

UNIVERSITATEA TEHNICĂ A MOLDOVEI
FACULTATEA CALCULATOARE, INFORMATICĂ,
MICROELECTRONICĂ

Rotaru Lilia, Oșovschi Mariana,
Cărbune Viorel, Ursu Adriana

GRAFICA PE CALCULATOR

ÎNDRUMAR METODIC PENTRU LUCRĂRI DE
LABORATOR



Chișinău
Editura „Tehnica-UTM” 2025

II. Grafica vectorială

Grafica vectorială este un tip de grafică pe calculator care utilizează descrierea matematică a formelor geometrice (linii, cercuri, dreptunghiuri, poligoane, etc.), spre deosebire de grafica raster, care se bazează pe pixeli. Aceasta este folosită pentru a crea imagini care pot fi scalate sau transformate fără pierderea calității, deoarece nu sunt bazate pe o grilă fixă de pixeli.

2.1 Caracteristici graficii vectoriale

1. **Independența de rezoluție:** imaginile vectoriale pot fi redimensionate, păstrându-și claritatea și detaliile. Fie că sunt mărite sau micșorate, ecuațiile matematice care stau la baza graficii vectoriale se vor recalcula, deci imaginile sunt "redesenate", astfel nu sunt pierderi în calitate, ceea ce le face ideale pentru tipărire și pentru utilizarea în diferite dispozitive cu rezoluții variate.

2. **Componente geometrice:** grafica vectorială este construită din obiecte geometrice definite prin formule matematice. De exemplu, un cerc este definit prin raza și coordonatele centrului său, o linie prin punctele de început și sfârșit, iar un poligon prin coordonatele colțurilor sale.

3. **Fișiere de dimensiuni mici:** deoarece grafica vectorială este descrisă prin formule matematice, ele ocupă un spațiu mai mic în fișiere fizice comparativ cu imaginile raster (care sunt formate dintr-o rețea de pixeli). Acest lucru le face eficiente pentru stocare și partajare.

4. **Editabilitate:** obiectele vectoriale pot fi editate individual fără a afecta alte părți ale imaginii. De exemplu, un cerc poate fi modificat fără a afecta alte elemente ale unei scene grafice, ceea ce face ca lucrul cu grafica vectorială să fie mai flexibil decât în cazul graficii raster.

5. **Domenii de aplicare:** grafica vectorială este folosită pentru crearea de logo-uri, iconițe, ilustrații, în tipografie și desene tehnice. Un alt domeniu este design graphic – crearea de materiale

publicitare, afișe, pliante și alte produse de design grafic. Grafica vectorială este utilizată crea animații, deoarece aceasta permite mișcarea și scalarea ușoară a elementelor grafice.

Avantaje ale graficii vectoriale:

- **Scalabilitate:** Fără pierderi de calitate la redimensionare.
- **Editabilitate:** Ușor de modificat și de personalizat fiecare element.
- **Dimensiune mică a fișierelor:** Mai eficiente pentru stocare și partajare.
- **Calitate constantă:** Imaginile nu devin pixelate indiferent de dimensiunea lor.

Dezavantaje:

- **Complexitate:** Crearea unor imagini foarte detaliate poate necesita un timp mai lung de lucru.
- **Limitări în detaliile fine:** Graficile vectoriale sunt mai potrivite pentru forme clare și precise, dar pot fi mai greu de folosit pentru imagini cu detalii fine sau efecte complexe de lumină și textură.

2.2 Formatul SVG

Formatul fișierilor grafice vectoriale SVG – Scalable Vector Graphics, este utilizat pentru descrierea imaginilor bidimensionale utilizând XML. SVG este un standard al organizației W3C a cărui proiectare a început în anul 1999. Acest format permite definirea imaginilor prin trei metode: text, grafică vectorială și "bitmap-uri" (harta de biți – fișiere în formatul BMP). Deși există aplicații specializate pentru crearea și editarea de SVG-uri, în acest scop poate fi utilizat orice editor textual. Vizualizarea unei imagini vectoriale SVG poate fi realizată prin utilizarea oricărui browser modern.

