Acta et Commentationes, Sciences of Education, nr. 1(35), 2024 ISSN 1857-0623
p. 35-46 E-ISSN 2587-3636

CZU: 373:378:004 DOI: 10.36120/2587-3636.v35i1.35-46

UN MODEL DE PROIECT STEAM ÎN CADRUL DISCIPLINEI PROGRAMAREA ORIENTATĂ PE OBIECTE

Ala GASNAŞ, doctor, conferențiar universitar
https://orcid.org/0000-0002-7174-7027
Universitatea Pedagogică de Stat ”Ion Creangă” din Chișinău

Rezumat. Abordarea STEM (Știință, Tehnologie, Inginerie și Matematică) este un domeniu în continuă expansiune, datorită beneficiilor sale evidente în lumea contemporană dominată de globalizare și tehnologizare. Știința, tehnologia și matematica constituie baza pentru progresele tehnologice din întreaga lume, iar acest lucru evidențiază necesitatea motivării studenților pentru a se angaja în învățarea STEM/STEAM. În acest articol, vom explora relevanța educației STEM/STEAM, beneficiile acesteia și vom prezenta un model de proiect în cadrul disciplinei POO (Programare Orientată pe Obiecte) care poate fi implementat în sălile de clasă pentru a pregăti viitorii inovatori.
Cuvinte cheie: educație STEM, educație STEAM, activități STEM

A STEAM PROJECT MODEL WITHIN THE OBJECT-ORIENTED PROGRAMMING DISCIPLINE

Abstract. The STEM (Science, Technology, Engineering, and Mathematics) approach is a continuously expanding field, thanks to its evident benefits in the contemporary world dominated by globalization and technological advancements. Science, technology, and mathematics form the foundation for technological progress worldwide, underscoring the necessity of motivating students to engage in STEM/STEAM learning. In this article, we will explore the relevance of STEM/STEAM education, its benefits, and present a project model within the Object-Oriented Programming (OOP) discipline that can be implemented in classrooms to prepare future innovators.
Keywords: STEM education, STEAM education, STEM activities.

Introducere

În contextul evoluției rapide din lumea contemporană, domeniile științei, tehnologiei, ingineriei și matematicii (STEM) joacă un rol esențial în definirea viitorului societății noastre. Educația STEM oferă studenților abilitățile și cunoștințele necesare pentru a prospera într-o lume condusă de tehnologie și inovație.

STEM este un concept educațional care se adresează tuturor nivelurilor de elevi, începând cu învățământul preșcolar până la universitate, care include predarea disciplinelor știință, tehnologie, inginerie și matematică într-o manieră integrată. Această abordare educațională este importantă prin modalitatea de transformare a cunoștințelor teoretice în abilități practice și prin crearea unor medii de proiectare și elaborare a produselor.

STEAM spre deosebire de STEM se concentrează nu numai pe abilități tehnice, cum ar fi rezolvarea problemelor, ci și pe abilități precum conversația și creativitatea, cultivând simțul artistic al studenților. Altfel zis, STEAM este o versiune avansată a educației STEM. Educația STEAM urmează o abordare educațională transversală în care știința, tehnologia,

ingineria, matematica și arta - sunt predate pentru a dezvolta creativitatea, abilitățile de gândire critică și logică. Programele și activitățile practice STEAM permit însușirea conceptelor din știință, matematică, fizică și a altor discipline din domeniul științelor reale la un nivel mult mai înalt decât în cazul modului tradițional de predare. Prin intermediul activităților practice elevul/studentul descoperă aplicații din lumea reală, își dezvoltă abilitățile sociale, comunicarea, flexibilitatea și inițiativa.

