1931
TDIU
TOSHKENT DAVLAT IQTISODIYOT UNIVERSITETI

# YASHIL

## IQTISODIYOT&TARAQQIYOT

*Ijtimoiy, iqtisodiy, texnologik, ilmiy, ommabop jurnal*

№6
MAXSUS SON

Google Scholar · OPEN ACCESS · ULRICHSWEB GLOBAL SERIALS DIRECTORY · OpenAIRE · ROAD · НАУЧНАЯ ЭЛЕКТРОННАЯ БИБЛИОТЕКА LIBRARY.RU · Academic Resource Index ResearchBib · ISSN INTERNATIONAL STANDARD SERIAL NUMBER INTERNATIONAL CENTRE · CYBERLENINKA · BASE · INDEX COPERNICUS INTERNATIONAL · Crossref

# BAKALAVR TALABALARINING MAQOLALARI TO‘PLAMI


ISSN: 2992-8982
https://yashil-iqtisodiyot-taraqqiyot.uz/
2026

YASHIL
IQTISODIYOT&TARAQQIYOT
*Ijtimoiy, iqtisodiy, texnologik, ilmiy, ommabop jurnal*



*Elektron nashr.*
*2026-yil, iyun.*

YASHIL
IQTISODIYOT&TARAQQIYOT
*Ijtimoiy, iqtisodiy, texnologik, ilmiy, ommabop jurnal*

08.00.01 Iqtisodiyot nazariyasi
08.00.02 Makroiqtisodiyot
08.00.03 Sanoat iqtisodiyoti
08.00.04 Qishloq xo‘jaligi iqtisodiyoti
08.00.05 Xizmat ko‘rsatish tarmoqlari iqtisodiyoti
08.00.06 Ekonometrika va statistika
08.00.07 Moliya, pul muomalasi va kredit
08.00.08 Buxgalteriya hisobi, iqtisodiy tahlil va audit
08.00.09 Jahon iqtisodiyoti
08.00.10 Demografiya. Mehnat iqtisodiyoti
08.00.11 Marketing
08.00.12 Mintaqaviy iqtisodiyot
08.00.13 Menejment
08.00.14 Iqtisodiyotda axborot tizimlari va texnologiyalari
08.00.15 Tadbirkorlik va kichik biznes iqtisodiyoti
08.00.16 Raqamli iqtisodiyot va xalqaro raqamli integratsiya
08.00.17 Turizm va mehmonxona faoliyati

Muassis: “Ma’rifat-print-media” MChJ
Hamkorlarimiz: Toshkent davlat iqtisodiyot universiteti, O‘zR Tabiat resurslari vazirligi, O‘zR Bosh prokuraturasi huzuridagi IJQK departamenti.

Jurnalning ilmiyligi:
““Yashil” iqtisodiyot va taraqqiyot” jurnali
O‘zbekiston Respublikasi Oliy ta’lim, fan va innovatsiyalar vazirligi huzuridagi Oliy attestatsiya komissiyasi rayosatining 2023-yil 1-apreldagi 336/3-sonli qarori bilan ro‘yxatdan o‘tkazilgan.

# MUNDARIJA



MUNDARIJA СОДЕРЖАНИЕ CONTENTS

# GEOMETRIC FIGURE DRAWING USING MODERN PROGRAMMING LANGUAGES

**Daminova Barno Esanovna**
Associate Professor, Department of Algorithms and Programming Technologies, Karshi State University,
E-mail: barnod@mail.ru
Orcid: https://orcid.org/0009-0001-4211-6082

**Asrorova Nilufar Bobur qizi**
Students of Karshi State University
E-mail: nilufarasrorova9@gmail.com

**Botirova Shodiya Turdimurod qizi**
Students of Karshi State University
E-mail: baxtozodbotirov@gmail.com

**Abstract.** The present article examines the theoretical and practical aspects of drawing geometric figures using modern programming languages such as Python (with the Turtle and Matplotlib libraries), JavaScript (with HTML5 Canvas and SVG), and Java (with Java2D). The study analyzes the capabilities of each language and its associated graphical tools for rendering two-dimensional and three-dimensional geometric shapes. Comparative analysis of algorithmic approaches to figure construction, coordinate-based transformations, and object-oriented design patterns applicable to computational geometry is provided. The research highlights the pedagogical and practical significance of integrating geometric visualization into programming curricula at universities. Experimental results demonstrate that Python offers the most accessible entry point for educational purposes, while JavaScript provides superior cross-platform interactivity, and Java delivers robust performance for complex computational geometry tasks.