În prezent SVG-ul are mai multe versiuni pentru a se adapta mai bine la diferite cerințe hardware. Astfel, versiunile "SVG Tiny" și "SVG Basic" au fost create special pentru dispozitivele mobile cu resurse limitate. În același timp, profilul "SVG Print" este destinat mediilor de printare a documentelor.

Pentru animarea unei imagini SVG organizația W3C recomandă standardul "SMIL". Pe lângă recomandarea oficială mai există și alte soluții, cum ar fi "ECMAScript".

Elementul HTML <svg> reprezintă un container pentru grafică vectorială SVG.

În figura 2.1 sunt specificate versiunile browserelor care suportă complet elementul <svg>.

Element					
<svg>	4.0	9.0	3.0	3.2	10.1

Figura 2.1. Versiunile browserelor care suportă <svg>

2.3 Primitive grafice 2D în <svg>

Tagul <svg> dispune de câteva metode pentru a desena linii, poligoane, dreptunghiuri, cercuri, text și imagini grafice.

Formatul SVG dispune de câteva elemente pentru descrierea unor forme predefinite care pot fi utilizate de către dezvoltator:

✓ Dreptunghi	<rect>
✓ Cerc	<circle>
✓ Elipsă	<ellipse>
✓ Linie	<line>
✓ Polilinie	<polyline>
✓ Poligon	<polygon>
✓ Cale	<path>

Elementul <line> - este utilizat pentru a desena un segment de dreaptă:

Atribute:

x1 – abscisa x a punctului de început al segmentului de dreaptă;

y1 – ordonata y a punctului de început al segmentului de dreaptă;

x2 – abscisa x a punctului de sfârșit al segmentului de dreaptă;

y2 – ordonata y a punctului de sfârșit al segmentului de dreaptă.

Exemplu de utilizare:

```
<line x1="0" y1="0" x2="200" y2="200"
stroke="black" stroke-width="2" />
```

Formatul SVG oferă un spectru larg de proprietăți pentru contur:

stroke – culoarea liniei, conturului textului ori a figurilor;

stroke-width – grosimea liniei, conturului textului ori a conturului unui element;

Modul de utilizare a diferitor linii este prezentat în exemplu de mai jos:

```
<svg height="80" width="300">
<g fill="none">
<line stroke="red" stroke-width="1", x1="0"
y1="10" x2="200" y2="10"/>
<line stroke="black" stroke-width="2", x1="0"
y1="20" x2="200" y2="20"/>
<line stroke="blue" stroke-width="3", x1="0"
y1="30" x2="200" y2="30"/>
</g>
</svg>
```

Rezultatul execuției acestui cod este arătat în figura 2.2



Figura 2.2. Definirea liniilor de culori și grosimi diferite

stroke-linecap – atributul specifică stilul extremităților liniei sau conturului. Există trei opțiuni principale pentru acest atribut:

- **butt**: capătul liniei este drept și se termină exact la coordonata specificată, este implicit;
- **round**: capătul liniei este rotunjit și extins cu o jumătate din lățimea liniei;

- **square**: capătul liniei este drept, dar se extinde dincolo de coordonată, adăugând o lungime egală cu jumătate din grosimea liniei;

Exemplu de folosirea a diferitor attribute a parametrului stroke-linecap, rezultatul execuției acestui cod este arătat în figura 2.3:

Exemplu:

```
<svg height="80" width="300">
  <g fill="none" stroke="black" stroke-
width="10">
    <line stroke-linecap="butt", x1="10" y1="20"
x2="200" y2="20"/>
    <line stroke-linecap="round", x1="10" y1="40"
x2="200" y2="40" />
    <line stroke-linecap="square", x1="10" y1="60"
x2="200" y2="60" />
  </g>
</svg>
```



Figura 2.3. Tipuri atributului stroke-linecap

Atributul stroke-dasharray – este utilizat pentru crearea liniilor punctate, valoarea constă într-o listă de numere care definesc lungimea liniuțelor și a spațiilor. De exemplu, **stroke-dasharray="5,5"** va crea o linie cu segmente de 5 unități, urmate de spații de 5 unități.