Învățarea STEAM promovează gândirea critică, curiozitatea, persistența, luarea deciziilor, leadershipul, antreprenoriatul, acceptarea eșecului și multe altele. Aceste abilități ajută la pregătirea elevilor pentru viitorul lor. În același timp acesta e un domeniu flexibil și adaptabil care răspunde nevoii constante a lumii noastre de inovare științifică, tehnologică, socială și economică. Persoanele implicate în învățarea prin abordarea STEAM au potențialul de a aduce contribuții semnificative în diverse domenii profesionale, inclusiv în sectorul medical, ingineria software, arhitectură și multe altele. În timp ce disciplinele tradiționale utilizează metode convenționale de predare, educația prin abordarea STEAM implică o abordare aplicată, care include experimente practice și proiecte de simulare. Este esențial ca practicile tradiționale de predare să fie adaptate pentru a pregăti viitoarea forță de muncă în conformitate cu cerințele în continuă schimbare ale pieței. [1]. În aceste condiții profesorul trebuie să acționeze ca un cercetător, deoarece noile cunoștințe sunt în continuă revizuire [2]. Conform cercetărilor, activitățile STEAM implică în mod activ studenții/elevii și încurajează învățarea autonomă și motivația [3]. În zilele noastre este esențial ca indivizii să dobândească competențe variate și să-și dezvolte abilități practice în domeniul ingineriei [4]. Accentul pus pe tehnologie și inginerie în cadrul abordării STEAM oferă elevilor o perspectivă interdisciplinară încă de la o vârstă fragedă, permițându-le să aplice cunoștințele teoretice în practică și consolidându-le poziția în era tehnologiei informaționale și a comunicațiilor [5]. Prin implementarea activităților de învățare STEAM, care integrează cunoștințe din domenii precum știința, tehnologia, ingineria și matematica, elevii sunt încurajați să-și îmbunătățească abilitățile de rezolvare a problemelor. Această abordare îi ajută să adopte gândirea critică și curiozitatea ca aspecte esențiale ale vieții lor [6]. Deși metodele de aplicare a educației STEAM în clasă pot fi variate, există un proces de integrare a beneficiilor provenite din mai multe discipline, bazat pe o abordare interdisciplinară.

Cea mai eficientă modalitate de a încorpora STEAM este introducerea învățării bazate pe proiecte (PBL) în care studenții sunt rugați să lucreze practic la proiecte.

Proiectul STEAM „Simularea unui Sistem Solar Utilizând Programarea Orientată pe Obiecte în limbajul de programare C++”

Grupul ţintă:

Grupul țintă al acestui proiect este format din studenții anului 2,3 de la facultatea

de fizică și matematică. Acești studenți au deja cunoștințe fundamentale în domeniile fizicii, matematicii și programării, iar acest proiect își propune să le ofere oportunitatea de a aplica și dezvolta aceste cunoștințe într-un context practic și interdisciplinar.

În mod specific, grupul țintă poate include:

1. Studenții anului 2, 3 de la facultatea de fizică și matematică ;
2. Studenți pasionați de informatică și programare;
3. Studenți interesați de cercetare științifică și dezvoltare tehnologică;
4. Profesori și mentori;
5. Comunitatea academică;
6. Publicul interesat de astronomie și științe spațiale.

Problema rezolvată prin proiect: ***Dezvoltarea unei aplicații software ce simulează mișcarea corpurilor cerești într-un sistem solar.***

Această simulare poate fi utilă pentru mai multe scopuri și poate aborda diverse probleme:

- Înțelegerea fenomenelor astronomice.
- Educație și învățare interactivă.
- Cercetare și explorare spațială.
- Testarea și validarea modelelor astronomice.
- Explorarea interacțiunilor gravitaționale.

Prin dezvoltarea acestei aplicații, proiectul abordează o problemă complexă din domeniul astronomiei și al științelor spațiului și oferă o soluție practică și interactivă pentru înțelegerea și explorarea acestor fenomene.

Obiectivul general al proiectului:

Obiectivul general al proiectului „Simularea unui Sistem Solar Utilizând Programarea Orientată pe Obiecte în C++” este de a dezvolta o aplicație software care oferă o simulare interactivă a mișcării corpurilor cerești într-un sistem solar, utilizând principiile și tehnicile de programare orientată pe obiecte în limbajul C++.

Acest obiectiv general implică mai multe aspecte:

- Dezvoltarea aplicației software
- Implementarea principiilor de programare orientată pe obiecte
- Simularea mișcării corpurilor cerești
- Testarea și validarea aplicației

În ansamblu, obiectivul general al proiectului este de a crea o aplicație software interactivă și educativă care permite utilizatorilor să exploreze și să înțeleagă mișcarea corpurilor cerești în sistemul solar, integrând conceptele de programare orientată pe obiecte cu cunoștințele de fizică și matematică într-un mod practic și inovator.

