

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ  
ІМЕНІ СЕМЕНА КУЗНЕЦЯ

**ЗАТВЕРДЖЕНО**

на засіданні кафедри  
інформаційних систем.  
Протокол № 1 від 22.08.2023 р.

**ПОГОДЖЕНО**

Проректор з навчально-методичної роботи



Каріна НЕМАШКАЛО

**ПРОГРАМУВАННЯ КОМП'ЮТЕРНОЇ ГРАФІКИ**  
**робоча програма навчальної дисципліни (РПНД)**

Галузь знань 12 “Інформаційні технології”  
Спеціальність 121 “Інженерія програмного забезпечення”  
Освітній рівень перший (бакалаврський)  
Освітні програми “Інженерія програмного забезпечення”

Статус дисципліни вибіркова  
Мова викладання, навчання та оцінювання англійська

Розробник:  
к.т.н., доцент

Олег ФРОЛОВ

Завідувач кафедри  
інформаційних систем

Дмитро БОНДАРЕНКО

Гарант програми

Олег ФРОЛОВ

Харків  
2024



## INTRODUCTION

A significant part of the information that a user encounters has a graphic form. Modern operating systems are configured for the graphical interface, the work of system and application programs is visualized with graphical means, news information streams are saturated with graphical resources, graphical methods and tools are used in the computer game industry, graphics are widely used in scientific activities, engineering and design work, etc. The very term "computer graphics" today is interpreted as a type of activity in which computer equipment and software are used as tools for creating and editing images, for digitizing visual information about the real world for the purpose of its further processing and preservation.

Computer graphics is a field of knowledge in which, on the one hand, a significant amount of knowledge has been accumulated, on the other hand, methods, algorithms and practical applications are constantly being developed, it is a complex and diverse course. Computer graphics programming is an important component of the education of a modern programmer. In many cases, graphics needs can be met by various existing graphics libraries and systems. However, there is a constant need to create special graphic software tools. This can be done if you master the practical skills of solving typical computer graphics problems and the corresponding theoretical knowledge.

The course "Computer graphics programming" is studied by students of the "Software engineering" specialty of all forms of education in the third year during the fall semester.

The purpose of teaching this educational course is the formation of students' knowledge and skills in methods, algorithms and methods of working with flat and spatial objects in the creation of graphic software applications, with modern professional tools for working with computer graphics, with the practice of using computer libraries of computer graphics and visualization in modern programming languages.

Tasks of the course are:

- acquaint applicants with modern effective algorithms and methods of computer graphics;
- formation of competencies for designing and writing software for displaying graphic information and visualizing objects of various forms with flat and spatial placement and support for visual effects.

Object of the course is the formation of images of objects of various shapes on the computer.

subject the course includes algorithms and methods of computer graphics, hardware and software tools for their implementation.

In the process of training, students acquire the necessary knowledge during lectures and performing laboratory work. Independent work of students is also of great importance in the process of studying and consolidating knowledge. All types of classes are developed in accordance with the transfer system of the organization of the educational process.

The learning outcomes and competencies formed by the course are defined in the table. 1.

Table 1

**Learning outcomes and competences formed by the course**

<b>Learning outcomes</b>	<b>Competences that must be mastered by a student of higher education</b>
LO12	SC01, SC02, SC14
LO13	GC01, GC02, SC02, SC14
LO15	GC02, SC10, SC11, SC13

where, GC01. Ability to think abstractly, analyze and synthesize;

GC02. Ability to apply knowledge in practical situations;

SC01. Ability to identify, categorize and formulate software requirements;

SC02. Ability to participate in the design of software, including modelling (formal description) of its structure, behaviour and processes of operation;

SC10. The ability to accumulate, process, and systematize professional knowledge about creating and maintaining software and recognize the importance of life long learning;

SC11. Ability to implement phases and iterations of the life cycle of software systems and information technologies based on appropriate software development models and approaches;

SC13. Ability to reasonably choose and master tools for software development and maintenance;

SC14. Ability to think algorithmically and logically;

LO12. Apply effective software design approaches in practice;

LO13. Know and apply methods of developing algorithms, designing software and data and knowledge structures;

LO15. Motivated choice of programming languages and development technologies to solve developing and maintaining software the problems.