Exemplu:

```
<svg height="80" width="300">
  <g fill="none" stroke="black" stroke-width="4">
    <line stroke-dasharray="5,5", x1="10" y1="20"
x2="200" y2="20" />
```

```

<line stroke-dasharray="10,10", x1="10" y1="40"
x2="200" y2="40" />
<line stroke-dasharray="20,10,5,5,5,10",
x1="10" y1="60" x2="200" y2="60" />
</g>
</svg>

```

Rezultatul execuției acestei secvențe de cod este arătat în figura 2.4.

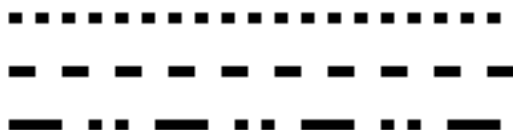


Figura 2.4. Tipuri de linii punctate

Elementul <polyline> – este utilizat pentru a crea linii frânte construite din segmente de drepte conectate prin câteva puncte.

Atribut:

points – (o listă de coordonate x,y) conține lista de perechi de coordonate (x, y) necesare pentru desenarea liniei frânte.

Exemplu:

```

<svg height="200" width="500">
<polyline points="20,20 40,25 60,40 80,120
120,140 200,180" style = "fill: none; stroke:
black; stroke-width:3" />
</svg>

```

Rezultatul rulării acestei secvențe de program este prezentat în figura 2.5.



Figura 2.5. Definirea liniei frânte

Elementul <circle> – definește un cerc

Atributele:

cx – coordonata pe axa x a centrului cercului;

cy – coordonata pe axa y a centrului cercului;

r – raza cercului.

Exemplu:

```
<svg width="100" height="100">  
  <circle cx="50" cy="50" r="40" stroke="green"  
stroke-width="4" fill="yellow" />  
</svg>
```

Rezultatul rulării acestei secvențe de program este prezentat în figura 2.6.



Figura 2.6. Definirea cercului în <svg>

Elementul <ellipse> – se utilizează pentru a desena o elipsă

Atributele:

cx– coordonata pe axa x a centrului elipsei;

cy– coordonata pe axa y a centrului cercului;

rx – lungimea razei elipsei pe axa x (raza orizontală – lățimea);

ry – lungimea razei elipsei pe axa y (raza verticală – înălțimea).

Elipsa reprezintă un caz generalizat al cercului. Diferența dintre aceste figuri constă în faptul că elipsă are lungimile diferite a razelor pe axele x și y, iar în cazul cercului acestea sunt egale.

Exemplu:

```
<svg height="140" width="500">
```



```
<ellipse cx="200" cy="80" rx="100" ry="50"
  style="fill:yellow;stroke:purple;stroke-width:
  2" />
</svg>
```

Rezultatul rulării acestei secvențe de program este prezentat în figura 2.7.

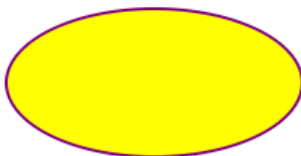


Figura 2.7. Definirea unei elipse

Elementul <rect> – permite crearea dreptunghiului și figurilor derivate de la acesta:

Atributele:

x – coordonata pe axa x a colțului din partea stângă de sus a dreptunghiului;

y – coordonata pe axa y a colțului din partea stângă de sus a dreptunghiului;

rx – raza pe axa x (pentru rotundirea vîrfurilor elementului);

ry – raza pe axa y (pentru rotundirea vîrfurilor elementului);

width – lățimea dreptunghiului;

height – înălțimea dreptunghiului.