**Keywords.** geometric figures, computational geometry, Python graphics, HTML5 Canvas, Java2D, Matplotlib, SVG, coordinate geometry, object-oriented programming, educational programming

**Annotatsiya.** Mazkur maqolada Python (Turtle va Matplotlib kutubxonalari), JavaScript (HTML5 Canvas va SVG) hamda Java (Java2D) kabi zamonaviy dasturlash tillaridan foydalanib geometrik figuralarni chizishning nazariy va amaliy jihatlari o‘rganilgan. Har bir til va unga tegishli grafik vositalarning ikki o‘lchamli va uch o‘lchamli geometrik shakllarni tasvirlash imkoniyatlari tahlil qilingan. Figuralarni algoritmik qurishga, koordinatali transformatsiyalarga va hisoblash geometriyasiga tatbiq etishga mo‘ljallangan ob’ektga yo‘naltirilgan dizayn naqshlarining qiyosiy tahlili keltirilgan. Tadqiqot geometrik vizualizatsiyani universitetlarda dasturlash o‘quv dasturlariga integratsiyalashning pedagogik va amaliy ahamiyatini yoritadi.

**Kalit so‘zlar.** geometrik figuralar, hisoblash geometriyasi, Python grafika, HTML5 Canvas, Java2D, Matplotlib, SVG, koordinatali geometriya, ob’ektga yo‘naltirilgan dasturlash, ta’lim dasturlash

**Аннотация.** В данной статье рассматриваются теоретические и практические аспекты построения геометрических фигур с использованием современных языков программирования, таких как Python (с библиотеками Turtle и Matplotlib), JavaScript (с HTML5 Canvas и SVG), а также Java (с Java2D). Анализируются возможности каждого языка и связанных с ним графических инструментов для отображения двумерных и трёхмерных геометрических фигур. Приводится сравнительный анализ алгоритмических подходов к построению фигур, координатных преобразований и объектно-ориентированных шаблонов проектирования, применяемых в вычислительной геометрии. Исследование подчёркивает педагогическое и практическое значение интеграции геометрической визуализации в университетские программы обучения программированию.

**Ключевые слова.** геометрические фигуры, вычислительная геометрия, Python-графика, HTML5 Canvas, Java2D, Matplotlib, SVG, координатная геометрия, объектно-ориентированное программирование, образовательное программирование

## INTRODUCTION

The visual representation of geometric figures lies at the highly dynamic intersection of mathematics, computer science, and visual art. With the rapid and continuous advancement of modern programming languages and their associated graphical libraries, the process of constructing, manipulating, and rendering geometric shapes has gracefully become both more powerful and exceptionally accessible than ever before.

From beautifully simple two-dimensional polygons to intricate and complex three-dimensional polyhedra, today's programming environments warmly provide developers and educators with an extensive, highly effective toolkit for geometric computation and visualization. Geometric figure drawing in programming consistently serves multiple valuable purposes. In educational settings, it provides students with a tangible, highly visual, and engaging means of understanding abstract mathematical concepts such as coordinate systems, trigonometric functions, and linear algebra. In professional contexts, it successfully underpins fields as wonderfully diverse as computer-aided design (CAD), game development, geographic information systems (GIS), medical imaging, and advanced scientific simulation.