Obiectivele specifice ale proiectului:

Obiectivele specifice ale proiectului „Simularea unui Sistem Solar Utilizând

Programarea Orientată pe Obiecte în C++" sunt detalii și acțiuni specifice pe care proiectul trebuie să le îndeplinească pentru a atinge obiectivul general. Acestea includ:

1. Definirea și implementarea claselor pentru corpurile cerești
2. Implementarea algoritmilor de simulare a mișcării
3. Interacțiunea utilizatorilor sau interfața utilizatorului
4. Optimizarea performanței și eficiența programului
5. Testarea și depanarea
6. Documentarea proiectului
7. Construirea unei machete „Zona de simulare a Sistemului solar".

Prin îndeplinirea acestor obiective specifice, proiectul va reuși să creeze o aplicație funcțională și interactivă care să permită utilizatorilor să exploreze și să înțeleagă mișcarea corpurilor cerești în sistemul solar folosind programarea orientată pe obiecte în limbajul C++.

Ipoteza proiectului:

Ipoteza proiectului „Simularea unui Sistem Solar Utilizând Programarea Orientată pe Obiecte în C++" este că prin aplicarea principiilor și tehnicilor de programare orientată pe obiecte în limbajul C++, putem dezvolta o aplicație software capabilă să simuleze mișcarea corpurilor cerești în sistemul solar într-un mod precis, eficient și interactiv. Astfel, ipoteza presupune că:

1. Programarea orientată pe obiecte este adecvată pentru dezvoltarea aplicației
2. Algoritmii și formulele matematice sunt corect implementate
3. Interfața utilizatorului permite interacțiunea intuitivă și eficientă
4. Simularea sistemului solar este precisă și realistică

Prin urmare, ipoteza proiectului sugerează că prin implementarea adecvată a programării orientate pe obiecte în limbajul C++, putem dezvolta o aplicație software care să ofere o simulare eficientă și precisă a mișcării corpurilor cerești în sistemul solar, contribuind astfel la înțelegerea și explorarea fenomenelor astronomice și a principiilor fizicii spațiului cosmic.

Scopul proiectului:

Scopul proiectului „Simularea unui Sistem Solar Utilizând Programarea Orientată pe Obiecte în C++" este de a crea o aplicație software interactivă și educațională care să permită utilizatorilor să exploreze și să înțeleagă mișcarea corpurilor cerești din sistemul solar. Scopurile specifice includ:

- Educația și înțelegerea;
- Explorarea interactivă;
- Promovarea interesului pentru știință și tehnologie;
- Utilizarea programării orientate pe obiecte;
- Contribuția la educația și cercetarea științifică.

Prin îndeplinirea acestor obiective, scopul general al proiectului este de a oferi o resursă educațională și de explorare interactivă pentru a înțelege și aprecia complexitatea și frumusețea sistemului solar, integrând simultan principiile programării orientate pe obiecte într-un mediu educațional și interactiv.

Materiale necesare:

Pentru proiectul „Simularea unui Sistem Solar Utilizând Programarea Orientată pe Obiecte în C++”, sunt necesare următoarele materiale:

1. Calculator sau laptop
2. Mediu de dezvoltare integrat (IDE)
3. Biblioteci de grafică (opțional)
4. Resurse pentru învățare
5. Date și informații despre sistemul solar
6. Plastic, lemn, hârtie
7. Timp și dedicare

Activități care vor fi realizate

Sarcini specifice	Intervalul de timp	Responsabilitatea
1. Planificarea și proiectarea aplicației: **Sarcina1.** Identificarea cerințelor și specificațiilor aplicației, inclusiv funcționalitățile necesare și interfața utilizatorului. **Sarcina2.** Proiectarea arhitecturii aplicației, inclusiv definirea claselor și a relațiilor între acestea pentru a reprezenta corpurile cerești din sistemul solar.	1 săptămână	Responsabilitatea pentru planificarea și proiectarea aplicației poate reveni unei echipe de studenți sau unei persoane desemnate, în funcție de dimensiunea și complexitatea proiectului. **Echipa de dezvoltare:** 1. Fiecare membru al echipei poate contribui la planificarea și proiectarea aplicației. 2. Un lider de proiect sau un manager de proiect poate fi responsabil de coordonarea activităților de planificare și proiectare și de asigurarea ca obiectivele proiectului să fie atinse în termenii stabiliți. 3. Studenții pot colabora pentru a propune soluții tehnice, pentru a identifica cerințele și pentru a stabili arhitectura aplicației. **Persoana desemnată:** În cazul unui proiect mai mic poate fi desemnată o singură persoană să aibă responsabilitatea principală pentru planificare și proiectare. Această persoană poate fi un student mai experimentat.