Exemplu:

```
<svg width="400" height="180">
  <rect x="50" y="20" rx="20" ry="20" width="150
height="150"
  style="fill:red;stroke:black;stroke-
width:5;opacity:0.5"/>
</svg>
```

Rezultatul rulării acestei secvențe de program este prezentat în figura 2.8.



Figura 2.8. Definirea unui dreptunghi

Elementul <polygon> – este utilizat pentru crearea elementelor grafice care trebuie să conțină cel puțin 3 laturi. Poligoanele sînt formate din linii drepte, iar figura reprezintă un contur închis.

Atributele:

points – conține coordonatele x și y a fiecărui punct din mulțimea punctelor care constituie vîrfurile poligonului;

Numărul total al punctelor trebuie să fie par.

fill-rule – reprezintă un atribut care definește algoritmul utilizat pentru reprezentarea interiorului figurii (metoda de umplere/colorare) și poate lua valorile evenodd sau nonzero.

Exemplu:

```
<svg height="250" width="500">  
  <polygon points="220,10 300,210 170,250  
123,234" style="fill:lime;stroke:purple;stroke-  
width:1" />  
</svg>
```

Rezultatul rulării acestei secvențe de program este prezentat în figura 2.9.

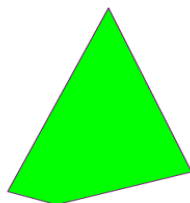


Figura 2.9. Definirea unui poligon cu patru laturi

Elementul <path> – este utilizat pentru a defini o cale.

Atributele:

d – parametrul path data reprezintă un set de comenzi care definește calea;

pathLength – dacă este prezent, calea este scalată astfel încât lungimea calculată a căii care trece prin puncte să fie egală cu această valoare;

transform – lista de transformări.

Parametrul path data poate lua valorile:

- ✓ **M** = moveto
- ✓ **L** = lineto
- ✓ **H** = horizontal lineto
- ✓ **V** = vertical lineto
- ✓ **C** = curveto
- ✓ **S** = smooth curveto
- ✓ **Q** = quadratic Bézier curve
- ✓ **T** = smooth quadratic Bézier curveto
- ✓ **A** = elliptical Arc
- ✓ **Z** = closepath

Toate comenzile de mai sus pot fi scrise și cu minuscule. Utilizarea majusculilor semnifică poziționarea absolută, utilizarea minusculilor semnifică poziționarea relativă.

Curbele Bézier sunt utilizate pentru modelarea liniilor netede care pot fi scalate la infinit fără pierderea calității. Pentru reprezentarea curbei Bézier utilizatorul trebuie să definescă două puncte care indică cele două capete ale curbei și unul sau două puncte de control. O curbă Bézier cu un punct de control poartă numele de curbă Bézier pătratică, iar cele cu două puncte de control – curbe Bézier cubice.

Elementul <text> – este utilizat pentru a defini un text.

Atributele:

x – lista de coordonate pe axa x a caracterelor.

Elementul **n** din lista de coordonate de pe axa **x** corespunde caracterului cu numărul de ordine **n** din text. Dacă există caractere suplimentare cărora nu le corespund valori pentru coordonata de pe

axa **x**, acestea sunt plasate după ultimul caracter. Valoare implicită este 0.

y – lista de coordonate pe axa **y** a caracterelor;

dx – lista valorilor pe axa **x** cu care se deplasează caracterele relative la poziția absolută a ultimului caracter desenat;

dy – lista valorilor pe axa **y** cu care se deplasează caracterele relative la poziția absolută a ultimului caracter desenat;

rotate – lista unghiurilor de rotație.

Valoarea **n** din lista de rotație se aplică caracterului **n** din textul afișat. Caracterele suplimentare nu se rotesc.

