Національний технічний університет України
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Department
of Radio Engineering Systems

# [RE-61] OBJECT-ORIENTED PROGRAMMING TECHNOLOGIES FOR RADIO ENGINEERS

## Curriculum of the academic discipline (Syllabus)

### Course details

| | |
|---|---|
| Level of higher education | First (bachelor's) |
| Field of knowledge | G - Engineering, manufacturing, and construction |
| Specialty | G5 - Electronics, electronic communications, instrument engineering, and radio engineering |
| Educational program | All |
| Discipline status | Elective (F-catalog) |
| Form of higher education | Full-time |
| Year of training, semester | Available for selection starting from the 2nd year, spring semester |
| Scope of the discipline | 4 credits (Lectures 16 hours, Practical classes 30 hours, Laboratory work 30 hours, Independent work 74 hours) |
| Semester Control/control measures | Credit |
| Class schedule | https://schedule.kpi.ua |
| Language of instruction | Ukrainian / English |
| Information about the course coordinator / lecturers | Lecturer: Neuymin O. S., Lab: Neuymin O. S., Independent work: Neuymin O. S. |
| Course location | https://do.ipo.kpi.ua/course/view.php?id=2521 |

### Curriculum

**1. Description of the course, its purpose, subject matter, and learning outcomes Object-oriented programming in C++**

**C++** is based on the C language and currently supports various programming styles, including object- oriented style, which makes it easy to maintain large projects. This

language remains one of the fastest and is used in a variety of applications, from graphical user interfaces and 3D gaming graphics to robotics. C++ is used by all major companies, such as *Amazon, Google, Microsoft*, and in Ukraine: *GlobalLogic, Samsung, Luxoft, Infopulse*. The language is also actively developing, as every three years a new standard is adopted by a special C++ standardization committee, which adds new and new features to the language. C++ programmers are in demand all over the world, as evidenced by the number of open vacancies and salary levels.

***The aim of the course is to develop students' abilities to:***

- *apply the basics of object-oriented programming technology and basic design patterns when creating software for radio-technical information systems;*
- *create software with appropriate functionality for radio-electronic equipment;*
- *debug and implement programs in different programming environments;*
- *find, evaluate, and use information from various sources necessary for solving professional tasks;*
- *work in a team.*

***Students will gain knowledge of:***

- *the basics of object-oriented programming technology;*
- *basic design patterns;*
- *relationships between classes and the basics of UML (class and sequence diagrams);*
- *basic tools of the C++ language and the standard STL library;*
- *the Git version control system.*

***Students will be able to:***

- *apply an object-oriented approach when designing complex software systems with appropriate functionality;*
- *implement software taking into account requirements for its quality, reliability, and performance characteristics;*
- *master the tools of the C++ language;*
- *implement and debug programs in various programming environments.*

## 2. Prerequisites and post-requisites of the discipline (place in the structural-logical scheme of training under the relevant educational program)

The disciplines that provide "Object-Oriented Programming for Radio Engineers" are: "**Higher Mathematics**," "**General Physics**," "**Computer Science**." For successful mastery of the discipline, students must have a basic level of proficiency in the C programming language.

The discipline is the basis for studying all subsequent disciplines in which it is necessary to use the principles of object-oriented programming to write software, for example, a thesis project.

## 3. Contents of the discipline

***Main topics of the discipline***

1. *Working with memory. Interpreting memory.*
2. *Static local variables in functions. Pointers to functions.*
3. *Basic concepts of object-oriented programming: **encapsulation**, **polymorphism**, **inheritance**.*
4. *Classes. Constructors. Destructors. Initialization rules. Overloaded constructors. Copy constructor and assignment operator overload. The **this** pointer.*
5. *Pointers to class members and their use.*
6. *Overloading operators. Rules. Understanding situations when overloading is necessary. Diagnostics of operators that are executed during overloading.*

7. *Overloading of operators **+, =, [], ++, (), type casting, ->, new. delete.***
8. *placement new.*
9. *Friend functions, friend classes, class forward declaration. Overloading of "friend" operators.*
10. *Static class members (static member variables, member methods), **Singleton** pattern.*
11. *Using const and mutable modifiers. Removing constancy (mutable modifier, const_cast).*
12. *Dynamic data structures. Singly linked list. Doubly linked list. Queue. Circular queue. Priority queue. Binary tree.*
13. *Working with the C++ Standard Library (**STL**). Containers. Iterators.*
14. *Algorithms (rules of use; pros and cons) **big O notation.***
15. *Inheritance. Initialization rules for inheritance. Single inheritance, multiple inheritance, virtual inheritance.*
16. *Virtual functions. Early and late binding. The need for a virtual destructor.*
17. *Abstract classes and interfaces.*
18. *Design patterns. Aggregation and awareness. **UML** class diagrams, object interactions object interactions. Creational patterns. Structural patterns. Behavioral patterns.*
19. *Exception handling. try; catch; throw. Using throw to simplify logic and speed. Writing wrapper classes (smart pointers).*
20. ***Git** version control system.*