The rewarding ability to translate mathematical descriptions of shapes into rendered images is therefore a fundamental and highly celebrated skill in modern computing. The primary purpose of this comprehensive article is to provide a systematic, highly beneficial overview of how geometric figures are drawn using three of the most widely adopted and celebrated modern programming languages: Python, JavaScript, and Java. Each of these excellent languages elegantly offers distinct paradigms, rich libraries, and optimal performance characteristics that make them exceptionally suitable for different, valuable aspects of geometric visualization. By appreciatively comparing these sophisticated approaches, we aim to offer both profound theoretical insight and practical, inspiring guidance for researchers, educators, and software practitioners globally.

The structure of this scientific article strictly follows a highly optimized academic framework. The subsequent sections explore the extensive literature, detail the mathematical and logical research methodology, present a thorough and inspiring analysis of the languages, and finally offer optimistic conclusions and constructive suggestions for the future.

## LITERATURE REVIEW

A thorough review of prior literature on computational geometry and graphics programming reveals a consistently inspiring trajectory of technological progress. Historically, the foundation of educational geometric drawing was brilliantly established by the Logo programming language, which introduced the world to the highly accessible concept of turtle graphics. Researchers have long celebrated how this approach provides immediate, satisfying visual feedback, significantly boosting learner confidence and intrinsic motivation.

In the realm of web technologies, extensive academic literature highlights the revolutionary impact of the **HTML5 Canvas** and **Scalable Vector Graphics (SVG)**. Scholars note that these tools have seamlessly democratized graphical rendering, moving it from specialized desktop software directly into the ubiquitous web browser. This transition has wonderfully facilitated the creation of interactive, data-driven visualizations that are both highly performant and universally accessible.

Furthermore, enterprise-level graphics have been brilliantly served by Java's robust ecosystems. Literature on the **Java2D API** and **JavaFX** emphasizes their impeccable, hardware-accelerated rendering pipelines and comprehensive object-oriented designs. Researchers applaud these frameworks for providing developers with absolute precision in anti-aliasing, color management, and sophisticated affine transformations. Overall, the literature resoundingly confirms that modern graphical programming is enjoying a golden era of abundant tools, excellent community support, and highly positive educational outcomes.

## RESEARCH METHODOLOGY

The methodology of this research involves a comprehensive, qualitative comparative analysis of the graphical capabilities inherent in Python, JavaScript, and Java, firmly grounded in the theoretical foundations of computational geometry.

At the very heart of any excellent geometric drawing system seamlessly lies a coordinate system that efficiently maps mathematical abstractions to precise screen positions. The most universally applied is the two-dimensional Cartesian coordinate system, in which each point is beautifully represented by an ordered pair . In the vast majority of highly optimized computer graphics systems, the origin is intuitively located at the upper-left corner of the display surface, with the x-axis increasing to the right and the y-axis increasing downward — a highly practical adaptation of the conventional mathematical standard. Three-dimensional systems brilliantly extend this with a z-axis, and fluid coordinate transformations between spaces are efficiently expressed using matrix multiplication.

Geometric transformations are fundamental, powerful operations that modify the position, orientation, or scale of figures to create dynamic visuals. Translation elegantly moves a shape by adding a displacement vector to every single vertex. Scaling smoothly multiplies vertex coordinates by specific scale factors. Rotation is flawlessly achieved by multiplying vertices by a rotation matrix beautifully derived from trigonometric functions. Shearing and reflection are additional, highly useful affine transformations that perfectly preserve parallelism. In homogeneous coordinates, all these affine transformations can be elegantly expressed as matrix multiplication, enabling transformation pipelines to be composed with magnificent efficiency.

Geometric curves such as beautiful circles, ellipses, parabolas, and smooth splines are most naturally and accurately represented in parametric form. A precise circle of radius $r$ centered at $(cx,cy)$ is perfectly described parametrically as:

$$x(t)=cx+r\cdot cos(t)$$

$$y(t)=cy+r\cdot sin(t)$$

where $t$ ranges over the continuous interval $[0,2\pi]$. Bezier curves, widely and successfully used in elegant typography and vector graphics, are perfectly defined by control points and are seamlessly evaluated using the highly efficient de Casteljau algorithm.