2.4 Integrarea graficii vectoriale în p5.js Web Editor

Pentru integrarea codul SVG în limbajul HTML cu ajutorul editorului web p5.js este nevoie de logarea cu ajutorul contului Google, Account sau GitHub pe platforma respectivă. Pentru vizualizarea fișierele din componența proiectului trebuie accesat butonul cu pictograma de săgeată din partea stângă a ferestrei, după cum este arătat în figura 2.10. În urma acțiunii date se deschide fereastra Sketch Files în care se pot vedea fișierele componente ale proiectului: index.html, sketch.js și style.css. Fișierul index.html conține codul HTML pentru marcarea interfeței componentelor paginii web a proiectului. Adăugarea codului SVG în proiect se face prin editarea fișierului index.html și anume înlocuirea tag-ului `<script src="sketch.js"></script>` din interiorul tag-ului `<body></body>` cu tagul `<svg></svg>` necesar, figura 2.10.

În exemplul de mai jos este prezentată crearea curbei Bézier pătratică, unde punctele **A** și **C** reprezintă vîrfurile iar punctul **B** reprezintă punctul de control:

Vizualizarea rezultatului execuției programului realizat poate fi observată în partea dreaptă a editorului p5.js Web Editor în fereastra Preview, după cum este arătat în figura 2.10. Dacă imaginea abținută

va fi mărită sau micșorată, se poate de observat că calitatea acesteia nu se modifică.

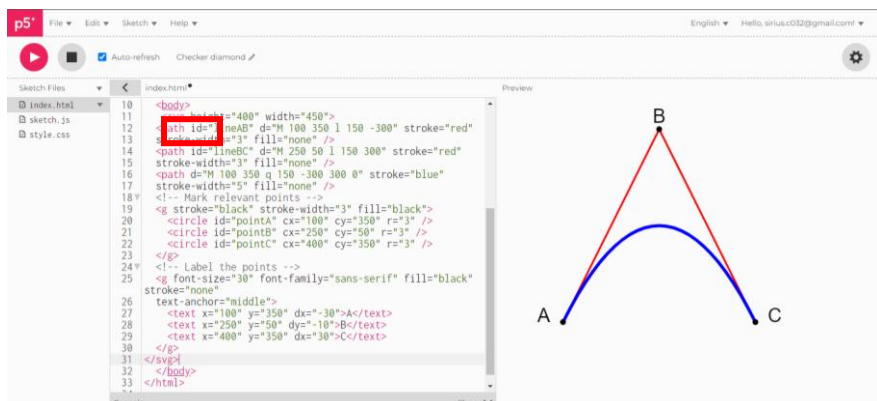


Figura 2.10. Fereastra de lucru a editorului p5.js

2.5 Software gratuit pentru grafica vectorială

Inkscape este un software de editare grafică pentru Windows, Mac OS X și Linux, este gratuit și open source.

Gravit Designer Online este o aplicație online gratuită pentru design și editare vectorială pentru Mac, Win, Linux, Chrome OS, Browser. Aceasta permite editarea vectorială, online, direct în browser.

SVG Viewer este un instrument software care permite vizualizarea, editarea sau manipularea online a fișierelor SVG (Scalable Vector Graphics). SVG Viewer este accesibil pe adresa <https://www.svgviewer.dev>, este un software de editare grafică gratuit, folosit pentru crearea graficii vectoriale ușor și intuitiv. Este simplu și potrivit atât pentru web cât și pentru platforme desktop. Este un editor vectorial gratuit, poate fi rulat pe un computer personal sau browser, este folosit pentru crea vectorilor și altor elemente grafice. Este o multiplatformă ușor de utilizat pe Mac, Win, Linux, Chrome OS, Browser.

Vectr este un instrument utilizat pe scară largă pentru adnotarea, editarea imaginilor și desenarea machetelor și diagramelor. Programul permite crearea desenelor vectoriale scalabile utilizând funcții simple și clare. Este ușor de învățat, perfect pentru cei care abia încep să se familiarizeze cu grafica vectorială și au suficiente capacități de bază de ilustrare.

Software-ul Vectr dispune de un număr mare de tutoriale interactive. Vectr permite crearea și editarea graficii vectoriale utilizând instrumentele de pe masa de lucru. Există multe ghiduri video pe Youtube <https://youtu.be/CnSRzM91FYY>.

