1.

Adaug varfurile, mai intai aplic o translatie, apoi scalarea, urmand sa reprezint poligoanele si dreptunghiul.

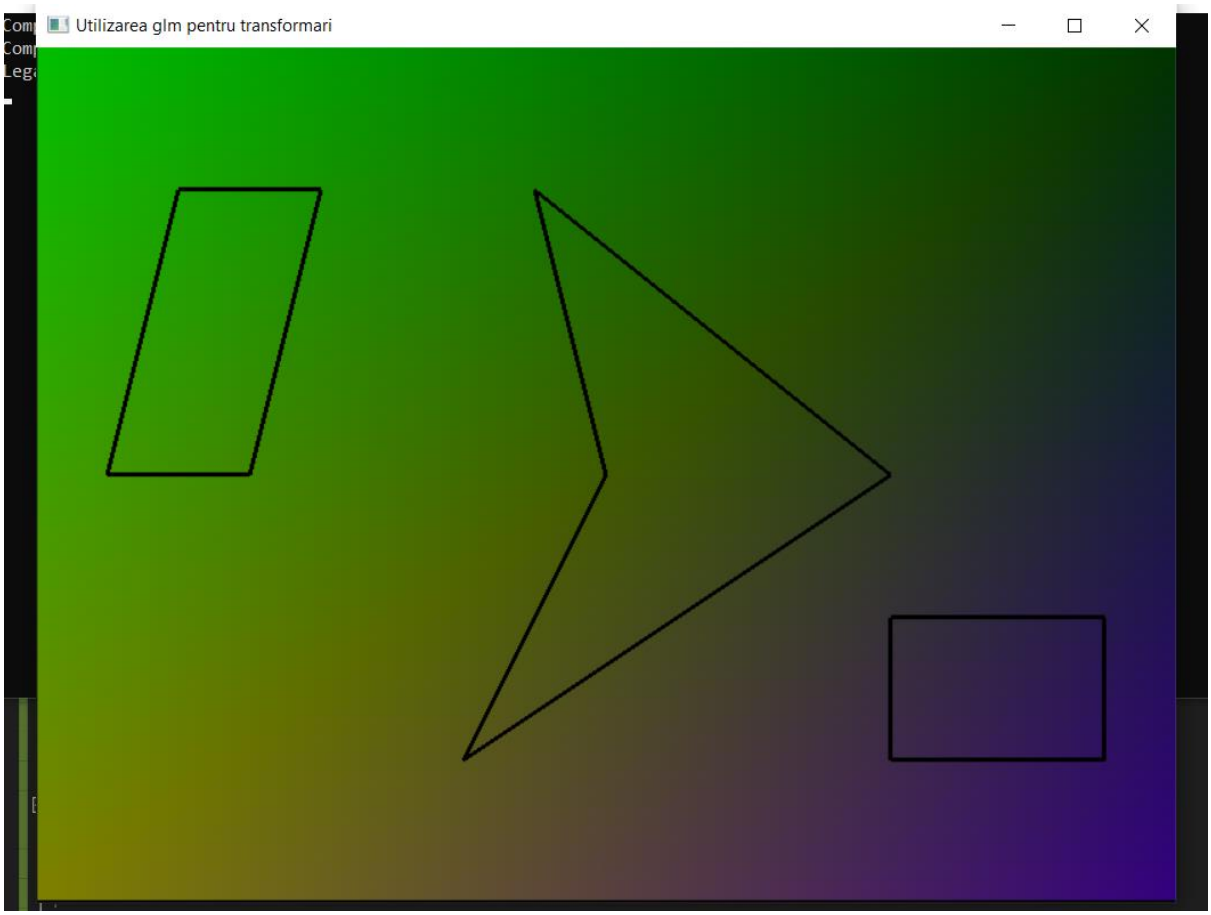
Cod:

```
77          //varfuri pentru p concav
78          300.0f,  100.0f, 0.0f, 1.0f,
79          600.0f,  300.0f, 0.0f, 1.0f,
80          350.0f, 500.0f, 0.0f, 1.0f,
81          400.0f,  300.0f, 0.0f, 1.0f,
82
83          //vf poligon CONVEX
84          50.0f,  300.0f, 0.0f, 1.0f,
85          150.0f,  300.0f, 0.0f, 1.0f,
86          200.0f, 500.0f, 0.0f, 1.0f,
87          100.0f,  500.0f, 0.0f, 1.0f,
88
89          // vf pentru dreptunghi
90          600.0f,  100.0f, 0.0f, 1.0f,
91          750.0f,  100.0f, 0.0f, 1.0f,
92          750.0f,  200.0f, 0.0f, 1.0f,
93          600.0f, 200.0f, 0.0f, 1.0f,
94
```

```
170     //vom aplica o translatie inainte sa facem scalarea
171     matrDeplasare = glm::translate(glm::mat4(1.0f), glm::vec3(-400.f, -300.f, 0.0));
172     //scalare
173     resizeMatrix = glm::scale(glm::mat4(1.0f), glm::vec3(1.f / width, 1.f / height, 1.0));
174
```

```
226
227         //matrice redimensionare
228         myMatrix = resizeMatrix *matrDeplasare;
229         myMatrixLocation = glGetUniformLocation(ProgramId, "myMatrix");
230         glUniformMatrix4fv(myMatrixLocation, 1, GL_FALSE, &myMatrix[0][0]);
231         // desenare puncte din colturi si axe
232         glPointSize(10.0);
233         glDrawArrays(GL_POLYGON, 0, 4); // pentru gradient
234         // pt axe
235         glDrawArrays(GL_LINES, 4, 4);
236
237
238         //---- poligon convex
239         glLineWidth(3.0);
240         glDrawArrays(GL_LINE_LOOP, 12, 4);
241
242         // -- poligon concav
243         glDrawArrays(GL_LINE_LOOP, 8, 4);
244         // -- dreptunghi
245         glDrawArrays(GL_LINE_LOOP, 16, 4);
```