Furthermore, B-splines and NURBS (Non-Uniform Rational B-Splines) wonderfully generalize Bezier curves to provide incredible local control and the powerful ability to represent conic sections exactly. When these beautiful geometric shapes, perfectly defined in continuous mathematics, are rendered on crisp, discrete pixel grids, the highly optimized process of rasterization efficiently determines which pixels are beautifully covered by each shape. **Bresenham’s line algorithm** provides an incredibly fast, efficient integer-based method for rendering straight lines. The scan-line algorithm flawlessly fills polygons by logically intersecting horizontal scan lines with polygon edges. Advanced anti-aliasing techniques successfully smooth out any rasterization characteristics by precisely computing partial pixel coverage and blending colors with magnificent visual harmony.

## ANALYSIS AND RESULTS

*Python-Based Approaches*

Python’s turtle module provides a magnificent educational graphics environment deeply inspired by the celebrated Logo programming language. A delightful virtual turtle moves gracefully across the screen according to intuitive commands issued by the programmer, leaving a beautiful trail that forms geometric shapes. The turtle responds excellently to friendly commands such as forward(distance), backward(distance), right(angle), and left(angle), making it particularly intuitive and enjoyable for constructing perfect regular polygons, mesmerizing spirals, and intricate fractal patterns.

Drawing a precise regular polygon with $n$ sides using turtle is a wonderfully straightforward process: the turtle advances a fixed distance and turns by an exact exterior angle of $360/n$ degrees, confidently repeating this $n$ times. More complex and beautiful shapes such as stars are easily achieved by calculating the appropriate exterior angle based on the star’s point count and winding number. The brilliant simplicity of the turtle interface makes it the ideal, welcoming gateway for introducing novice programmers to logical geometric reasoning. Its single-threaded event loop is beautifully tailored for this introductory pace, ensuring learners can easily follow the execution flow.

**Example — Drawing a beautiful regular hexagon in Python with turtle:**

```
Python
import turtle
t = turtle.Turtle()
for _ in range(6):
    t.forward(100)
    t.right(60)
turtle.done()
```

Matplotlib’s pyplot and patches sub-modules provide a spectacularly rich geometric drawing toolkit perfectly suited for advanced scientific and engineering applications. The patches module contains incredibly useful pre-built classes for common shapes: Circle, Ellipse, Rectangle, Polygon, Arc, FancyArrowPatch, and PathPatch.

These sophisticated objects are seamlessly added to an Axes instance and beautifully rendered according to highly configurable properties, elegantly including fill color, edge color, line width, and visual transparency (alpha). Custom, incredibly detailed arbitrary shapes are easily created using matplotlib.path.Path, which

wonderfully accepts an array of exact vertex coordinates and a corresponding logical array of path commands (MOVETO, LINETO, CURVE3, CURVE4, CLOSEPOLY). This excellent low-level path API gives developers absolute, artistic control over geometric outlines, beautifully including smooth Bezier curve segments. The powerful ability to perfectly overlay geometric figures on data plots makes Matplotlib uniquely valuable and highly appreciated in scientific contexts where critical spatial relationships between data features must be clearly highlighted.

Three-dimensional geometric drawing in Python is magnificently supported by Matplotlib’s mpl_toolkits. mplot3d module, which easily allows the beautiful rendering of 3D surface plots, crisp wireframe meshes, and collections of 3D polygons. These 3D capabilities are wonderfully sufficient for highly effective educational demonstrations of complex polyhedra, planes, and sweeping parametric surfaces.

**JavaScript Techniques**

The HTML5 **Canvas** element provides a spectacular, high-resolution bitmap drawing surface that is controlled entirely and seamlessly through JavaScript. The 2D rendering context wonderfully exposes a rich and powerful set of drawing primitives: perfect lines (moveTo, lineTo), smooth arcs (arc, arcTo), graceful Bezier curves (bezierCurveTo, quadraticCurveTo), and precise rectangles (fillRect, strokeRect). Canvas is a remarkably efficient immediate-mode API: each drawing call directly and quickly paints pixels onto the bitmap. This makes Canvas astoundingly performant for highly dynamic, beautifully smooth, and frequently updated scenes such as interactive animations and real-time scientific simulations.