***Additional topics of the discipline***

1. *Working with files and directories in C++.*
2. *N amespace.*
3. *Templates. Function templates. Class templates (template specialization, template parameters, non-standard template parameters).*
4. *Boost library classes.*
5. *Dynamic DLL libraries.*
6. *Network applications. Blocking and non-blocking modes, protocols.*
7. *Working with databases, creating databases and tables.*
8. *Working with the Qt framework.*
9. *Functions with an indefinite number of parameters. File format.*

## 4. Teaching materials and

**resources Basic**

**literature:**


1. Vasilyev, O. Programming in C++ in examples and tasks: textbook / O. Vasilyev. — Kyiv: Lira-K, 2017. — 382 p.
2. Programming in C and C++: textbook / D.D. Tatarchuk, Yu.V. Didenko. — Kyiv: , 2012. — 112 p. [Electronic resource]. Access mode: https://ela.kpi.ua/bitstream/123456789/25787/1/NP_PM_C_ta_C%2B%2B.pdf
3. Eric Freeman. Head First. Design Patterns. Kharkiv: Fabula, 2020.
4. Neuymin, O. S. Object-Oriented Programming Technologies for Radio Engineers. Homework Assignment [Electronic resource]: a textbook for bachelor's degree students in the educational programs "Radio Engineering Computerized Systems,"
5. "Information and Communication Radio Engineering," "Intelligent Technologies of Radio Electronics" specialty 172 "Telecommunications and Radio Engineering" 172
6. "Electronic Communications and Radio Engineering" / O. S. Neumin, O. Yu. Myronchuk; Igor Sikorsky KPI
7. Sikorsky. – Electronic text data (1 file: 1 MB). – Kyiv: Igor Sikorsky KPI, 2023. – 17 p. – Title from the screen. https://ela.kpi.ua/handle/123456789/55285


**Additional literature:**


1. Stephen Prata. C++ Primer Plus. Addison-Wesley Professional; 6th edition. - 1440 P.
2. Scott Chacon, Ben Straub. Pro Git book. Link: https://git-scm.com/book/uk/v2
3. Nicolai M. Josuttis. The C++ Standard Library - A Tutorial and Reference, 2nd Edition. Addison Wesley Longman, 2012.
4. Anthony Williams. C++ Concurrency in Action: Practical Multithreading. Manning;

1st edition, 2012 – 528 P.

Some books can be found at the link - https://ua.booksee.org/
Telegram community on C++ at Igor Sikorsky Kyiv Polytechnic Institute - https://t.me/itkpi_cpp

## Educational content

### 5. Methodology for mastering the academic discipline (educational component)

- *Lectures are conducted according to the topics listed in section 3, "Course content."*
- *Practical classes are not provided for in the curriculum of the discipline.*
- *Seminar classes are not included in the curriculum.*
- *Laboratory work is carried out in GitHub Classroom or Moodle.*

*Laboratory work topics:*

1. *"Basics of using OOP in C++." The goal of the work is to familiarize students with the basic principles of object-oriented programming.*
2. *"Overloading operations."*
3. *"Application of inheritance and polymorphism."*
4. *"Dynamic data structures."*
5. *"Working with the standard C++ library (STL)"*
6. *"Handling Exceptions"*

LP No. 1. Array
implementation LR
No. 2. Stack
implementation
LR No. 3. Uploading a project to GitHub
LR No. 4. Overloading operators for the Array class LR No.
Bidirectional linked list

**Independent work by students**

***Title of the topic for independent study***

1. *Working with files and directories in C++.*
2. *N amespace.*
3. *Templates. Function templates. Class templates (template specialization, template parameters, non-standard template parameters).*
4. *Boost library classes.*
5. *Dynamic DLL libraries.*
6. *Network applications. Blocking and non-blocking modes, protocols.*
7. *Working with databases, creating databases and tables.*
8. *Working with the Qt framework.*
9. *Functions with an indefinite number of parameters. File format.*

***Homework***

*The purpose of the homework assignment is to deepen understanding of the theoretical course material and reinforce the skills of independently applying the acquired knowledge. The assignment covers all topics.*

