**FACULTATEA: Automatica si Calculatoare  
SPECIALIZAREA: Calculatoare si Tehnologia Informatiei  
DISCIPLINA: Prelucrare grafică  
PROIECT: CASTLE OpenGL Scene**

**Prof. coordonator:**

**prof.ing. Nandra Cosmin**

**Student:**

**Sand Elena-Andreea**

**Grupa: 30233**

**AN UNIVERSITAR**

**2022- 2023**

**Cuprins**

[**1. Prezentarea temei alese** 3](#_Toc64285969)

[**2. Scenariul** 3](#_Toc64285970)

[**2.1 Descrierea scenei si a obiectelor** 3](#_Toc64285971)

[**2.2 Functionalitati** 5](#_Toc64285972)

[**3. Detalii de implementare** 6](#_Toc64285973)

[**3.1 Functii si algoritmi** 6](#_Toc64285974)

[**3.2 Modelul grafic** 9](#_Toc64285975)

[**3.3 Structuri de date** 10](#_Toc64285976)

[**3.4 Ierarhia de clase** 10](#_Toc64285977)

[**4. Prezentarea interfeței grafice utilizator / manual de utilizare** 11](#_Toc64285978)

[**5. Concluzii și dezvoltări ulterioare** 12](#_Toc64285979)

[**6. Referințe** 13](#_Toc64285980)

**1. Prezentarea temei alese**

Tema acestui proiect consa in realizarea unei scene 3D, cat mai realista si apropiata de detaliile

vietii de zi cu zi. Scena pe care am ales sa o modelez cuprinde un castelcu diferite personaje Disney. Am urmarit calitatea texturarii imaginilor, aranjarea obiectelor in scena precum si calitatea randarii.

Complexitatea scenei a depins foarte mult atat de obiectele texturate pe care le contine, cat si de animatiile care ii confera un caracter interactive.

Folosind librariile OpenGL, GLFW si GLM, am reusit sa realizez urmatoarele aspecte:

* Intrarea in scena cu ajutorul unei animatii de prezentare;
* Vizualizarea scenei: scalare, translatie, rotatie, miscarea camerei, atat de la tastatura, cat si cu ajutorul mausului;
* Vizualizarea scenei in modurile solid, wireframe, poligonal si smooth;
* Utilizarea luminii directionale pentru a lumina toata scena, precum si a unei lumini de tip spotlight, pentru a crea o atmosfera de seara in cadrul desertului;
* Maparea texturilor si definirea materialelor pentru un aspect cat mai placut al obiectelor
* Generarea umbrelor si a unui cub de lumina ce simuleaza miscarea soarelui de-a lungul unei zile (am incercat sa fac acest lucru cat mai realistic posibil,)
* Fotorealism, prin introducerea efectului de ploaie si ceata in scena, si, de asemenea, prin animarea diferitelor obiecte.

Scopul acestui proiect a fost invatarea principiilor din OpenGL si familiarizarea cu utilizarea

practia a acestora in mediul de programare Visual Studio, utilizand librariile prezentate la laborator (OpenGl, GLFW, GLM, etc.), de asemenea utilizarea mediului de modelare 3D Blender.

Utilizatorul trebuie sa aiba posibilitatea de a controla scena prin intermediul mouse-ului si tastaturii.

# **2. Scenariul**

## **2.1 Descrierea scenei si a obiectelor**

Scena a fost construita cu ajutorul software-ului graphic “Blender” si cu ajutorul unormodele 3Dcu extensia “.obj” preluate de pe diferite site-uri care propun diverse modele 3D. Aceasta este relative simpla, continand un numar mediu de obiecte,prezentand un castel cu diferite personaje Disney:un dinosaur,un caine care se poate deplasa de la tastatura, un soarece, o printesa moderna, un dragon care se roteste tot timpul in jurul castelului, etc.

Pentru a ne apropia cat mai mult de realitate, am incercat sa pastrez cat mai bine posibil dimensionalitatea obiectelor, precum si pozitionarea logica a acestora

## 

## **2.2 Functionalitati**

In cadrul acestei scene exista mai multe posibilitati de navigare: navigarea prin intermediul tastaturiisi prin intermediul mouse-ului.