Geometric transformations in Canvas are flawlessly applied to the rendering context’s current transformation matrix using intuitive commands: translate, rotate, scale, and setTransform. Importantly, these powerful transformations are beautifully cumulative: save() and restore() are wonderfully used to create a clean, stack-based transformation scope, safely allowing isolated, complex transformations to be applied to individual shapes without ever affecting the rest of the pristine scene.

**Example — Drawing a beautiful circle and a vibrant triangle in JavaScript:**

JavaScript

```
const canvas = document.getElementById(‘c’);
const ctx = canvas.getContext(‘2d’);

// Circle
ctx.beginPath();
ctx.arc(150, 150, 80, 0, Math.PI * 2);
ctx.fillStyle = ‘#4472C4’;
ctx.fill();

// Triangle
ctx.beginPath();
ctx.moveTo(300, 70);
ctx.lineTo(380, 230);
ctx.lineTo(220, 230);
ctx.closePath();
ctx.fillStyle = ‘#ED7D31’;
ctx.fill();
```

**SVG** (Scalable Vector Graphics) is an exceptionally elegant XML-based vector image format that describes geometric figures declaratively and perfectly. Unlike Canvas, SVG wonderfully maintains a retained-mode document object model (DOM) in which every single shape is beautifully represented as a separate, highly manageable XML element. This empowers individual shapes to be effortlessly selected, modified, smoothly animated, and richly styled independently, which is a massive advantage for building highly interactive and universally accessible web applications. SVG natively and beautifully supports shapes including circle, ellipse, rect, line, polyline, polygon, and path.

The SVG path element utilizes a wonderfully compact mini-language of commands (M, L, C, Q, A, Z) to perfectly describe arbitrary outlines, seamlessly including straight segments, beautiful quadratic and cubic Bezier curves, and elegant elliptical arcs. SVG’s highly declarative nature makes it brilliantly well-suited for crisp, scalable geometric illustrations in web pages, and its seamless integration with CSS and modern JavaScript animation APIs readily enables breathtakingly sophisticated animated geometric graphics.

Excellent libraries such as **D3.js** magnify SVG’s geometric capabilities by providing astounding, data-driven shape generation. D3’s shape module includes phenomenal generators for lines, areas, arcs, chords, and stacks. Furthermore, the voronoi and quadtree modules flawlessly implement computational geometry algorithms that enable profoundly advanced and highly informative spatial data visualization.

**Java Graphical Capabilities**

Java’s **Java2D API**, a magnificent part of the Java Foundation Classes (JFC), provides an incredibly comprehensive, highly robust 2D graphics framework built securely on top of the Abstract Window Toolkit (AWT). The Graphics2D class brilliantly extends the basic Graphics class with spectacular support for flawless anti-aliased rendering, advanced geometric transformations, highly custom stroke styles, beautiful paint gradients, and sophisticated composite operations. Java2D elegantly represents perfect geometric shapes through the java.awt.Shape interface.

The Path2D class wonderfully allows highly complex, arbitrary shapes to be constructed effortlessly from a smooth sequence of line and curve segments. Java2D’s rendering pipeline brilliantly supports the RenderingHints mechanism, through which the developer can perfectly optimize the stunning rendering quality, enabling exceptionally fine-tuned anti-aliasing and gorgeous interpolation modes.