## **COURSE CONTENT**

### **Content module 1. Basics of computer graphics programming**

**Topic 1.** Introduction to computer graphics. Subject and field of application of computer graphics.

A brief history of the development of computer graphics. Technical means of computer graphics support: displays , input devices, video adapter, printers, scanners, plotters . Computer graphics software: device drivers, graphics program libraries, specialized graphics systems and program packages.

**Topic 2.** Graphical tools of programming languages.

**2.1.** Software color management. Graphical means of programming languages. Tools and methods. Screen coordinate systems. Display windows. Attributes and properties of a pencil/pen ( Pen ). Attributes and properties of the "brush" ( Brush ). Display functions of geometric primitives in OpenGL .

**2.2.** Raster conversion of graphic primitives. Algorithms of Bresenham bitmap. segment discretization. Bresenham algorithms for raster discretization of

circle and ellipse. Algorithms for filling internal areas.

**Topic 3.** Coordinate systems and geometric transformations in computer graphics problems.

**3.1.** Coordinate systems on the plane. Types of coordinate system. Formulas of mutual transition between polar and Cartesian coordinate systems. Geometric transformations of coordinates on the plane. Shift transformation (broadcast). Scaling conversion. Transform rotation (rotation) around the origin. Superposition of geometric transformations.

**3.2.** Geometric transformations on a plane in uniform coordinates. Concept of uniform coordinates. Elementary geometric transformations in homogeneous coordinates. Superposition of transformations in homogeneous coordinates. An example of using geometric transformations for animation programming.

**3.3.** Stereometric coordinate systems. Geometric transformations in space. Superposition of geometric transformations in 3D space. Stereometric geometric transformations in homogeneous coordinates. Content of uniform coordinates in space. Geometric transformations in space in uniform coordinates. Superposition of geometric transformations in space in homogeneous coordinates.

**3.4.** Geometric transformations in OpenGL.

**Topic 4.** Shaders : general concept, execution sequence.

GLSL. Syntax. Operators, declarations, specifiers (uniform, attribute, varying, const). Built-in functions of the GLSL OpenGL language . Vertex and fragment shaders : assignment, vertex attributes, input and output variables, uniform variables, varying variables. The basics of building and using shaders

## **Content module 2. Models and algorithms for representing objects and building images with them**

**Topic 5.** Images of 3D objects. Projections

**5.1.** Athene transformations in space. Projections. Design models. Classification of projections, orthographic, axonometric, oblique. Prospective projections. Methods of creating promising species.

**5.2.** Three-dimensional pipeline of observations. Reference system of observations. Conversion of external coordinates into observation coordinates. Design transformation. Field of view transformation and 3D screen coordinates. OpenGL three-dimensional observation functions.

**Topic 6.** Representation of geometric information.

**6.1 .** Representation and smoothing of curves. Basic concepts. Concept of parametric line. Cubic parametric lines. Polynomial interpolation. Smoothing splines. Spline curves. Bezier curves. B- spline curves

**6.2.** Spatial forms. Polyhedra. Curvilinear surfaces. Bilinear and linear surfaces. Bezier surfaces. B - c-plane surfaces.

**6.3.** Curves and surfaces in OpenGL.

**Topic 7.** Lighting and texturing models .

**7.1.** OpenGL lighting modeling. Diffuse and specular lighting. Coloring according to Gouro and Fong. Setting lighting parameters in OpenGL.

**7.2.** Texturing in OpenGL. Texture filtering: nearest pixel sampling, bilinear, trilinear, anisotropic. Texture and lighting. Automatic calculation of texture coordinates. Environment cards. Light maps ( lightmaps ). Multitexturing . Pixel operations: color mixing and translucent objects. Pixel operations: stencil buffer. Shadows and reflections.

**Topic 8.** Cutting geometric primitives.

Cutting ( clipping ) lines. Algorithm for dividing a segment in half. Sutherland — Cohen codes . Clipping polygons. Polygonal region hatching. Transition to 3D clipping with a visibility pyramid.

**Topic 9 .** Removal of hidden lines and surfaces

Historical tour. Iterative type methods. Z-buffer method. Methods of removing non-face faces of a polyhedron. Algorithms of Warnak and Weiler - Azerton. Methods of priorities (artist, floating horizon). The method of binary partitioning of space. Algorithms of sequential scanning for curved surfaces. Algorithm for determining visible surfaces by ray tracing.

