***Laboratory work #3***

**Topic :" Methodology of object-oriented modeling. Stage of creating a physical model"**

**1. Purpose of the work:**

Introduction to the basic elements of defining, presenting, designing, and modeling software systems using the UML language.

**Task:**

1. Review the material on object-oriented modeling (Appendix 1 (lab. rab. 2 and lecture materials))
2. Decide on diagrams from the family UML models at the stage of creating a physical model of an automated system (the topic was selected during laboratory work 1).
3. Develop UML diagrams of the physical model creation stage.
4. Document use cases.

TThe primary description should contain the following sections.

* 1. Short description.
  2. Participating entities.
  3. Prerequisites required for initiating a use case.
  4. A detailed description of the event stream, which includes: the main stream, which can be broken down to show subordinate event streams (subordinate streams can be further divided into even smaller streams, in order to make the document readability more convenient); alternative streams for identifying exceptional situations.
  5. Postconditions that determine the state of the system, after which the use case is completed (**Example in Appendix 1**).

1. Write at least 10 requirements according to the requirements syntax:

[circumstances] [subject] [action] [object] [restriction]

Example: When a signal is received [circumstances], the system [subject] must set the [action] signal discharge [object] for two seconds [limit].

1. Formulate non-functional requirements.
2. Develop a data processing algorithm (see the example of code design with explanations in the archive **pikpo3\_python.zip or pikpo3\_java.zip**).
   1. The processing algorithm should be implemented using an object-oriented approach. The data handler is implemented in a separate class (**DataProcessor**), which has 3 basic methods: read the data source (**read**), starting data processing (**run**), displaying the result on the screen (**print\_result**).
   2. Implement the necessary processing methods in the class (for example, cleaning, assigning categories, etc.).
   3. Use the "Factory" pattern to call different instances DataProcessor depending on the type of input data set (for example, csv-file, txt-file).
   4. Upload the code to **GitHub (see " Memo for working with GIT»)**.
   5. Develop UML class diagram the data handler.
3. **Write a report.** The report should include a set of static and dynamic models, a description of the data processing algorithm, screenshots with the results of the algorithm execution (a fragment of the input data set and output after processing), and a screenshot GitHub- a repository with uploaded project files.

**Additional information:**

**example**implementations of the data processing algorithm using**Factory Pattern**located in the archives**:**

**pikpo3\_python.zip –**example on **Python**

**pikpo3\_java.zip –**example on **Java (higher difficulty task)**

To run the example on Python you will need to install an additional package for working with dataset files:

**pip install pandas**

To run the example on Java you must install the package importer **Maven** and wednesday **Liberica** **OpenJDK 8**. The installer JDK you can download it from the link:

<https://bell-sw.com/pages/downloads/#mn>

During the installation process, don't forget to set the system variables **PATH** and **JAVA\_HOME**:

During the installation process, don't forget to set the system variables **PATH** and **JAVA\_HOME**:

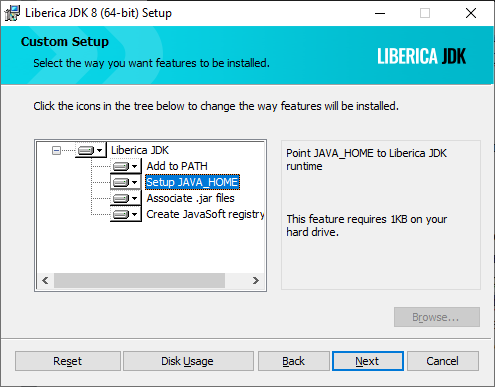


Figure 1-Setting OS Environment Variables

Before you start working with the project**in IDE don't forget to install the appropriate development environment Java (SDK)**in the project settings **(File -> Project Structure):**