Fereastra de lucru a pachetului Vectr este arătată în figura 2.11.

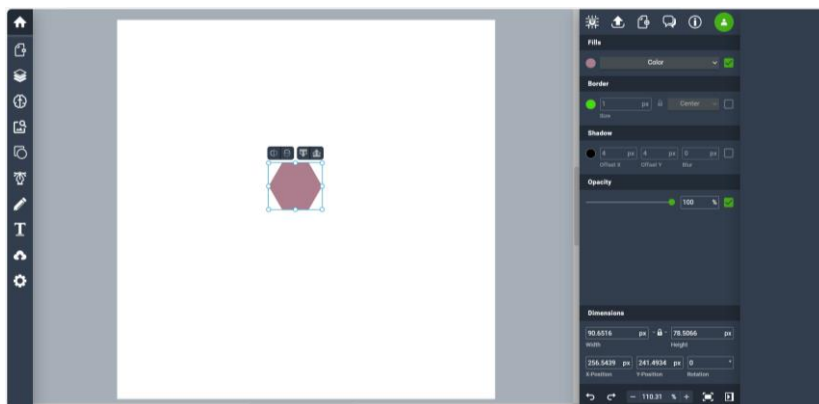


Figura 2.11. Fereastra de lucru a editorului Vectr

Pentru începerea lucrului cu editorul online Vectr, se încarcă o imagine în Vectr. Este permisă importarea fișierelor în formate EPS, AI, SVG, PNG sau JPEG. Interfața Grafică Utilizator este organizată convenabil. Filele „Pagini” și „Straturi” din colțul din stânga sus oferă control asupra procesului de crearea proiectului. Panoul de filtrare este situat în partea dreaptă a interfeței. Aplicația oferă opțiuni de schimbare a unghiului, adăugarea umbrelor, marginilor sau diferitor efecte.

Odată cu finalizarea fazei de proiectare, programul oferă opțiunea de a exporta desene în formatul PNG, JPEG sau SVG. SVG

este singurul format vectorial scalabil care poate fi deschis în alte aplicații, în timp ce hărțile de biți PNG și JPEG sunt mai potrivite pentru web. Aplicația oferă informații utile și sfaturi practice în secțiunea de instruire. Dezvoltatorii editorului Vectr au oferit utilizatorilor o serie de instrucțiuni și tutoriale pentru rezolvarea celor mai populare sarcini de design: crearea pictogramelor, logo-urilor, tipografiilor, meniurilor, colajelor, infograficile și multe altele.

Lucrare de laborator nr.2

Tema: Generarea imaginilor vectoriale

Obiectivele lucrării:

1. **Înțelegerea conceptului de grafică vectorială:** familiarizarea cu principiile de bază ale graficii vectoriale și diferențele dintre grafica vectorială și grafica raster.

2. **Utilizarea primitivelor grafice SVG:** aplicarea primitivelor grafice de bază SVG, cum ar fi: <rect>, <circle>, <ellipse>, <line>, <polyline>, etc.

3. **Manipularea atributelor SVG:** înțelegerea și aplicarea atributelor corespunzătoare pentru a personaliza aspectul elementelor grafice cum ar fi –fill, stroke, stroke-width, opacity, transform, și altele.

4. **Crearea scene grafice vectoriale 2D:** proiectarea și generarea unei scenei vectoriale 2D complexe folosind primitivele menționate, pentru a le organiza într-o scenă coerentă.

5. **Exportul și vizualizarea imaginilor SVG:** înțelegerea procesului de export al imaginilor SVG și vizualizarea acestora în diverse aplicații sau browsere pentru a verifica corectitudinea și calitatea rezultatelor.

6. **Obținerea abilităților practice în integrarea elementelor grafice:** combinarea mai multor forme și elemente grafice pentru a construi o scenă completă, dezvoltând abilități în crearea unor imagini vectoriale mai complexe și atractive.

Numărul de ore necesare pentru realizare – 4 ore academice.