Rezultat:



2. Gradient: - am adaugat cele 4 colturi si le-am dat culori diferite pentru a face un gradient

```
66      // cele 4 varfuri din colturi, folosim si pentru gradient
67    0.0f, 0.0f, 0.0f, 1.0f,
68     0.0f,  600.0f, 0.0f, 1.0f,
69     800.0f, 600.0f, 0.0f, 1.0f,
70     800.0f, 0.0f, 0.0f, 1.0f,
```

```
102     // culorile varfurilor din colturi
103     GLfloat Colors[] = {
104       0.5f, 0.5f, 0.0f, 1.0f,
105       0.0f, 0.75f, 0.0f, 1.0f,
106       0.0f, 0.2f, 0.0f, 1.0f,
107       0.2f, 0.0f, 0.5f, 1.0f,
108     };
109
```

```
233     glDrawArrays(GL_QUADS, 0, 4); // pentru gradient
```

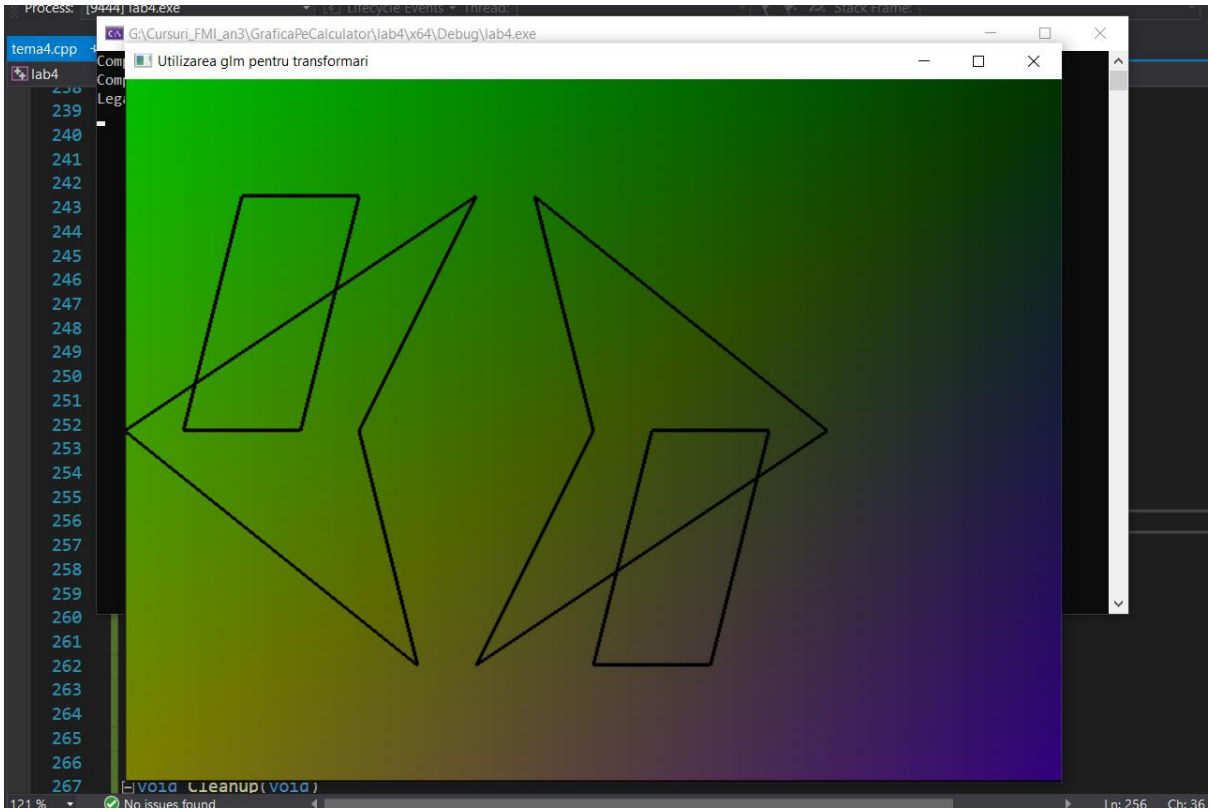
3.

Cod: pentru a face rotatia fata de un punct diferit de origine, mai intai am aplicat o translatie, apoi rotatia de unghi pi, iar apoi o translatie pentru a reveni dupa translatia intiala.

```
183     //-- aplicare rotatie
184     matrRot = glm::rotate(glm::mat4(1.0f), PI , glm::vec3(0.0, 0.0, 1.0));
185     //-- alegem punctul P( 300, 300) care se afla intre poligoane
186     matrTransl1 = glm::translate(glm::mat4(1.0f), glm::vec3(-300.f, -300.f, 0.0));