2. Implementarea claselor și obiectelor: **Sarcina1.** Crearea claselor pentru diferitele tipuri de corpuri cerești, cum ar fi planetele, sateliții, cometele etc. **Sarcina2.** Definirea atributelor și metodelor asociate fiecărei clase pentru a stoca informațiile relevante și pentru a simula comportamentul mișcării acestor corpuri.	1 săptămână	Responsabilitatea pentru implementarea claselor și obiectelor în cadrul proiectului "Simularea unui Sistem Solar Utilizând Programarea Orientată pe Obiecte în C++" poate fi atribuită studenților care studiază și cunosc limbajul C++ și programarea orientată pe obiecte. Ei sunt responsabili de: - Proiectarea structurii claselor și obiectelor - Definirea atributelor și metodelor - Implementarea relațiilor între clase - Gestionarea erorilor și excepțiilor - Documentarea codului.
3. Implementarea algoritmilor de simulare: **Sarcina1.** Dezvoltarea algoritmilor și a logicii necesare pentru a simula mișcarea corpurilor cerești în sistemul solar. **Sarcina2.** Utilizarea legilor gravitației și a altor principii fizice pentru a calcula traiectoriile orbitale și pentru a prevedea pozițiile viitoare ale corpurilor cerești.	2-3 săptămâni	Responsabilitatea pentru implementarea algoritmilor poate fi atribuită studenților cu cunoștințe în fizică, matematică și algoritmizare, în special cei care au o înțelegere solidă a mișcării corpurilor cerești și a legilor fizicii care guvernează comportamentul lor în spațiu. Mai specific, responsabilitățile pentru implementarea algoritmilor de simulare ar putea include: - Studierea și înțelegerea legilor fizicii - Proiectarea algoritmilor de simulare - Implementarea algoritmilor în limbajul C++ - Optimizarea performanței - Testarea și validarea algoritmilor Documentarea algoritmilor
4 **Crearea interfeței utilizatorului:** **Sarcina1.** Proiectarea și implementarea unei interfețe utilizator intuitive și ușor de utilizat pentru a permite utilizatorilor să interacționeze cu simularea sistemului solar. **Sarcina2.** Integrarea elementelor grafice și a controalelor pentru a permite utilizatorilor să navigheze prin sistemul solar și să controleze simularea.	2-3 săptămâni în paralel cu implementarea algoritmilor de simulare	Responsabilitatea pentru crearea interfeței utilizatorului în cadrul proiectului poate fi atribuită unui student familiarizat cu dezvoltarea interfețelor utilizator. Acest responsabil va fi însărcinat cu următoarele: - Proiectarea interfeței utilizatorului (UI) - Implementarea interfeței utilizatorului (UI) - Integrarea funcționalităților interfeței utilizatorului (UI) cu logica aplicației - Testarea și optimizarea interfeței utilizatorului (UI): - Documentarea interfeței utilizatorului (UI)
5. **Testarea și depanarea:** **Sarcina1.** Efectuarea testelor riguroase pentru a verifica corectitudinea și funcționarea	1 săptămână	Responsabilitatea pentru testarea și depanarea în cadrul proiectului poate fi atribuită unui student cu aptitudini de testare de software sau poate fi împărțită între membrii echipei de dezvoltare.