Scopul lucrării: obținerea cunoștințelor practice în sinteza și generarea scenelor grafice vectoriale 2D, utilizând primitivele grafice simple în formatul SVG. Explorarea conceptelor esențiale precum definirea formelor geometrice și utilizarea atributelor pentru personalizarea acestora.

Sarcina lucrării:

Creați o scenă grafică vectorială 2D utilizând primitivele grafice necesare cum ar fi: `<rect>`, `<circle>`, `<ellipse>`, `<line>`, `<polyline>`, `<polygon>`, `<path>`, împreună cu attributele corespunzătoare. Scena trebuie să conțină un element `<text>` plasat în colțul de jos, în partea dreaptă a ecranului, care ar indica numele, prenumele și grupa studentului. Elaborați versiunea vectorială a personajul desenat în lucrarea de laborator nr.1. Variantele sunt indicate în **Tabelul 1.3**.

Pentru reproducerea formelor sau curbilor complexe este permisă utilizarea editoarelor grafice vectoriale.

Exemplu: Crearea imaginii vectoriale bidimensionale care desenează o buburuză.

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/p5.js
/1.9.1/p5.js"></script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/p5.js
/1.9.1/addons/p5.sound.min.js"></script>
    <link rel="stylesheet" type="text/css"
href="style.css">
    <meta charset="utf-8" />
```



```

<style>
  svg {
    filter: drop-shadow(20px 10px 5px
rgba(0,0,0,0.66));
  }
</style>
</head>
<body>
  <svg width="288" height="321" viewBox="0 0 288
321" fill="none" >
    <linearGradient id="grad1" x1="0%" y1="0%"
x2="100%" y2="0%">
      <stop offset="0%" style="stop-
color:rgb(255,255,255);stop-opacity:1" />
      <stop offset="100%" style="stop-
color:rgb(201,16,16);stop-opacity:5" />
    </linearGradient>
    <path d="M235.806 221.399C199.733 279.997
132.807 303.604 86.4958 275.096C40.185 246.587
31.1209 176.2 67.1938 117.602C103.267 59.0032
170.193 35.3956 216.504 63.9043C262.815 92.413
271.879 162.8 235.806 221.399Z"
fill="url(#grad1)" stroke="black" stroke-
width="3"/>
    <ellipse cx="158" cy="250.5" rx="18" ry="15.5"
fill="black"/>
    <ellipse cx="97.5" cy="213.5" rx="13.5" ry="11.5"
fill="black"/>
    <circle cx="213" cy="181" r="18" fill="black"/>
    <circle cx="213" cy="181" r="18" fill="black"/>
    <circle cx="134.5" cy="133.5" r="22.5"
fill="black"/>
    <circle cx="134.5" cy="133.5" r="22.5"
fill="black"/>

```