**Topic 10.** Methods of creating realistic images

**10.1.** Direct and reverse ray tracing. Rays of wood. Shading of objects. Calculation of the intersection of the beam with the main geometric objects. Optimization of the ray tracing method. Lighting models in ray tracing.

**10.2.** Theoretical multiple operations (CSG). Procedural and noise textures. Emissivity: basic idea and system of linear equations. Emissivity: calculation of form factors.

The list of laboratory studies in the course is given in table 2.

Table 2

**The list of laboratory studies**

Name of the topic and/or task	Content
Topic 1. Task 1.	Raster conversion of basic graphic objects.
Topic 2. Task 2.	Methods of cutting segments. Windows and output areas.
Topic 3, Topic 4, Topic 5. Task 3.	Visualization of a three-dimensional object and its transformation.
Topic 6, Topic 7, Topic 8. Task 4.	Application of surface texturing using lighting
Topic 9, Topic 10. Task 5.	Visualization of the object model developed by means of a three-dimensional editor, construction of shadows from the object.

The list of self-studies in the course is given in table 3.

Table 3

**List of self-studies**

Name of the topic and / or task	Content
Topic 1 - 10	Studying lecture material

Topic 1 - 10	Preparation for laboratory classes
Topic 1 - 10	Preparation for the exam

The number of hours of lectures, laboratory classes, and hours of self-study is given in the technological card of the course.

## TEACHING METHODS

In the process of teaching the course, in order to acquire certain learning outcomes, to activate the educational process, it is envisaged to use such teaching methods as:

Problem lecture (Topic 1), verbal (lecture (Topic 2, 4, 5, 6, 7, 8, 9, 10)), lecture-dialogue (Topic 3).

In person (demonstration (Topic 1 - 10)).

Practical (laboratory work (Topic 1 - 10), case studies (Topic 4)).

## FORMS AND METHODS OF ASSESSMENT

The University uses a 100-point cumulative system for assessing the learning outcomes of students.

**Current control** is carried out during lectures, laboratory classes and is aimed at checking the level of readiness of the student to perform a specific job and is evaluated by the amount of points scored:

– for courses with a form of semester control as an exam: maximum amount is 60 points; minimum amount required is 35 points.

**The final control** includes current control and an exam.

**Semester control** is carried out in the form of a semester exam.

The maximum number of points that a student of higher education can receive during the exam is 40 points. The minimum amount for which the exam is considered passed is 25 points.

*The final grade in the course* is determined:

– for courses with a form of semester control as an exam: maximum amount is 60 points; minimum amount required is 35 points.

During the teaching of the course, the following control measures are used:

Current control: presentation of laboratory tasks (48 points), current control works (12 points).

Semester control: Grading including Exam (40 points)

More detailed information on the assessment system is provided in technological card of the course.

An example of an exam card and assessment criteria.

## An example of an exam card

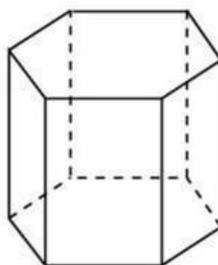
Semyon Kuznets Kharkiv National University of Economics  
First (bachelor) level of higher education  
Specialty "Software Engineering"  
Educational program "Software engineering"  
Course "Computer graphics programming"

### EXAM CARD No. 1

**Task 1.** The CDA algorithm for the raster representation of a straight line. The essence, an example of software implementation, advantages and disadvantages.

**Task 2.** Vertex shader - purpose, types of variables transferred. An example of a vertex shader in GLSL.

**Task 3.** Construct the central and parallel 3D projections of the proposed figure. Implement animation and one of the simpler lighting effects.



Protocol No. \_\_\_\_ dated " \_\_\_\_ " \_\_\_\_\_ 20\_\_ was approved at the meeting of the Department of Information Systems.

Examiner Ph.D. , Assoc. Frolov O. V.

Chief Department of Ph.D. , Assoc. Bondarenko D. O.

### Evaluation criteria

**The final marks for the exam** consist of the sum of the marks for the completion of all tasks, rounded to a whole number according to the rules of mathematics.

The first and second tasks of the examination ticket are evaluated for a maximum of 12 points each. The third task is worth 16 points. The number of points obtained from the answers to each question of the examination ticket is summed up. As a result of such a calculation, the applicant can receive from 0 to 40 points.

