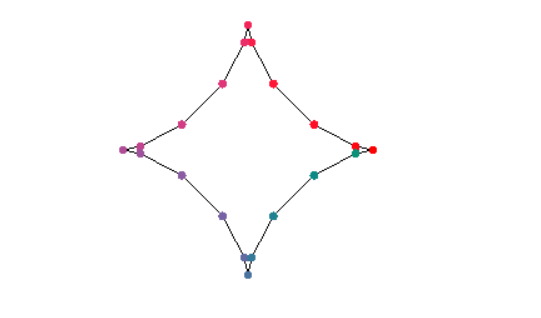
Băicoianu Bianca

Grupa 351

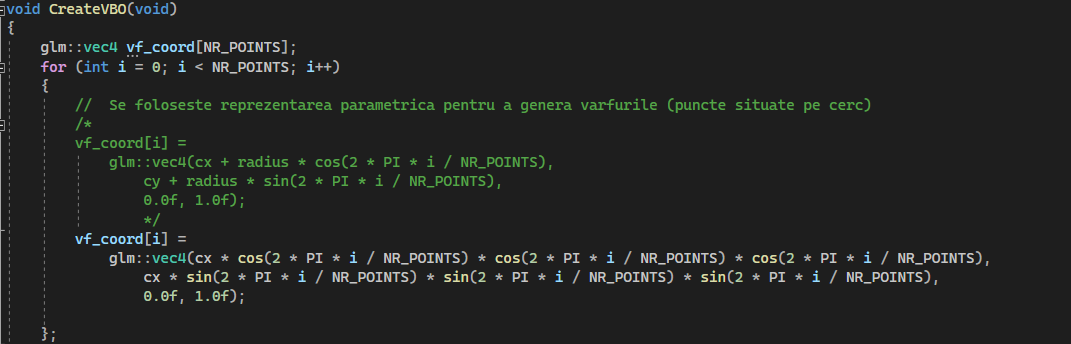
Laborator 8

**Exercițiu 1** - *prezentat laborator*

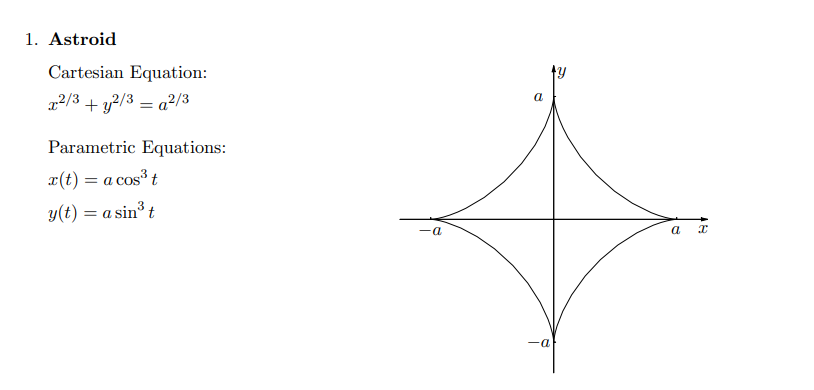
Rezultat:



Cod:

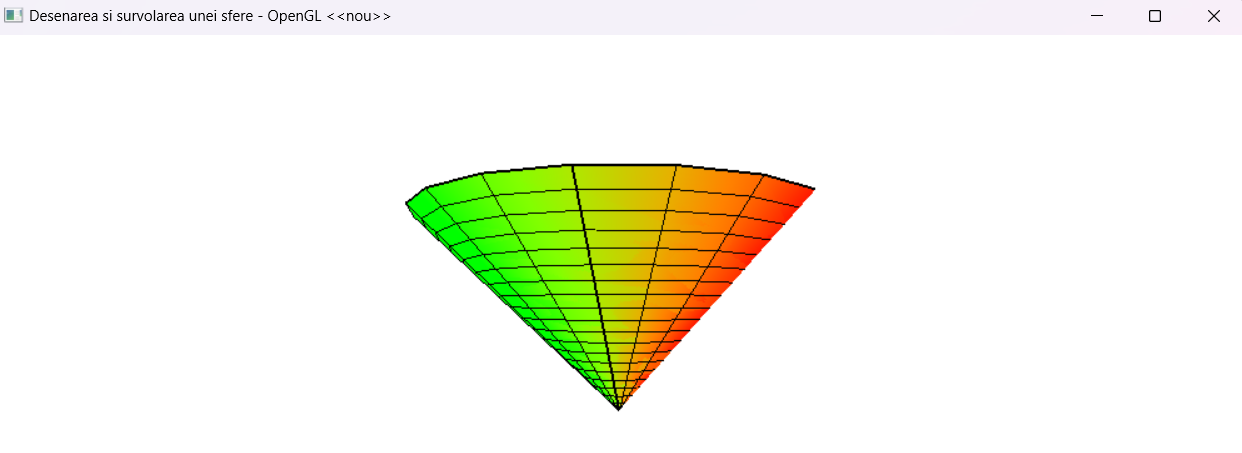


* pentru generarea varfurilor (de aceasta dată punctele vor fi dispuse ca astroidă), m-am folosit de ecuațiile parametrice următoare:



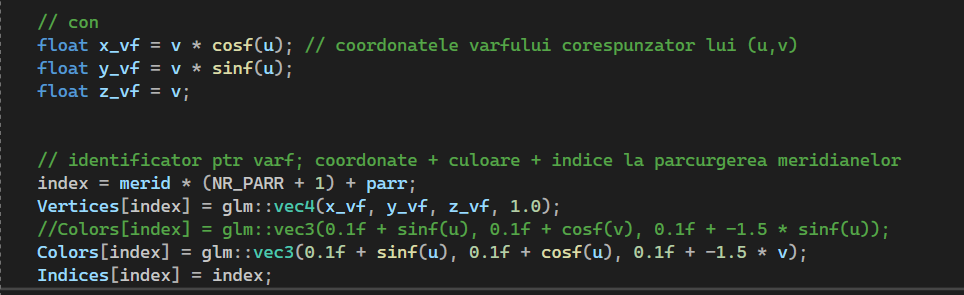
**Exercițiu 2 -** *prezentat laborator*

Rezultat:



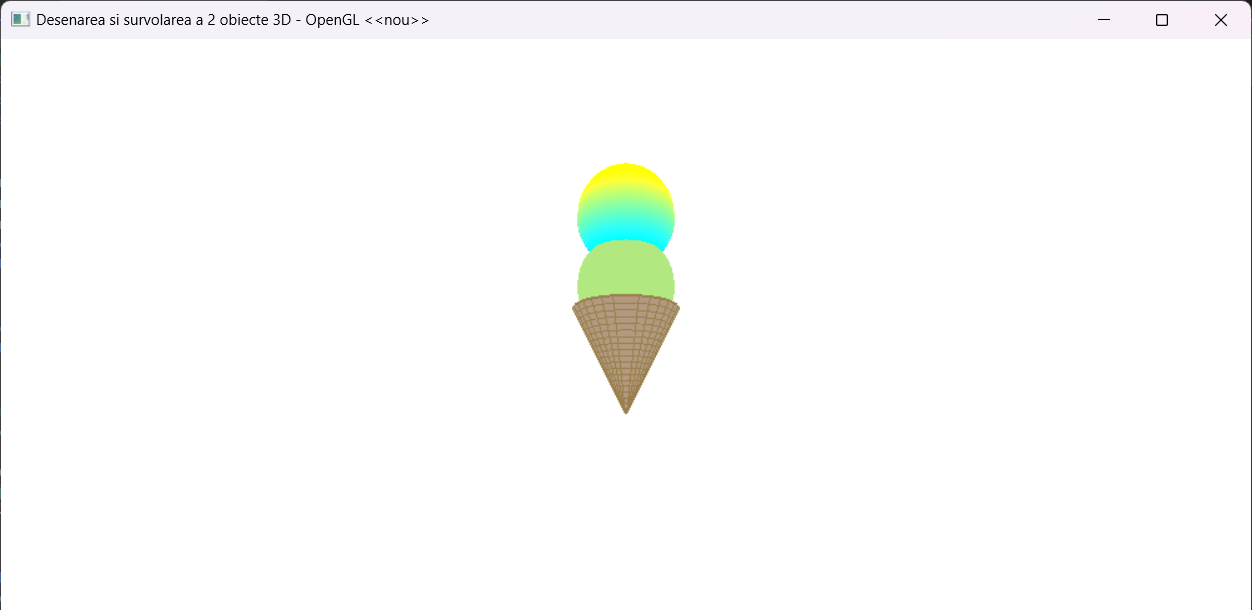
Cod:

* în create Vbo am modificat x\_vf, y\_vf, z\_vf (pentru a obține un con) și apoi în Colors[index]

****

**Exercițiul 3 - con cu 2 cupe de înghețată**

Rezultat



Cod

* pentru realizarea **conului de înghețată**, în funcția CreateVAO2: am definit x\_vf, y\_vf, z\_vf (in prealabil am declarat U\_MIN = 0, U\_MAX = 2\*PI

| **float** x\_vf = v \* cosf(u); // coordonatele varfului corespunzator lui (u,v) **float** y\_vf = v \* sinf(u); **float** z\_vf = v; |
| --- |

* vom defini culoarea conului

Colors2[index] = glm::vec3(0.7f, 0.6f, 0.5f);

* apelam funcția CreateVAO2 în Initialize()
* în funcția RenderFunction, desenam conul și meridianele/paralelele pentru a reda modelul de pe conul de înghețată (am definit codCol = 2 pentru striațiile de pe con ca să completeze culoarea conului)

| **void** RenderFunction(**void**) {  ......  // CON  glBindVertexArray(VaoId2);  codCol = 0;  glUniform1i(codColLocation, codCol);  for (**int** patr = 0; patr < (NR\_PARR + 1) \* NR\_MERID; patr++)  {  if ((patr + 1) % (NR\_PARR + 1) != 0)  glDrawElements(  GL\_QUADS,  4,  GL\_UNSIGNED\_SHORT,  (GLvoid\*)((2 \* (NR\_PARR + 1) \* (NR\_MERID)+4 \* patr) \* **sizeof**(GLushort)));  }  // desenarea muchiilor - meridiane/paralele  codCol = 2;  glUniform1i(codColLocation, codCol);  glLineWidth(2.0);  // fiecare meridian separat  for (**int** merid = 0; merid < NR\_MERID; merid++)  {  glDrawElements(  GL\_LINE\_STRIP,  NR\_PARR + 1,  GL\_UNSIGNED\_SHORT,  (GLvoid\*)((merid \* NR\_PARR + merid) \* **sizeof**(GLushort)));  }  // fiecare cerc paralel separat  for (**int** parr = 1; parr < NR\_PARR; parr++)  {  glDrawElements(  GL\_LINE\_LOOP,  NR\_MERID,  GL\_UNSIGNED\_SHORT,  (GLvoid\*)(((NR\_PARR + 1) \* (NR\_MERID)+parr \* NR\_MERID) \* **sizeof**(GLushort)));  } .......... } |
| --- |

* pentru **cupele de înghețată,** am creat funcțiile CreateVAO1 și CreateVAO3

în aceste funcții am definit

| **float** x\_vf = radius \* cosf(u) \* cosf(v);  **float** y\_vf = radius \* cosf(u) \* sinf(v); **float** z\_vf = (radius \* sinf(u) + 130); |
| --- |

* pentru culori am folosit:

| Colors1[index] = glm::vec3(0.7f, 0.91f, 0.5f); //în CreateVAO1 |
| --- |

și

| Colors3[index] = glm::vec3(0.7f + sinf(u), 0.9f + cosf(v), 0.5f + -0.9 \* sinf(u)); //în CreateVAO3 |
| --- |

\* am vrut sa redau efectul de topping pe deasupra cupei

* apelăm funcțiile în Initialize()
* în RenderFunction:

| **void** RenderFunction(**void**) {  ….  // SFERA1  glBindVertexArray(VaoId3);  codCol = 0;  glUniform1i(codColLocation, codCol);  for (**int** patr = 0; patr < (NR\_PARR + 1) \* NR\_MERID; patr++)  {  if ((patr + 1) % (NR\_PARR + 1) != 0)  glDrawElements(  GL\_QUADS,  4,  GL\_UNSIGNED\_SHORT,  (GLvoid\*)((2 \* (NR\_PARR + 1) \* (NR\_MERID)+4 \* patr) \* **sizeof**(GLushort)));  }   // idem SFERA2   } |
| --- |