Au mai fost implementate functionalitati cheie pentru un realism sporit: lumina directionalasi lumina punctiforma, precum si shadere: pentru obiecte, pentru crearea hartii de adancime dupa care vor fi desenate umbrele, pentru SkyBox,etc.

In plus, o alta functionalitate este prezenta animatiilor in scena: rotirea dragonului in jurul castelului si deplasarea cainelui de la tastatura.

Deasemenea, mai exista functionalitati modificabileprin tastatura, precum pornirea cetii sau a ploii, modificarea de la zi la noapte sau vizualizarea scenei in mai multe moduri: in modul solid, wireframesi punctiform.

Utilizatorul se poate deplasa prin scena cu ajutorul mouse-ului si a tastelor **W, A, S, D.**

* Tasta W: **inaintare** in scena.
* Tasta A: deplasare **stanga**.
* Tasta D: deplasare **dreapta**.
* Tasta S: **departare** in scena.
* Tasta Q: rotirea la dreapta
* Tasta E: rotirea la dreapta
* Tasta 9: deplasarea in sus
* Tasta 0: deplasarea in jos

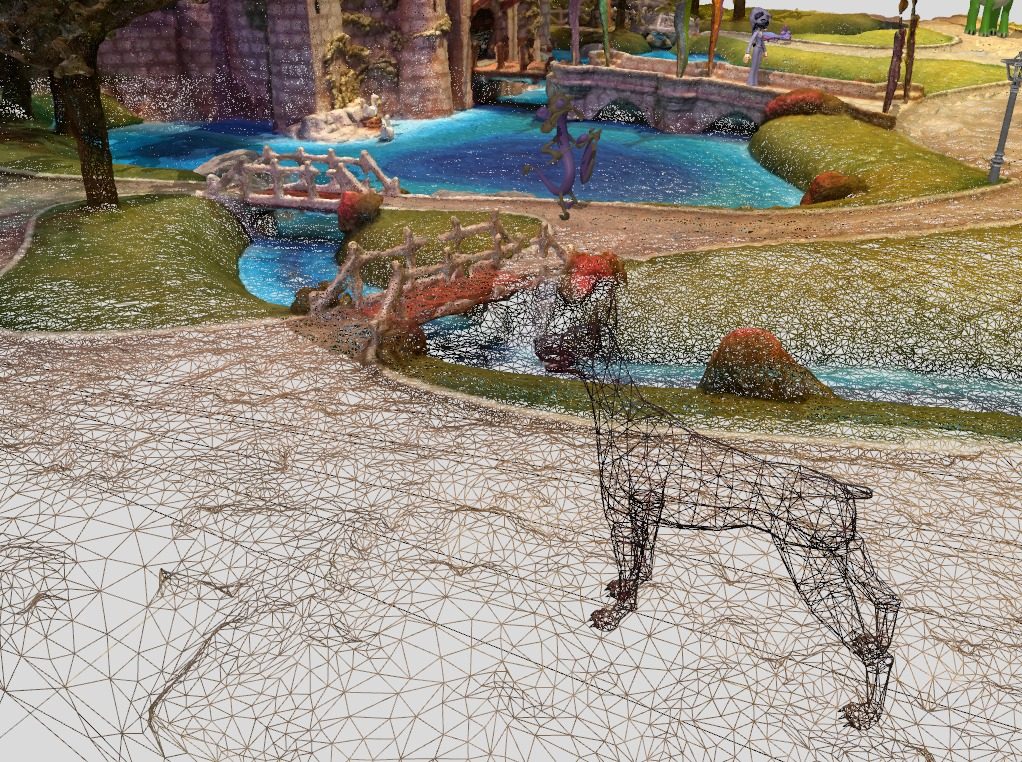
De asemenea, utilizatorul poate roti scena în sus, în jos, la stânga și la dreapta cu ajutorul mausului.