corectă a aplicației în diferite scenarii și condiții. **Sarcina2.** Identificarea și remedierea oricăror erori sau bug-uri care pot apărea în timpul dezvoltării sau testării aplicației.		Responsabilitățile pentru testare și depanare includ următoarele aspecte: – Elaborarea planului de testare – Crearea și implementarea testelor – Execuția testelor – Identificarea și raportarea bug-urilor – Reproducerea și investigarea problemelor – Depanarea și remedierea problemelor – Retestarea. Documentarea testelor și a rezultatelor
6. **Documentarea proiectului:** **Sarcina1.** Documentarea adecvată a codului sursă, a algoritmilor, a interfeței de utilizator și a altor aspecte ale proiectului. **Sarcina2.**Crearea documentației tehnice și a ghidurilor pentru utilizatori pentru a facilita înțelegerea și utilizarea aplicației.	Pe tot parcursul proiectului	Responsabilitatea pentru documentarea proiectului poate fi atribuită unui membru al echipei sau poate fi împărțită între membrii echipei de dezvoltare. Principalele responsabilități ale responsabilului pentru documentație includ: – Documentarea codului sursă – Documentarea arhitecturii aplicației – Ghiduri pentru instalare și configurare – Ghiduri pentru utilizator – Documentarea proceselor de dezvoltare – Manual de mentenanță și depanare: – Actualizarea și menținerea documentației – Asigurarea accesibilității și a relevanței
7. **Optimizarea performanței:** **Sarcina1.** Asigurarea că aplicația este optimizată pentru a funcționa eficient și fără probleme, chiar și atunci când simulează un număr mare de corpurile cerești.	Pe tot parcursul proiectului	Responsabilitatea pentru optimizarea performanței în cadrul proiectului " poate fi atribuită unui membru al echipei de dezvoltare care are cunoștințe și experiență în optimizarea codului și a algoritmilor. Acest responsabil ar putea un student pasionat de perfecționarea performanței aplicațiilor software. Principalele responsabilități ale responsabilului pentru optimizarea performanței includ: – Analiza și evaluarea performanței: – Optimizarea algoritmilor: – Optimizarea structurii datelor: – Optimizarea codului sursă: – Gestionarea resurselor – Testarea și validarea performanței – Documentarea optimizărilor
8. Construirea unei machete "Zona de simulare a sistemului solar". **Sarcina1.** Construirea platformei de afișare a sistemului solar: pe care să fie montate modelele reduse ale corpurilor cerești din sistemul solar, cum ar fi Soarele, planetele, luna. **Sarcina2.** Integrarea unui sistem de	Pe tot parcursul proiectului	Responsabilitatea pentru construirea machetei "Zona de simulare a sistemului solar" poate fi atribuită unui membru/membri ai echipei, care au aptitudini și interes în domeniul ingineriei, designului sau construcțiilor.

iluminare pe platformă, care reproduce lumina solară și oferă un aspect realist al simulării. **Sarcina3**. (Elemente de design și estetică). Utilizarea culorilor și materialelor care ar evidenția frumusețea și complexitatea sistemului solar, pentru realizarea conceptului atractiv și estetic al machetei.		

Realizarea programului de simulare a sistemului solar utilizând programarea orientată pe obiecte în limbajul C++

Pentru realizarea codului de program studenții/elevii trebuie să cunoască:

1. *Din informatică*:
   - Programarea orientată pe obiecte (POO): Cunoștințe despre clase, obiecte, moștenire, încapsulare și polimorfism.
   - Utilizarea bibliotecilor: Capacitatea de a folosi biblioteci grafice precum SFML sau OpenGL pentru a crea interfețe grafice pentru simulare.
   - Structuri de date: Înțelegerea structurilor de date precum vectori, matrice și liste este importantă pentru gestionarea și manipularea informațiilor legate de obiectele din sistemul solar.
   - Matematică computațională: Cunoștințele despre algebră liniară, geometrie și calcule vectoriale sunt utile pentru a efectua calculele necesare pentru simularea mișcării corpurilor cerești.
2. *Matematică*:
   - Trigonometrie: Înțelegerea conceptelor trigonometrice pentru calcularea pozițiilor și a mișcării corpurilor pe orbite.
   - Calcul diferențial și integral: Cunoștințele despre calcul diferențial și integral sunt utile pentru a înțelege mișcarea corpurilor cerești și pentru a dezvolta algoritmi pentru simularea acesteia.
3. *Fizică:*
   - Legea gravitației universale: Cunoașterea legii gravitației universale a lui Newton și modul în care aceasta influențează mișcarea corpurilor cerești pentru a realiza simularea corectă a sistemului solar.
   - Concepte fundamentale de fizică: Înțelegerea conceptelor de forță, masă, viteză și accelerație pentru a simula corect mișcarea corpurilor cerești în cadrul sistemului solar.

Prin combinarea cunoștințelor din aceste domenii, studenții/elevii vor fi capabili să dezvolte un cod de program funcțional și precis pentru proiectul de simulare a sistemului solar folosind programarea orientată pe obiecte în limbajul C++.