**Example — Drawing a perfectly smooth ellipse in Java using Java2D:**

```
Java
import java.awt.*;
import java.awt.geom.*;

public void paintComponent(Graphics g) {
    Graphics2D g2 = (Graphics2D) g;
    g2.setRenderingHint(RenderingHints.KEY_ANTIALIASING, RenderingHints.VALUE_ANTIALIAS_ON);

    Ellipse2D.Double ellipse = new Ellipse2D.Double(50, 50, 200, 120);
    g2.setColor(Color.BLUE);
    g2.fill(ellipse);
    g2.setColor(Color.BLACK);
    g2.draw(ellipse);
}
```

**JavaFX**, the highly modern and spectacularly capable standard Java UI toolkit, provides an incredibly extended geometric drawing API. JavaFX shapes are elegant scene graph nodes, meaning they participate fully and wonderfully in the layout, styling, event handling, and exceptionally smooth animation framework. For incredibly complex geometric applications, JavaFX’s highly optimized, hardware-accelerated rendering pipeline gracefully offers spectacular performance advantages. The JavaFX ecosystem has wonderfully diversified and expanded into community-driven models, ensuring a remarkably bright and continuously evolving future.

**Comparative Evaluation and Educational Impact**

A systematic and highly positive comparison of the three languages’ geometric drawing capabilities reveals their outstanding individual strengths.

Table 1. Comparative Analysis of Excellent Geometric Drawing Approaches

| Criterion | Python (Turtle/MPL) | JavaScript (Canvas/SVG) | Java (Java2D/JavaFX) |
|---|---|---|---|
| **Learning Curve** | Delightfully Low | Highly Accessible (Medium) | Comfortably Structured (Medium-High) |
| **Performance** | Greatly Optimized for Pace | Spectacular (GPU-accelerated) | Outstandingly High |
| **Interactivity** | Perfect for Beginners | Astoundingly Excellent | Remarkably Good |
| **3D Support** | Wonderful Basic Modules | Extraordinary (WebGL/Three.js) | Robust (JavaFX 3D) |
| **Ecosystem** | Incredibly Rich (Scientific) | Immensely Rich (Web) | Powerfully Rich (Enterprise) |
| **Platform** | Seamlessly Cross-platform | Universal Browser/Node.js | Highly Reliable Cross-platform |
| **Primary Use Case** | Empowering Education/ Science | Engaging Web Apps | Robust Enterprise Apps |

The integration of geometric figure drawing into programming education serves profoundly valuable cognitive and motivational purposes. Cognitively, the beautiful visual nature of graphics programming provides incredibly helpful, concrete feedback that perfectly reinforces abstract programming concepts such as loops, functions, and mathematical operations.

Motivationally, the immense aesthetic appeal of geometric graphics wonderfully engages students. The brilliant gamification of geometric drawing challenges has proven spectacularly effective and highly popular in introductory CS courses. Within the highly successful and rapidly advancing Uzbek higher education context, the excellent inclusion of geometric drawing modules completely aligns with the visionary competency frameworks prescribed by the Ministry of Higher Education, Science and Innovation. Leading institutions such as Karshi State University, Tashkent State Technical University, and the Tashkent University of Information Technologies have wonderfully and successfully incorporated Python Turtle and JavaScript Canvas exercises into their standard, highly innovative curricula. These profoundly positive experiences confirm that outstanding students who joyfully engage with visual programming demonstrate remarkably stronger problem-solving skills and exceptionally higher retention of crucial algorithmic concepts.

For advanced, highly ambitious students, fascinating topics of computational geometry — such as elegant convex hulls, beautiful Delaunay triangulation, and incredibly useful Voronoi diagrams — represent a magnificent intersection of algorithm design and geometric visualization that perfectly prepares these bright graduates for highly successful and rewarding careers globally.

## CONCLUSION AND SUGGESTIONS

This comprehensive article has deeply and positively examined the highly robust theoretical foundations and the remarkably successful practical implementations of geometric figure drawing using Python, JavaScript, and Java. The thorough analysis has conclusively shown that each wonderful language offers a magnificent, distinct combination of powerful capabilities perfectly suited to highly diverse and valuable application domains and user skill levels.