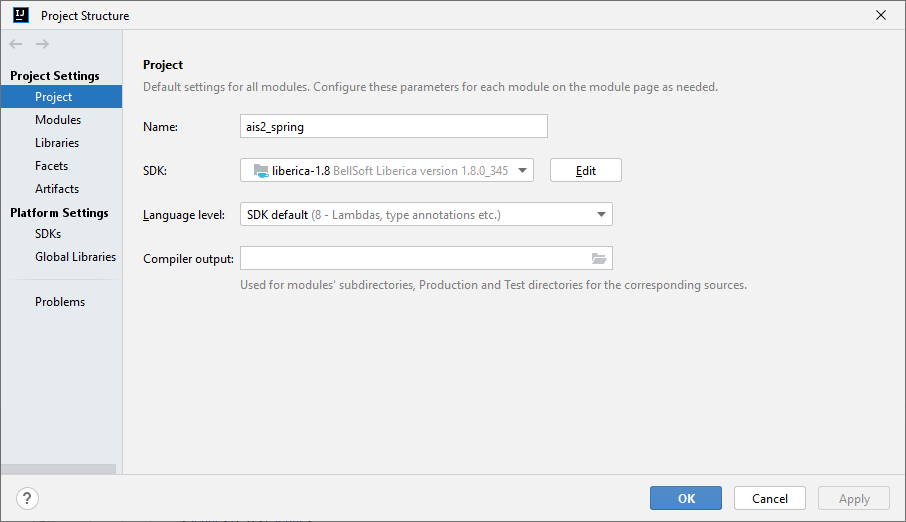


Figure 2 - Installing the Java SDK in the IDEA IDE

Next, you need to install the dependency collector **Maven**:

<https://maven.apache.org/install.html>

To set all the parameters specified in **pom.xml** dependencies run the commands **clean** and **install** in the dashboard Maven and update the project:

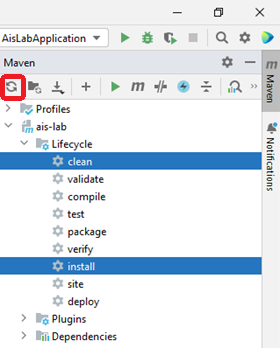


Figure 3-Panel Maven in IDEA IDE

Or enter the command in the terminal IDE:

**mvn clean install**

**Required software:**

https://www.jetbrains.com/ru-ru/pycharm/download **–** development environment on **Python** (PyCharm Community)

https://www.jetbrains.com/ru-ru/idea/download **–** development environment on **Java** (IDEA Community)

**Realization Factory Pattern:**

http://cpp-reference.ru/patterns/creational-patterns/factory-method/

https://refactoring.guru/ru/design-patterns/factory-method/python/example (**Python**)

https://refactoring.guru/ru/design-patterns/factory-method/java/example (**Java**)

**Tools for working with data files:**

[https://pandas.pydata.org/docs/reference/frame.html](https://pandas.pydata.org/docs/reference/frame.html%20) (DataFrame in**Python Pandas**)

<https://pandas.pydata.org/docs/reference/general_functions.html> (Basic methods for working with DataFrame in **Python Pandas**)

https://joinery.sh/v1.10/api/reference/joinery/DataFrame (**Java Joinery**)

***Report design requirements:***

The text execution method should be the same for all work. **Font**– **Times New Roman**, size 14, **line spacing**– 1,5, **field sizes**: left – 30 mm; right-10 mm, top-20 mm; bottom-20 mm. Abbreviations of words in the text are allowed only in general terms. **Paragraph indent (1,25)**it should be the same in all work. **Page numbering**the main text should be end-to-end. The page number is not indicated on the title page. The number itself is located at the bottom center of the page or on the right.

Appendix 1

|  |  |
| --- | --- |
| Use case | Conclusion of an agreement |
| Short description | This use case is necessary for registering a new subscriber in the network. |
| Subject | Operator, client. |
| Prerequisites | The operator needs to get acquainted with the existing telecom operators and issue a questionnaire form to a potential subscriber. |
| Main stream | After the customer selects the appropriate operator, they fill out the form, after which the operator checks the correctness of filling out the form on paper and enters the data into the system with the following action "Choosing a telecom operator-Communication services agreement".  After that, the form for signing a network subscriber in the system opens in the "Customer Service" system.  At the same time, the system first asks who will register: an individual or a Legal Entity, and only after that the corresponding registration form is displayed.  The operator enters information about the client organization, the contact person of the legal entity, and also enters the account numbers of the organization.  Then the agreement is saved. A communication session is made with the Server, during which this data is transmitted to the server. |
| Alternative flow | If the user has not entered all the fields, the system displays the message "Enter all fields" and allows you to go through the registration process again if an error occurs.  Also, the operator has the option of refusing to register a subscriber by selecting the appropriate command. |
| Postconditions | After successful completion of the use case, the client is entered in the Network Subscribers database on the server. |