Tema 1

1. Instalarea mediului de lucru Visual Paradigm Community Edition (Link: https://www.visual-paradigm.com/download/community.jsp).

   R-S: Done

2. Ce este UML și pentru ce este utilizat?

   R-S: UML, short for Unified Modeling Language, is a standardized modeling language consisting of an integrated set of diagrams, developed to help system and software developers for specifying, visualizing, constructing, and documenting the artifacts of software systems, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems. The UML is a very important part of developing object oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects. Using the UML helps project teams communicate, explore potential designs, and validate the architectural design of the software.

3. Ce sunt modelele și care este utilitatea lor?

   R-S: Models are abstract representations of a system. A model uses UML or other notation to describe a system at various levels of abstraction. Models often contain one or more diagrams that display an aspect of a model or a subset of the model's elements graphically. In this way, a diagram represents a certain aspect or part of a model. Diagrams can exist outside a model or within a model. When a diagram is deleted from a model, the elements remain part of the model.

   Models can be used to do the following things:
   - Visually represent a system that you want to build
   - Communicate your vision of a system to customers and colleagues
   - Develop and test the architecture of a system
   - Use the UML diagrams to direct code generation

In addition, models can also contain profiles which can be predefined or customized to a particular system or application.

4. Familiarizarea cu mediul de lucru mai sus amintit prin crearea anumitor diagrame din sectiunile 2.3.1 și 2.3.2 din [1] (class diagram – diagrama de clasă, use case diagram – diagrama de cazuri de utilizare, activity diagram – diagrama de activitate, sequence diagram – diagrama de secvente, state machine diagram – diagrama mașinii de stare).

   R-S: [Attachments]

5. Ce tipuri de diagrame sunt prezentate în [1] – Capitolul 2? Care sunt asemănările și deosebirile?

   R-S: There are two main types of diagrams presented: structure diagrams and behavior diagrams. The main difference between these categories is that structure diagrams doesn't consider the dynamic behavior of the elements in question, only the components of the system and the relations between them, while behavior diagrams enable you to define behavior in detail. Behavior refers to the direct consequences of an action of at least one object. It affects how the states of objects change over time, and the behavior can either be specified through the actions of a single object or result from interactions between multiple objects.

6. Caracterizați succint fiecare diagramele din [1] – Capitolul 2 (cele 5 menționate mai sus)

   R-S:

   1. Class diagrams originate from conceptual data modeling and object-oriented software development and are used to specify the data structures and object structures of a system. The class diagram is based primarily on the concepts of class, generalization, and association.

   2. Use case diagrams enable you to define the requirements that a system must fulfill. This diagram describes which users use which functionalities of the system but does not address specific details of the implementation. The units of functionality that the system provides for its users are called use cases.

   3. Activity diagrams allow to model processes of any kind both business processes and software processes. Activity diagrams offer control flow

   mechanisms as well as data flow mechanisms that coordinate the actions that make up an activity, that is, a process.

   4. Sequence diagrams describe the interactions between objects to fulfill a specific task, for example, registration for an exam in a university administration system. The focus is on the chronological order of the messages exchanged between the interaction partners. Various constructs for controlling the chronological order of the messages as well as concepts for modularization allow you to model complex interactions.
   5. State machine diagrams describe the life cycle of the objects from the system as they go through different states. This diagram describes the permissible behavior of an object in the form of possible states and state transitions triggered by various events.

7. Folosind Visual Paradigm, realizați diagramele de clasă:
   O_DiagramaDeClase.png
   O_DiagramaDeSecvente.png
   R-S: [Attachments]