Utilizatorul poate alege sa vizualizeze scena utilizand un alt mod de prezentare. Aceste moduri se pot schimba cu ajutorul tastelor Z, X, C care au urmatorul effect:

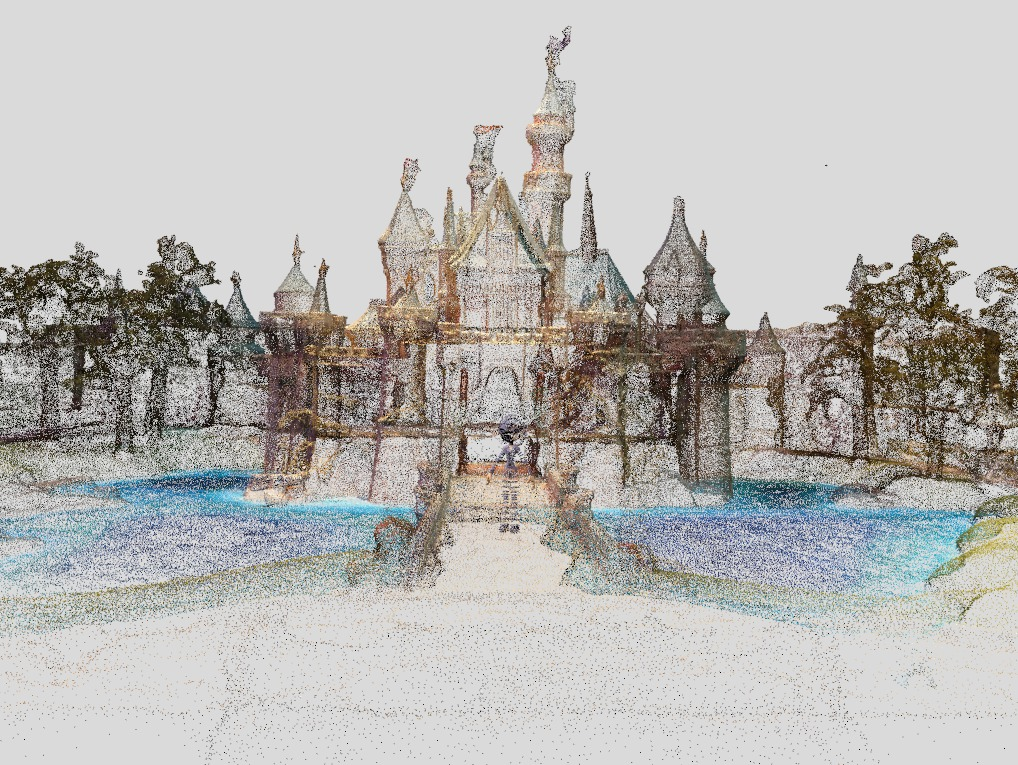
* Tasta Z: vizualizarea scenei in modul de reprezentare **smooth/ plat.**



* Tasta X: vizualizarea scenei in modul de reprezentare **wireframe.**



* Tasta C: vizualizarea scenei in modul de reprezentare **punct.**



De asemenea, utilizatorul poate active ploaia sau ceata.

* Tasta P: activare **poaie**



* Tasta F: activarea **fog**



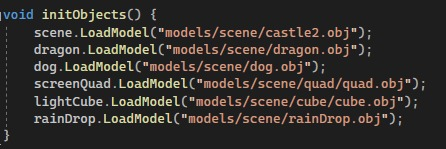
# **3. Detalii de implementare**

## **3.1 Functii si algoritmi**

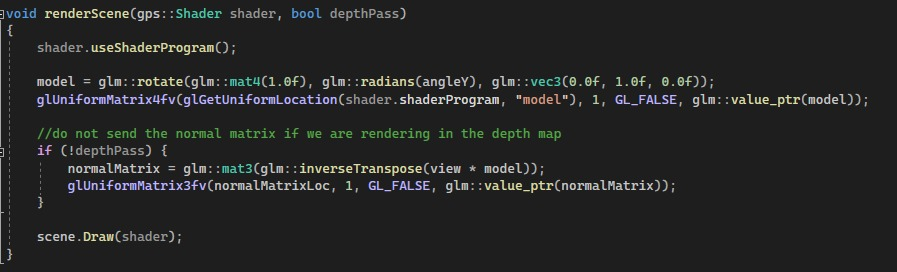
Cel mai important pas in realizarea proiectului a fost reprezentat de **aducerea obiectelor in scena**, prin operatia initiala de incarcare a modelului 3D, urmand ca acesta sa treaca prin 2 etape de rendering: una pentru generarea umbrelor, iar cealalta pentru plasarea in scena.