*The task must be completed in accordance with one of the options. The option for each task is selected according to the student's number in the group journal.*

*When completing each task, you must first develop a class diagram using UML, and then write the*

*program code in C++ using a development environment such as Visual Studio.*

*Requirements for the final project*

- *The project must use at least one third-party library.*
- *The program consists of at least two classes.*
- *The project must have a UML diagram.*
- *The final result of the project must be posted on GitHub.*

*Example task*

Create classes with the specifications listed below. Define constructors and methods set.. (), get.. (), toString (). Set the data selection criteria and output this data to the console or GUI window. Each class that contains information must have several constructors declared.
The '" operator must be overloaded to output data to the console or GUI window. It must be possible to write the data of all objects to an encrypted file (the student chooses the encryption method) or database (DB).

**Student**: id, Last name, First name, Date of birth, Phone number, Faculty, Course, Group. Create an array of objects. Output: a) a list of students from a given faculty; b) lists of students for each faculty and course; c) a list of students born after a given year; d) a list of students in a study group.
*The time allocated for the completion of the DCR is at least two weeks.*

## Policy and control

### 1. Academic discipline (educational component) policy

- *Students are required to attend all lectures and laboratory classes.*
- *, students are awarded incentive points for their activity in classes .*
- *Students must complete their homework assignments by the date set by the instructor.* • *Penalty points will be applied to students for plagiarism.*
- *In case of failure to complete the curriculum and the presence of valid reasons for this, the student may be given an individual assignment.*

### 2. Types of control and rating system for assessing learning outcomes

*Ongoing assessment: quizzes on the topic of the lesson, MCW, test.*

*Calendar assessment: conducted twice per semester to monitor the current status of syllabus*

*requirements. Semester assessment:* credit/defense of the final course project

*Conditions for admission to semester assessment:*

- *minimum passing grade for individual assignments – 5 points;*
- *completion of all laboratory work;*
- *semester rating of more than 40 points.*

***Table of correspondence between rating points and university scale grades***

| Number of points | Grade |
|---|---|
| 100-95 | Excellent |
| 94 | Very good |
| 84 | Good |
| 74-65 | Satisfactory |

| | |
|---|---|
| 64-60 | Sufficient |
| Less than 60 | Unsatisfactory |
| Admission requirements not met | Not admitted |

**3. Additional information on the discipline (educational component)**

*Examples of questions that are included in the semester exam:*

1. Basic concepts of object-oriented programming: encapsulation.
2. Basic concepts of object-oriented programming inheritance.
3. Basic concepts of object-oriented programming: polymorphism.
4. Rule for initializing class members.
5. Why are methods and constructors needed?
6. Why do we need destructors?
7. Access specifiers.
8. What is operator overloading (examples: +, =, [], ++, (), type casting, ->).
9. What rules of operator overloading do you know?
10. Can a constructor be overloaded?
11. What is a copy constructor for?
12. Why do we need assignment operator overloading?
13. What are shallow and deep copying? The this pointer.
14. Features of static class members (static member variables, member methods).
15. How does the initialization of a descendant object occur?
16. What is the relationship between derived and base classes.
17. What is the protected specifier for?
18. Virtual functions, destruction.
19. What is an abstract base class?
20. What is an interface
21. What are the differences between a linked list and an array?
22. What STL containers are you familiar with?
23. What is an iterator
24. When is the Singleton design pattern needed?
25. Why do you need a virtual destructor?
26. For what purposes is the const keyword used?
27. How can you protect an object from being copied?

28. What is the difference between struct and class?
29. What is the difference between malloc and new?
30. What are the differences between delete and delete []?

*Possibility of crediting certificates of completion of distance or online courses on relevant topics;*

A student may be credited with a certificate of completion of online courses on the relevant subject only with the prior agreement of the instructor and approval of the online course on the subject of the C++ curriculum.
The grade is given based on the results of an interview with the student after completing the online course.

***Description of material, technical, and informational support for the discipline***

Classes are held on computers located in the RTS department's classroom. Students may use their own laptops.

The following software is used: **Microsoft Visual Studio, Qt Creator, Git, Conan.**

Link to the *Moodle* distance learning platform: https://do.ipo.kpi.ua/course/view.php?id=2521

---

Work program for the academic discipline (syllabus):
**Compiled by** Neuymin O. S.;
**Approved by** the RTS Department (Minutes No. 06/2025 dated 06/24/2025)
**Approved by** the methodological commission of the faculty/research institute (protocol No. 06/2025 dated 25.06.2025)