**The name of the academic discipline:**

**“Object-oriented programming technologies and design standards”**

|  |  |
| --- | --- |
| **Specialty code and name** | 6-05-0612-01 Software Engineering |
| **Year of study** | 2 |
| **Semester of study** | 4 |
| **Number of in-class academic hours:** | 46 |
| **Lectures**  **Seminar classes**  **Practical classes**  **Laboratory classes** | 24 |
| - |
| - |
| 22 |
| **Form of the current assessment (*credit/ graded credit /exam*)** | credit |
| **Number of credit points** | 3 |
| **Competences** | Mastering the academic discipline “Object-oriented programming technologies and design standards” should provide the following competencies: use an object-oriented approach in the technology of developing software systems. |
| **Summary of the academic discipline:**  Module and object. Module, principles of modular programming. Encapsulation. Object as a dynamic module. Class as a module description. Exceptions. Errors and exceptions. Exception handling. Resource leak protection. Robust programming. Class and object. Object creation. Methods, constructors, destructors. Property. Virtual field (property). Access methods (get/set). Indexers. Class extension. Inheritance. Overriding class elements. Object compatibility, type checking. Virtual methods. Polymorphism. Abstract and dynamic methods. Virtual properties. Delegate and event. Delegate (method reference). Event as a list of delegates. Metaclass. Class methods. Virtual constructors. Interface. Reference counting mechanism. Globally unique identifier (GUID). Compatibility with classes. Template. Parameterized class. Parameter constraints. Attribute. Reflection. Custom attributes. Anonymous function. Delegate pattern. Constraints in expressions. OOP patterns. Iterator, Singleton, Deputy, Bridge, Observer, etc. Visual programming. Components. Program extensibility. OOP in Java. Module, interface, template, annotation. Differences from C++. | |