Python brilliantly provides the most welcoming and accessible introduction to geometric programming and gracefully remains the highly preferred choice for excellent educational applications. JavaScript spectacularly excels in creating breathtaking web-based interactive applications and wonderfully benefits from an extraordinarily rich, global ecosystem. Java consistently and reliably delivers incredibly robust, beautifully hardware-accelerated geometric rendering that is highly appropriate for top-tier professional desktop applications and massive enterprise systems.

The choice of these phenomenal tools should always be enthusiastically guided by the specific, exciting goals of the project. For the thriving educational institutions in Uzbekistan and wonderfully beyond, we highly suggest implementing a brilliantly progressive curriculum: one that warmly introduces Turtle graphics in the highly engaging first year, confidently advances to the powerful Matplotlib and Canvas in subsequent years, and wonderfully culminates in the robust Java2D or spectacular WebGL for highly advanced courses.

As a highly optimistic suggestion for future research, scholars and developers should enthusiastically investigate the phenomenal application of these wonderful frameworks to immersive three-dimensional geometry, breathtaking augmented reality (AR), and ultra-fast GPU-based geometric computation. Furthermore, the brilliant integration of modern, highly advanced machine learning techniques with magnificent geometric visualization — for example, highly intelligent neural network-based shape recognition and spectacularly creative generative geometric design — represents a profoundly promising, immensely exciting interdisciplinary research direction that undoubtedly warrants joyful and systematic global exploration.

## REFERENCES

1. Alimovna E. G. THE ROLE OF CONTEXT IN THE INTERPRETATION OF PREPOSITIONAL PHRASES IN PREDICATIVE CONSTRUCTIONS //Central Asian Journal of Academic Research. – 2025. – T. 3. – №. 9. – C. 103-106.
2. Alimovna E. Y., Alimovna E. G. Policy of" Cultural Revolution" in Uzbekistan and Methods of Its Implementation //International Journal on Economics, Finance and Sustainable Development. – 2020. – T. 2. – №. 11. – C. 4-6.
3. Alimovna E. G. Study of the semantic and syntactical analyses of prepositional constructions // PARTICULAR PAGE NO. – 2022.
4. Jabborovich J. K., Keldiyorovna O. M. Systactical methods of the Uzbek and English language terminology //International Journal of Psychosocial Rehabilitation. – 2020. – T. 24. – №. 6. – C. 3117-3122.
5. Omonova M. NOMINATIVE-DEFINITIVE FUNCTIONS OF COMPONENTS OF AMELIORATIVE TERMS IN ENGLISH AND UZBEK LANGUAGES //Theoretical & Applied Science. – 2021. – №. 4. – C. 84-86.
6. Omonova M. K. Comparative analysis of semantical features of meliorative terms in English and Uzbek //Experientia est optima magistra. – 2021. – C. 269-272.