187     matrTransl2 = glm::translate(glm::mat4(1.0f), glm::vec3(300.f, 300.f, 0.0));
188
```

```
250     // -------------------------- poligoanele rotite
251     //-- matrice de stabilire a pozitiei
252     myMatrix = resizeMatrix * matrDeplasare* matrTransl2 * matrRot * matrTransl1;
253     myMatrixLocation = glGetUniformLocation(ProgramId, "myMatrix");
254     glUniformMatrix4fv(myMatrixLocation, 1, GL_FALSE, &myMatrix[0][0]);
255     // -- poligon convex
256     glDrawArrays(GL_LINE_LOOP, 12, 4);
257     // -- poligon concav
258     glDrawArrays(GL_LINE_LOOP, 8, 4);
259
```

Rezultat:



4.

Cod: Am definit matricea de scalare si apoi am inmultit-o pentru a realiza scalarea.

```
257     /*dreptunghi --> aplicare scalare*/
258     myMatrix = resizeMatrix * matrDeplasare*matrScale;
259     myMatrixLocation = glGetUniformLocation(ProgramId, "myMatrix");
260     glUniformMatrix4fv(myMatrixLocation, 1, GL_FALSE, &myMatrix[0][0]);
261     glDrawArrays(GL_LINE_LOOP, 16, 4);
262
```

```
192
193     matrScale = glm::scale(glm::mat4(1.0f), glm::vec3(0.5, 0.5f, 1.0)); // " mareste de a lungul axelor dif"
194
```

Rezultat: