.security.interfaces**
  Provides interfaces for generating RSA (Rivest, Shamir and Adleman AsymmetricCipher algorithm) keys as defined in the RSA Laboratory Technical Note PKCS#1, and DSA (Digital Signature Algorithm) keys as defined in NIST's FIPS-186.
- **java.security.spec**
  Provides classes and interfaces for key specifications and algorithm parameter specifications.
- **java.sql**
  Provides the API for accessing and processing data stored in a data source (usually a relational database) using the Java™ programming language.
- **java.text**
  Provides classes and interfaces for handling text, dates, numbers, and messages in a manner independent of natural languages.
- **java.text.spi**
  Service provider classes for the classes in the java.text package.
- **java.time**
  The main API for dates, times, instants, and durations.
- **java.time.chrono**
  Generic API for calendar systems other than the default ISO.
- **java.time.format**
  Provides classes to print and parse dates and times.
- **java.time.temporal**
  Access to date and time using fields and units, and date time adjusters.
- **java.time.zone**
  Support for time-zones and their rules.

- **java.util**

  Contains the collections framework, some internationalization support classes, a service loader, properties, random number generation, string parsing and scanning classes, base64 encoding and decoding, a bit array, and several miscellaneous utility classes.
- **java.util.concurrent**

  Utility classes commonly useful in concurrent programming.
- **java.util.concurrent.atomic**

  A small toolkit of classes that support lock-free thread-safe programming on single variables.
- **java.util.concurrent.locks**

  Interfaces and classes providing a framework for locking and waiting for conditions that is distinct from built-in synchronization and monitors.
- **java.util.function**

  Functional interfaces provide target types for lambda expressions and method references.
- **java.util.jar**

  Provides classes for reading and writing the JAR (Java ARchive) file format, which is based on the standard ZIP file format with an optional manifest file.
- **java.util.logging**

  Provides the classes and interfaces of the Java™ 2 platform's core logging facilities.
- **java.util.prefs**

  This package allows applications to store and retrieve user and system preference and configuration data.
- **java.util.regex**

  Classes for matching character sequences against patterns specified by regular expressions.
- **java.util.spi**

  Service provider classes for the classes in the java.util package.
- **java.util.stream**

  Classes to support functional-style operations on streams of elements, such as map-reduce transformations on collections.

- **java.util.zip**
  Provides classes for reading and writing the standard ZIP and GZIP file formats.
- **javax.accessibility**
  Defines a contract between user-interface components and an assistive technology that provides access to those components.
- **javax.annotation.processing**
  Facilities for declaring annotation processors and for allowing annotation processors to communicate with an annotation processing tool environment.
- **javax.crypto**
  Provides the classes and interfaces for cryptographic operations.
- **javax.crypto.interfaces**
  Provides interfaces for Diffie-Hellman keys as defined in RSA Laboratories' PKCS #3.
- **javax.crypto.spec**
  Provides classes and interfaces for key specifications and algorithm parameter specifications.
- **javax.imageio**
  The main package of the Java Image I/O API.
- **javax.imageio.event**
  A package of the Java Image I/O API dealing with synchronous notification of events during the reading and writing of images.
- **javax.imageio.metadata**
  A package of the Java Image I/O API dealing with reading and writing metadata.
- **javax.imageio.plugins.bmp**
  Package containing the public classes used by the built-in BMP plug-in.
- **javax.imageio.plugins.jpeg**
  Classes supporting the built-in JPEG plug-in.
- **javax.imageio.plugins.tiff**
  Public classes used by the built-in TIFF plug-ins.
- **javax.imageio.spi**
  A package of the Java Image I/O API containing the plug-in interfaces for readers, writers, transcoders, and streams, and a runtime registry.
- **javax.imageio.stream**
  A package of the Java Image I/O API dealing with low-level I/O from files and streams.

- **javax.lang.model**

  Classes and hierarchies of packages used to model the Java programming language.
- **javax.lang.model.element**

  Interfaces used to model elements of the Java programming language.
- **javax.lang.model.type**

  Interfaces used to model Java programming language types.
- **javax.lang.model.util**

  Utilities to assist in the processing of program elements and types.
- **javax.management**

  Provides the core classes for the Java Management Extensions.
- **javax.management.loading**

  Provides the classes which implement advanced dynamic loading.
- **javax.management.modelmbean**

  Provides the definition of the ModelMBean classes.
- **javax.management.monitor**

  Provides the definition of the monitor classes.
- **javax.management.openmbean**

  Provides the open data types and Open MBean descriptor classes.
- **javax.management.relation**

  Provides the definition of the Relation Service.
- **javax.management.remote**

  Interfaces for remote access to JMX MBean servers.
- **javax.management.remote.rmi**

  The RMI connector is a connector for the JMX Remote API that uses RMI to transmit client requests to a remote MBean server.
- **javax.management.timer**

  Provides the definition of the Timer MBean.
- **javax.naming**

  Provides the classes and interfaces for accessing naming services.
- **javax.naming.directory**

  Extends the javax.naming package to provide functionality for accessing directory services.

- **javax.naming.event**
  Provides support for event notification when accessing naming and directory services.
- **javax.naming.ldap**
  Provides support for LDAPv3 extended operations and controls.
- **javax.net**
  Provides classes for networking applications.
- **javax.net.ssl**
  Provides classes for the secure socket package.
- **javax.print**
  Provides the principal classes and interfaces for the Java™ Print Service API.
- **javax.print.attribute**
  Provides classes and interfaces that describe the types of Java™ Print Service attributes and how they can be collected into attribute sets.
- **javax.print.attribute.standard**
  Package javax.print.attribute.standard contains classes for specific printing attributes.
- **javax.print.event**
  Package javax.print.event contains event classes and listener interfaces.
- **javax.rmi.ssl**
  Provides implementations of RMIClientSocketFactory and RMIServerSocketFactory over the Secure Sockets Layer (SSL) or Transport Layer Security (TLS) protocols.
- **javax.script**
  The scripting API consists of interfaces and classes that define Java™ Scripting Engines and provides a framework for their use in Java applications.
- **javax.security.auth**
  This package provides a framework for authentication and authorization.
- **javax.security.auth.callback**
  This package provides the classes necessary for services to interact with applications in order to retrieve information (authentication data including usernames or passwords, for example) or to display information (error and warning messages, for example).
- **javax.security.auth.kerberos**
  This package contains utility classes related to the Kerberos network authentication protocol.
- **javax.security.auth.login**
  This package provides a pluggable authentication framework.

- **javax.security.auth.spi**

  This package provides the interface to be used for implementing pluggable authentication modules.
- **javax.security.auth.x500**

  This package contains the classes that should be used to store X500 Principal and X500 Private Credentials in a Subject.
- **javax.security.cert**

  Provides classes for public key certificates.
- **javax.security.sasl**

  Contains class and interfaces for supporting SASL.
- **javax.smartcardio**

  Java™ Smart Card I/O API.
- **javax.sound.midi**

  Provides interfaces and classes for I/O, sequencing, and synthesis of MIDI (Musical Instrument Digital Interface) data.
- **javax.sound.midi.spi**

  Supplies interfaces for service providers to implement when offering new MIDI devices, MIDI file readers and writers, or sound bank readers.
- **javax.sound.sampled**

  Provides interfaces and classes for capture, processing, and playback of sampled audio data.
- **javax.sound.sampled.spi**

  Supplies abstract classes for service providers to subclass when offering new audio devices, sound file readers and writers, or audio format converters.
- **javax.sql**

  Provides the API for server side data source access and processing from the Java™ programming language.
- **javax.sql.rowset**

  Standard interfaces and base classes for JDBC RowSet implementations.
- **javax.sql.rowset.serial**

  Provides utility classes to allow serializable mappings between SQL types and data types in the Java programming language.
- **javax.sql.rowset.spi**

The standard classes and interfaces that a third party vendor has to use in its implementation of a synchronization provider.

- **javax.swing**

  Provides a set of "lightweight" (all-Java language) components that, to the maximum degree possible, work the same on all platforms.
- **javax.swing.border**

  Provides classes and interface for drawing specialized borders around a Swing component.

  javax.swing.colorchooser

  Contains classes and interfaces used by the JColorChooser component.
- **javax.swing.event**

  Provides for events fired by Swing components.
- **javax.swing.filechooser**

  Contains classes and interfaces used by the JFileChooser component.
- **javax.swing.plaf**

  Provides one interface and many abstract classes that Swing uses to provide its pluggable look-and-feel capabilities.
- **javax.swing.plaf.basic**

  Provides user interface objects built according to the Basic look and feel.
- **javax.swing.plaf.metal**

  Provides user interface objects built according to the Java look and feel (once codenamed Metal), which is the default look and feel.
- **javax.swing.plaf.multi**

  Provides user interface objects that combine two or more look and feels.
- **javax.swing.plaf.nimbus**

  Provides user interface objects built according to the cross-platform Nimbus look and feel.
- **javax.swing.plaf.synth**

  Synth is a skinnable look and feel in which all painting is delegated.
- **javax.swing.table**

  Provides classes and interfaces for dealing with javax.swing.JTable.
- **javax.swing.text**

  Provides classes and interfaces that deal with editable and noneditable text components.
- **javax.swing.text.html**

Provides the class HTMLEditorKit and supporting classes for creating HTML text editors.

- **javax.swing.text.html.parser**
  Provides the default HTML parser, along with support classes.
- **javax.swing.text.rtf**
  Provides a class (RTFEditorKit) for creating Rich-Text-Format text editors.
- **javax.swing.tree**
  Provides classes and interfaces for dealing with javax.swing.JTree.
- **javax.swing.undo**
  Allows developers to provide support for undo/redo in applications such as text editors.
- **javax.tools**
  Provides interfaces for tools which can be invoked from a program, for example, compilers.
- **javax.transaction.xa**
  Provides the API that defines the contract between the transaction manager and the resource manager, which allows the transaction manager to enlist and delist resource objects (supplied by the resource manager driver) in JTA transactions.
- **javax.xml**
  Defines constants for XML processing.
- **javax.xml.catalog**
  Provides the classes for implementing XML Catalogs OASIS Standard V1.1, 7 October 2005.
- **javax.xml.crypto**
  Common classes for XML cryptography.
- **javax.xml.crypto.dom**
  DOM-specific classes for the javax.xml.crypto package.
- **javax.xml.crypto.dsig**
  Classes for generating and validating XML digital signatures.
- **javax.xml.crypto.dsig.dom**
  DOM-specific classes for the javax.xml.crypto.dsig package.
- **javax.xml.crypto.dsig.keyinfo**
  Classes for parsing and processing KeyInfo elements and structures.
- **javax.xml.crypto.dsig.spec**
  Parameter classes for XML digital signatures.

- **javax.xml.datatype**

  Defines XML/Java Type Mappings.
- **javax.xml.namespace**

  Defines XML Namespace processing.
- **javax.xml.parsers**

  Provides the classes for processing XML documents with a SAX (Simple API for XML) parser or a DOM (Document Object Model) Document builder.
- **javax.xml.stream**

  Defines interfaces and classes for the Streaming API for XML (StAX).
- **javax.xml.stream.events**

  Defines event interfaces for the Streaming API for XML (StAX).
- **javax.xml.stream.util**

  Provides utility classes for the Streaming API for XML (StAX).
- **javax.xml.transform**

  Defines the generic APIs for processing transformation instructions, and performing a transformation from source to result.
- **javax.xml.transform.dom**

  Provides DOM specific transformation classes.
- **javax.xml.transform.sax**

  Provides SAX specific transformation classes.
- **javax.xml.transform.stax**

  Provides StAX specific transformation classes.
- **javax.xml.transform.stream**

  Provides stream and URI specific transformation classes.
- **javax.xml.validation**

  Provides an API for validation of XML documents.
- **javax.xml.xpath**

  Provides an object-model neutral API for the evaluation of XPath expressions and access to the evaluation environment.
- **jdk.dynalink**

  Contains interfaces and classes that are used to link an invokedynamic call site.
- **jdk.dynalink.beans**

  Contains the linker for ordinary Java objects.
- **jdk.dynalink.linker**

Contains interfaces and classes needed by language runtimes to implement their own language-specific object models and type conversions.

- **jdk.dynalink.linker.support**

  Contains classes that make it more convenient for language runtimes to implement their own language-specific object models and type conversions by providing basic implementations of some classes as well as various utilities.
- **jdk.dynalink.support**

  Contains classes that make using Dynalink more convenient by providing basic implementations of some classes as well as various utilities.
- **jdk.javadoc.doclet**

  The Doclet API provides an environment which, in conjunction with the Language Model API and Compiler Tree API, allows clients to inspect the source-level structures of programs and libraries, including API comments embedded in the source.
- **jdk.jfr**

  This package provides classes to create events and control Flight Recorder.
- **jdk.jfr.consumer**

  This package contains classes for consuming Flight Recorder data.
- **jdk.jshell**

  Provides interfaces for creating tools, such as a Read-Eval-Print Loop (REPL), which interactively evaluate "snippets" of Java programming language code.
- **jdk.jshell.execution**

  Provides implementation support for building JShell execution engines.
- **jdk.jshell.spi**

  Defines the Service Provider Interface for pluggable JShell execution engines.
- **jdk.jshell.tool**

  Provides a mechanism to launch an instance of a Java™ shell tool.
- **jdk.management.jfr**

  This package contains classes to control and monitor Flight Recorder over Java Management Extensions (JMX).
- **jdk.nashorn.api.scripting**

  This package provides the javax.script integration, which is the preferred way to use Nashorn.
- **jdk.nashorn.api.tree**

Nashorn parser API provides interfaces to represent ECMAScript source code as abstract syntax trees (AST) and Parser to parse ECMAScript source scripts.

- **jdk.net**

  Platform specific socket options for the java.net and java.nio.channels socket classes.
- **jdk.nio**

  Defines JDK-specific channel APIs.
- **jdk.security.jarsigner**

  This package defines APIs for signing jar files.
- **netscape.javascript**

  Provides Java code the ability to access the JavaScript engine and the HTML DOM in the web browser.
- **org.ietf.jgss**

  This package presents a framework that allows application developers to make use of security services like authentication, data integrity and data confidentiality from a variety of underlying security mechanisms like Kerberos, using a unified API.
- **org.w3c.dom**

  Provides the interfaces for the Document Object Model (DOM).
- **org.w3c.dom.bootstrap**

  Provides a factory for obtaining instances of DOMImplementation.
- **org.w3c.dom.css**

  Provides interfaces for DOM Level 2 Style Specification.
- **org.w3c.dom.events**

  Provides interfaces for DOM Level 2 Events.
- **org.w3c.dom.html**

  Provides interfaces for DOM Level 2 HTML Specification.
- **org.w3c.dom.ls**

  Provides interfaces for DOM Level 3 Load and Save.
- **org.w3c.dom.ranges**

  Provides interfaces for DOM Level 2 Range.
- **org.w3c.dom.stylesheets**

  Provides interfaces for DOM Level 2 Style Specification.
- **org.w3c.dom.traversal**

  Provides interfaces for DOM Level 2 Traversal.
- **org.w3c.dom.views**

Provides interfaces for DOM Level 2 Views.

- **org.w3c.dom.xpath**

  Provides interfaces for DOM Level 3 XPath Specification.

- **org.xml.sax**

  Provides the core SAX APIs.

- **org.xml.sax.ext**

  Provides interfaces to SAX2 facilities that conformant SAX drivers won't necessarily support.

- **org.xml.sax.helpers**

  Provides helper classes, including support for bootstrapping SAX-based applications.