Codul de program:

```
const double G = 6.67430e-11; // Constanta gravitațională
const double AU = 1.496e11; // Unitate astronomică (distanța medie de la Pământ la Soare)

// Clasa pentru reprezentarea corpurilor cerești
class CorpCeresc {
protected:
   double masa;
   double x, y; // Poziția în plan
   double vitezaUnghiulara; // Viteza unghiulară (rad/s)
   double radius; // Rază orbitală
public:
   // Constructor
   CorpCeresc(double m, double r, double angularVel) : masa(m), x(0), y(0), vitezaUnghiulara(angularVel),
radius(r) {}

   // Funcție pentru actualizarea poziției
   void updatePosition(double dt) {
      double angle = vitezaUnghiulara * dt;
      x = radius * cos(angle);
      y = radius * sin(angle);
   }

   // Funcții pentru obținerea poziției
   double getX() const { return x; }
   double getY() const { return y; }
};

// Clasa pentru reprezentarea sistemului solar
class SolarSystem {
private:
   CorpCeresc soare;
   CorpCeresc pamant;
   CorpCeresc luna;
public:
   // Constructor
   SolarSystem(double soareMasa, double pamantMasa, double lunaMasa, double pamantDistanta,
double lunaDistanta, double pamantVitezaUnghiulara, double lunaVitezaUnghiulara) :
      soare(soareMasa, 20, 0),
      pamant(pamantMasa, pamantDistanta, pamantVitezaUnghiulara),
      luna(lunaMasa, lunaDistanta, lunaVitezaUnghiulara) {}

   // Funcție pentru simularea sistemului solar
   void simulate(double dt) {
```

```
        pamant.updatePosition(dt);
        luna.updatePosition(dt);
    }

    // Funcții pentru obținerea pozițiilor corpurilor cerești
    double getPamantX() const { return pamant.getX(); }
    double getPamantY() const { return pamant.getY(); }
    double getLunaX() const { return luna.getX(); }
    double getLunaY() const { return luna.getY(); }
};

int main() {
    // Inițializarea sistemului solar
    double soareMasa = 1.989e30; // Masă a Soarelui în kg
    double pamantMasa = 5.972e24; // Masă a Pământului în kg
    double lunaMasa = 7.342e22; // Masă a Lunii în kg
    double pamantDistanta = AU; // Distanța medie între Pământ și Soare în metri
    double lunaDistanta = 3.844e8; // Distanța medie între Pământ și Lună în metri

    // Viteze unghiulare inițiale pentru Pământ și Lună
    double pamantVitezaUnghiulara = 2 * 3.14 / (365.25 * 24 * 3600); // 1 rotație pe an
    double lunaVitezaUnghiulara = 2 * 3.14 / (27.3 * 24 * 3600); // 1 rotație pe lună

    SolarSystem solarSystem(soareMasa, pamantMasa, lunaMasa, pamantDistanta, lunaDistanta,
pamantVitezaUnghiulara, lunaVitezaUnghiulara);
    // Crearea ferestrei SFML
    sf::RenderWindow window(sf::VideoMode(800, 800), "Simulare Sistem Solar");
   window.setFramerateLimit(60);

    while (window.isOpen()) {
        sf::Event event;
        while (window.pollEvent(event)) {
            if (event.type == sf::Event::Closed)
                window.close();        }

        // Simularea sistemului solar pentru un cadru
       solarSystem.simulate(1.0/60); // Simulare pentru un cadru

        // Clear window
        window.clear(sf::Color::Black);
        // Desenarea Soarelui
        sf::CircleShape soare(40);
        soare.setFillColor(sf::Color::Yellow);
        soare.setPosition(400 - soare.getRadius(), 400 - soare.getRadius());
```

```
        window.draw(soare);

        // Desenarea Pământului
        sf::CircleShape pamant(15);
        pamant.setFillColor(sf::Color::Blue);
        pamant.setPosition(200 + solarSystem.getPamantX() / AU - pamant.getRadius(), 200 +
solarSystem.getPamantY() / AU - pamant.getRadius());
        window.draw(pamant);

         // Desenarea Lunii
        sf::CircleShape luna(5);
        luna.setFillColor(sf::Color::White);
        luna.setPosition(220 + solarSystem.getLunaX() / AU - luna.getRadius(), 220 + solarSystem.getLunaY()
/ AU - luna.getRadius());
        window.draw(luna);

        // Afisarea ferestrei
        window.display();
      }
    return 0;
}
```