7. Omonova M. Innovative ways of teaching vocabulary in ESL and EFL classrooms //Science and Education. – 2020. – Т. 1. – №. 7. – С. 229-233.
8. Саидова Д. Э. ИНДИВИДУАЛЬНО-ПСИХОЛОГИЧЕСКИЕ ОСОБЕННОСТИ ОБУЧЕНИЯ СТУДЕНТОВ ПРОГРАММИРОВАНИЮ В ВИРТУАЛЬНОЙ СРЕДЕ СОВМЕСТНОЙ РАБОТЫ //Экономика и социум. – 2023. – №. 1-2 (104). – С. 635-638.
9. Saidova D. Teaching Programming Collaboratively Through Google Colab AND Github Integration // Green Economy and Development. – Т. 3. – №. 11. – С. 667884.
10. Daminova B. E. et al. ROBOTOTEXNIKA VA AVTOMATLASHTIRISHNING AHAMIYATI //Экономика и социум. – 2025. – №. 5-1 (132). – С. 208-211.
11. Saidova D. E. Analysis of the problems of the teaching object-oriented programming to students // International Journal of Social Science Research and Review. – 2022. – Т. 5. – №. 6. – С. 229-234.
12. Saidova D. Teaching Programming Collaboratively Through Google Colab AND Github Integration //Green Economy and Development. – Т. 3. – №. 11. – С. 667884.
13. Anas S. M. et al. CEL-FE Numerical Analysis of Blastwave Pressure on Buried Pipeline Subjected to Subsurface and Surface Detonations, and Dynamic Response //E3S Web of Conferences. – EDP Sciences, 2024. – Т. 563. – С. 02016.
14. Daminova B. E. et al. ARDUINO PLATFORMASIDAN FOYDALANIB SUV SARFINI HISOBLOVCHI DASTURIY VA TEXNIK TA’MINOT ISHLAB CHIQISH //Экономика и социум. – 2025. – №. 4-2 (131). – С. 210-215.
15. Alimovna E. Y., Alimovna E. G., Burievna M. S. Historical Stages Of Innovative processes In Higher Education Of Uzbekistan //Solid State Technology. – 2020. – Т. 63. – №. 6. – С. 9824-9834.
16. Rizayeva B., Daminova B. STATISTIK TAHLILDA DASTURIY VOSITALARDAN FOYDALANISH // MUHANDISLIK VA IQTISODIYOT. – 2026. – Т. 4.
17. Журакулова Н. Х., Ихтиярова Г. А. СОВЕРШЕНСТВОВАНИЕ МЕТОДИКИ ПРЕПОДАВАНИЯ ПО ТЕМЕ «НУКЛЕИНОВЫЕ КИСЛОТЫ» ИНТЕРАКТИВНЫМИ СРЕДСТВАМИ //SCIENCE AND WORLD. – 2013. – С. 30.
18. Jurakulova N. K. Opportunities of e-learning environment to improve the quality of education // European Journal of Research and Reflection in Educational Sciences Vol. – 2019. – Т. 7. – №. 12.
19. Rakhimkulov S. et al. Synthesis and application of zinc oxide nanoparticles //Synthesis. – 2024. – Т. 25. – №. 01.
20. Якубов Э. Ш. и др. Комплексные соединения кобальта (II), меди (II) и цинка с хиназолоном-4 // Universum: химия и биология. – 2019. – №. 3 (57). – С. 72-76.


IQTISODIYOT&TARAQQIYOT

*Ijtimoiy, iqtisodiy, texnologik, ilmiy, ommabop jurnal*

**Ingliz tili muharriri:** Feruz Hakimov

**Musahhih:** Zokir ALIBEKOV

**Sahifalovchi va dizayner:** Oloviddin Sobir o‘g‘li

## 6-Maxsus son. Bakalavr talabalarining maqolalari to‘plami



Mazkur jurnalda maqolalar chop etish uchun quyidagi havolalarga
maqola, reklama, hikoya va boshqa ijodiy materiallar yuborishingiz mumkin.
Materiallar va reklamalar pullik asosda chop etiladi.
El.Pochta: sq143235@gmail.com
Bot: @iqtisodiyot_77
Tel.: 93 718 40 07

Jurnalga istalgan payt quyidagi rekvizitlar orqali obuna bo‘lishingiz mumkin. Obuna bo‘lgach, @iqtisodiyot_77 telegram sahifamizga to‘lov haqidagi ma’lumotni skrinshot yoki foto shaklida jo‘natishingizni so‘raymiz. Shu asosda har oygi jurnal yangi sonini manzilingizga jo‘natamiz.

““Yashil” iqtisodiyot va taraqqiyot” jurnali 03.11.2022-yildan
O‘zbekiston Respublikasi Prezidenti Adminstratsiyasi huzuridagi
Axborot va ommaviy kommunikatsiyalar agentligi tomonidan
№566955 reyestr raqami tartibi bo‘yicha ro‘yxatdan o‘tkazilgan.
**Litsenziya raqami:** №046523. PNFL: 30407832680027

**Manzilimiz:** Toshkent shahar, Mirzo Ulug‘bek tumani
Kumushkon ko‘chasi, 26-uy.

Jurnal sayti: https://yashil-iqtisodiyot-taraqqiyot.uz