```

<line x1="167.328" y1="280.303" x2="188.328"
y2="320.303" stroke="black" stroke-width="3"/>
<line x1="44.6949" y1="203.329" x2="0.694881"
y2="226.329" stroke="black" stroke-width="3"/>
<line x1="50.6949" y1="150.329" x2="6.69488"
y2="173.329" stroke="black" stroke-width="3"/>
<line x1="84.2833" y1="93.473" x2="6.28327"
y2="108.473" stroke="black" stroke-width="3"/>
<line x1="254.222" y1="179.131" x2="286.222"
y2="224.131" stroke="black" stroke-width="3"/>
<line x1="223.328" y1="234.303" x2="244.328"
y2="274.303" stroke="black" stroke-width="3"/>
<ellipse cx="212.5" cy="62.5" rx="54.5" ry="41.5"
fill="#1E1E1E"/>
<ellipse cx="191" cy="50.5" rx="10" ry="11.5"
fill="white"/>
<ellipse cx="228" cy="73.5" rx="10" ry="11.5"
fill="white"/>
<ellipse cx="191.5" cy="50.5" rx="3.5" ry="5.5"
fill="black"/>
<ellipse cx="227.5" cy="73.5" rx="3.5" ry="5.5"
fill="black"/>
<ellipse cx="235.5" cy="26" rx="17.5" ry="13"
fill="#1E1E1E"/>
<line x1="227" y1="16" x2="227" stroke="black"
stroke-width="2"/>
<line x1="245.211" y1="25.3861" x2="259.211"
y2="7.38606" stroke="black" stroke-width="2"/>
<line x1="91.5791" y1="274.73" x2="175.579"
y2="143.73" stroke="#1E1E1E"/>
</svg>
</body>
</html>

```

Rezultatul rulării parogramului este arătat în figura 2.12, pentru colorarea personajului a fost utilizat gradientul și pentru crearea imaginii mai complexe este utilizată umbrirea.

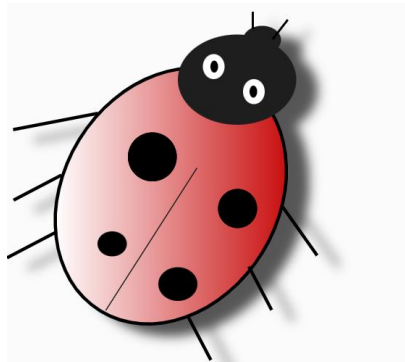


Figura 2.12. Rezultarul execuției programului

Criterii de evaluare:

1. Corectitudinea codului și a implementării (30%): evaluarea corectitudinii utilizării primitivelor grafice SVG, cum ar fi <rect>, <circle>, <ellipse>, <line>, <polyline>, <polygon>, și <path>. Verificarea dacă toate primitivele necesare au fost folosite corect complet, și conform cerințelor lucrării. Dacă elementele grafice sunt corect desenate și aranjate pentru formarea scenei dorite.

2. Utilizarea corectă a atributelor SVG (20%): se evaluează aplicarea corectă a atributelor SVG, cum ar fi fill, stroke, stroke-width, opacity, transform și altele, pentru personalizarea elementelor grafice. Se verifică dacă modificările stilistice sunt aplicate corespunzător fiecărei forme, respectând cerințele de design.

3. Creativitatea și complexitatea compoziției (15%) – măsurarea gradului de creativitate și complexitate a imaginii generate, inclusiv modul în care primitivele sunt combinate pentru a crea o compoziție interesantă. Se apreciază gradul de originalitate și

complexitate al lucrării, adică modul în care studentul adaugă detalii suplimentare pentru a crea o scenă mai interesantă.

4. Estetica și organizarea scenei (10%): se evaluează aspectul vizual al scenei, inclusiv aranjamentul și compunerea elementelor grafice. Detaliile vizuale sunt adecvate scopului propus și demonstrează o înțelegere a principiilor de desenare.

5. Evaluarea cunoștințelor (20%): explicațiile oferite despre procesul de realizare a lucrării, ceea ce poate include descrierea funcțiilor principale și a logicii utilizate.

6. Respectarea termenului de predare (5%): se va lua în considerare respectarea termenului limită stabilit de profesor pentru predarea lucrării.

Întrebări de autoevaluare a cunoștințelor:

1. Ce reprezintă SVG și cum diferă grafica vectorială de grafica raster?
2. Enumerați primitivele grafice utilizate în SVG și explicați la ce folosește fiecare.
3. Cum este definit un dreptunghi în SVG și care sunt principalele atribute necesare?
4. Explicați rolul atributelor **fill**, **stroke** și **stroke-width**. Cum afectează acestea aspectul primitivei grafice?
5. Ce efect are atributul **opacity** asupra unui element SVG?
6. Cum pot fi modificate dimensiunile unei primitive SVG, cum ar fi un cerc sau dreptunghi, folosind atributele disponibile?
7. Ce avantaje și dezavantaje ale utilizării graficii vectoriale sunt în comparație cu grafica raster?
8. Cum pot fi folosite funcțiile **polyline** și **polygon** pentru a crea forme complexe în SVG?