The evaluation of the exam result is formed according to the following rules:

1. Task 1 can be assigned from 0 to 12 points (for the presence of an array input algorithm - 5 points, for an example of a software implementation of the algorithm - 5 points, for wrapping the advantages and disadvantages of the algorithm - 2 points); task 2 can be awarded from 0 to 12 points (for the definition and description of GLSL language means - 6 points, for the given example - 6 points); task 3 can be awarded from 0 to 16 points (composing a geometric model of the object - 6 points, constructing a projection - 2 points, animating the object - 4 points, implementing lighting - 4 points).

2. 1 point is added for several solutions to one of the tasks.

3. 1 point is added for a reasoned choice of the option which is the optimal one from several solution options.

## RECOMMENDED LITERATURE

### Main

1. Журавчик Л.М. Програмування комп'ютерної графіки та мультимедійні засоби: навчальний посібник/ Л.М. Журавчик, О.М. Левченко. – Львів: Вид-во Львівської політехніки, 2019.-276 с.
2. Пічугін М.Ф. Комп'ютерна графіка: навч. посіб. / М.Ф. Пічугін, І.О. Канкін, В.В. Воротніков – К.: «Центр учбової літератури», 2019. – 346 с.
3. Смолій В.В. Навчальний посібник з дисципліни «Системи візуалізації та розпізнавання образів» [навчальний посібник] / В.В. Смолій, Я.А. Савицька, М.Д. Місюра, В.В. Шкарупило. - К.: ФОП Ямчинський О.В., 2020.- 200 с.
4. Комп'ютерна графіка : навчальний посібник : в 2-х кн.1. для здобувачів спеціальності 151 «Автоматизація та комп'ютерно- інтегровані технології» / Укладачі : О.В. Тотосько, П.Д. Стухляк, А.Г. Микитишин, В.В. Левицький, Р.З. Золотий. – Тернопіль : ТНТУ імені Івана Пулюя, 2023 – 304 с.
5. Інженерна і комп'ютерна графіка. Методичні рекомендації до самостійної роботи студентів спеціальності 186 "Видавництво та поліграфія" першого (бакалаврського) рівня [Електронний ресурс] / уклад. А. С. Гордєєв; Харківський національний економічний університет ім. С. Кузнеця. — Електрон. текстові дан. (107 КБ). — Харків : ХНЕУ ім. С. Кузнеця, 2022. — 23 с. - Режим доступу: <http://repository.hneu.edu.ua/handle/123456789/28149>

### Additional

6. Collomosse J.P. Fundamentals of Computer Graphics - CM20219 / John Collomosse. University of Bath, UK – 2019. – 100 p.
7. Stemkoski, L., & Pascale, M. (2021). Developing Graphics Frameworks with Python and OpenGL (1st ed.) / Lee Stemkoski, Michael Pascale. - CRC Press, 2021. - 344p., <https://doi.org/10.1201/9781003181378>.
8. Castorina M., Sassone G. Mastering Graphics Programming with Vulkan: Develop a modern rendering engine from first principles to state-of-the-art techniques / Marco Castorina, Gabriel Sassone. - Packt Publishing, 2023. – 382 p.
9. Marschner S., Shirley P. Fundamentals of Computer Graphics, 5th Edition / Steve Marschner, Peter Shirley - A K Peters/CRC Press, 2021. – 717 p.
10. Войтко Б. С. Використання матриць перетворень для побудови тривимірних об'єктів за допомогою OpenGL в комп'ютерній графіці / Б. С. Войтко, М. М. Марченко, П. В. Римар - Всеукраїнська науково-практична конференція для здобувачів, аспірантів та молодих вчених "Прикладні інформаційні технології". – Вінниця, 2020. – С. 167 – 170.

### Information resources

11. Computer Graphics with Modern OpenGL and C++ [Electronic resource]. – Access mode: <https://ua.udemy.com/course/graphics-with-modern-opengl/> .

12. Learn OpenGL with Python for Graphics and Games [Electronic resource].  
– Access mode: <https://ua.udemy.com/course/learn-opengl-with-python-for-graphics-and-games/> .

13. Learn the Vulkan API with C++ [Electronic resource]. – Access mode:  
<https://ua.udemy.com/course/learn-the-vulkan-api-with-cpp/> .