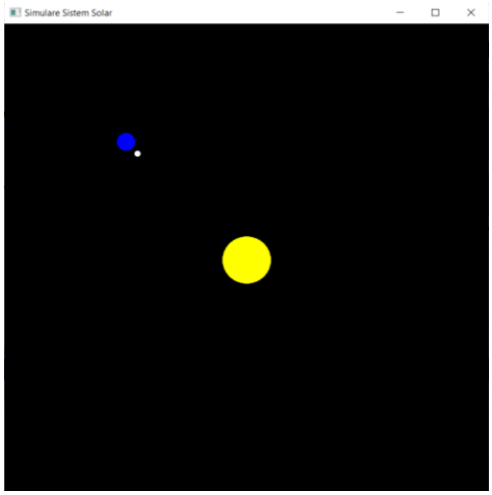

Figura 1. Simularea Sistemului Solar

Concluzii

Proiectul STEAM de simulare a unui Sistem Solar, realizat prin intermediul Programării Orientate pe Obiecte în limbajul C++, demonstrează nu doar abilitățile practice în dezvoltarea software, ci și importanța integrării STEAM în contextul educației. Acest proiect oferă studenților/elevilor oportunitatea de a explora concepte științifice complexe și de a le aplica într-un mediu interactiv, consolidându-și înțelegerea și competențele tehnologice într-un mod captivant și relevant pentru lumea actuală. Prin

această experiență, studenții/elevii sunt încurajați să devină inovatori și să abordeze problemele viitoare cu încredere și creativitate.

Implementarea unui model de proiect STEAM în cadrul disciplinei de Programare Orientată pe Obiecte oferă studenților/elevilor oportunități remarcabile de a integra cunoștințele teoretice cu abilitățile practice, consolidându-și astfel înțelegerea și competențele în domenii cheie ale tehnologiei. Prin intermediul acestui model, studenții/elevii sunt încurajați să exploreze și să inoveze, pregătindu-se pentru provocările și oportunitățile din lumea tehnologică în continuă evoluție.

Articol realizat în cadrul proiectului de cercetări științifice „Metodologia implementării TIC în procesul de studiere a științelor reale din perspectiva conceptului STEAM și Inteligenței Artificiale”, codul 040101, din cadrul Programului instituțional de cercetare (2024-2027), aprobat prin Ordin MEC nr. 102 din 01.02.2024

Bibliografie

1. TZAGKARAKI, E.; PAPADAKIS, S.; KALOGIANNAKIS, M. Exploring the Use of Educational Robotics in primary school and its possible place in the curricula. In: *Educational Robotics International Conference*, 2021. pp. 216-229. Springer, Cham. https://doi.org/10.1007/978-3-030-77022-8 19
2. CHATZOPOULOS, A.; KALOGIANNAKIS, M.; PAPADAKIS, S.; PAPOUTSIDAKIS, M.; ELZA, D.; PSYCHARIS, S. DuBot: An Open-Source, Low-Cost Robot for STEM and Educational Robotics. In: *Handbook of Research on Using Educational Robotics to Facilitate Student Learning,* 2021. pp. 441-465. IGI Global. https://doi.org/10.4018/978-1-7998-6717-3.ch018
3. POULTSAKIS, S.; PAPADAKIS, S.; KALOGIANNAKIS, M.; PSYCHARIS, S. The management of digital learning objects of natural sciences and digital experiment simulation tools by teachers. In: *Advances in Mobile Learning Educational Research*, 2021. Nr. 1(2), pp. 58-71. https://doi.org/10.25082/AMLER.2021.02.002
4. SIEW, N. M.; AMIR, N.; CHONG, C. L. The perceptions of pre-service and in-service teachers regarding a project-based STEM approach to teaching science. In: SpringerPlus, 2015. Nr. 4(1), pp. 17-20. https://doi.org/10.1186/2193-1801-4-8
5. SMYRNOVA-TRYBULSKA, E.; MORZE, N.; KOMMERS, P.; ZUZIAK, W.; GLADUN, M. Educational robots in primary school teachers’ and students’ opinion about STEM education for young learners. In: *Proc. of International Conferences ITS, ICEduTech and STEM* 2016. pp. 197-204. www.roboty.bielsko.pl
6. GUNAWAN, S.; SHIEH, C. J. Effects of the implementation of STEM curriculum integration model to living technology teaching on business school students’ learning effectiveness. In: *Contemporary Educational Technology*, 2020. Nr. 12(2), pp. 1-7. https://doi.org/10.30935/cedtech/8583