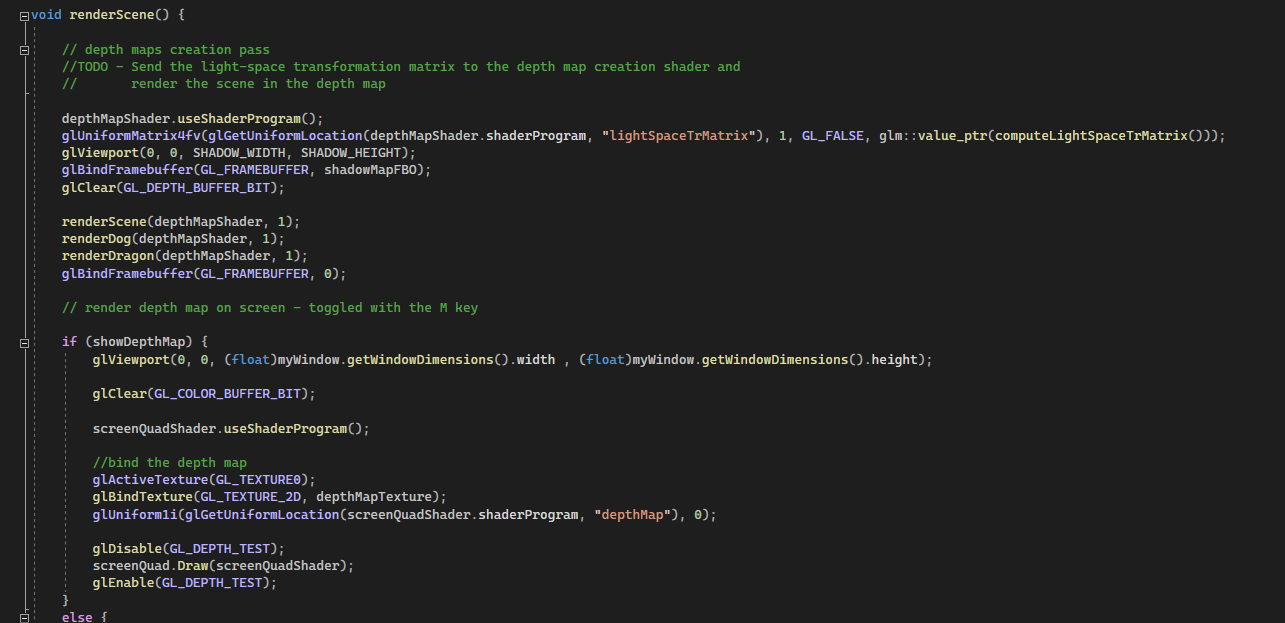
In cod, aceste etape au fost implementate dupa cum urmeaza:

* Incarcarea obiectului:

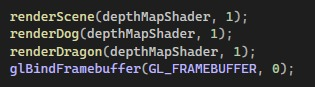


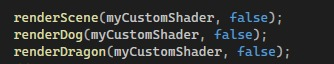
* Functia de rendering a obiectului:





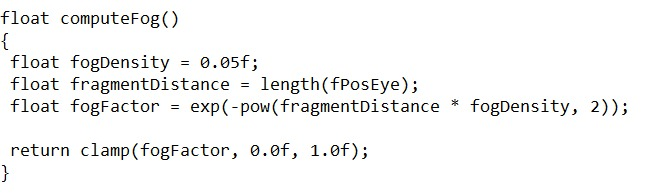
* Apelul functiei de rendering, mai intai pentru a trece prin shader-ul ce contine harta de adancime specifica umbrelor, iar mai apoi prin shaderul obisnuit:



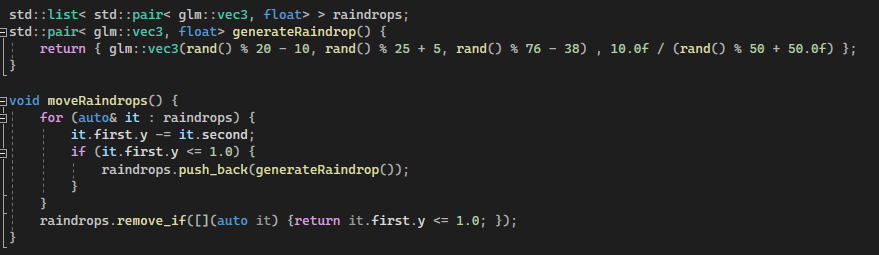


Pentru a realiza o scena cat mai realista, am implementat aparitia **cetii** (activate prin apasarea tastei F) si a **ploii(**activate prin pasarea tastei P)

* Crearea cetii



* Crearea ploii



Se defineste o lista de perechi numite “raindrops”in care fiecare perechecontine un vector 3D (glm::vec3) reprezentand pozitia unei picaturi de ploaie si un float reprezentand viteza picaturii de plaoie (y-ul cu care se scade pozitia initiala).Functia “generateRaindrops()” creeaza o noua pereche cu o pozitie aleatoare si o viteza.Functia moveRaindrops() itereaza prin fiecare pereche din lista “rainDrops”, actualizand pozitia picaturii de ploaie scazand viteza acesteia din coordonata y. Daca coordonata y a picaturii de ploaie scade sub 1.0, o noua picatura de ploaie este generate si adaugata la lista. Lista este actualizata, eliminand picaturile a caror coordonata y este sub 1.0 (adica sub scena).

**Lumina globala si lumina punctiforma**

Pentru cele doua surse de lumina am ales sa implementez o lumina globala directionala si o lumina de tip punctifroma, care provine de la felinar.

Pentru lumina globala, in cadrul fragment shader-ului exista functia “computeLightComponents()”. Pentru aceasta lumina sunt folosite componentele: ambientala, difuza si speculara,

calculate fiecare individual. Acest tip de lumina este calculat tinand cont si de pozitia observatorului(a camerei) care conteaza pentru componenta speculara. Lumina ambientala , nu vine dintr-o anumita directie, este aproximarea luminii globale imprastiate in jurul scenei si reprezinta zona umbrita a obiectului. Lumina difuza este cea imprastiata in toate directiile de o sursa de lumina, intensitatea luminii fiind mai puternica pe fetele orientate direct catre sursa de lumina iar cea speculara este reflectata direct de suprafata si se refera la cat de asemanatoare este suprafata cu o oglinda, deci depinde si de ”reflection”, pentru a imbunatatii efectul de iluminare. Pentru lumina punctiforma, in cadrul fragment shader-ului exista functia: ”CalcPointLight()”. Pentru realizarea acesui tip de lumina a fost nevoie de 3 parametrii (constant, linear si quadratic) preluati din main si transmisi functiei din fragment shader si de un vector care sa specifice pozitia luminilor. Cu cat cei 3 parametrii au o valoare mai mica, cu atat intensitatea luminii va creste. Factorul constant este lasat la valoarea 1, si se modifica numai linear si quadratic, care sunt initializati cu 0.09 si 0.032.

**Miscarea camerei**

Pentru explorarea scenei sunt folosite mouse-ul si tastatura. Rotatia camerei se realizeaza cu ajutorul mouse-ului, iar miscarile la stanga sau la dreapta, sau cele de fata, spate cu ajutorul tastaturii. Pentru rotatia camerei cu ajutorul mouse-ului exista in main functia ”mouseCallback” care preia coordonatele initiale ale mouse-ului, iar mai apoi realizeaza rotatia camerei in functie de miscarile mouse-ului, coordonatele pointer-ului acestuia. Pentru navigarea in scena cu ajutorul tastaturii a fost implementata in ”Camera.cpp” functia ”move” care primeste ca si parametru directia de miscare a camerei si viteza cu care aceasta sa se deplaseze in directia indicata de primul parametru. In interiorul acestei functii, cu ajutorul acestor doi parametrii se vor modifica parametrii camerei: ”cameraPosition” si ”cameraTarget”, in functie de directia in care se doreste deplasarea camerei.

## **3.2 Modelul grafic**

Modelele grafice sunt realizate pe baza modelelor geometrice care descriu forma si geometria obiectelor, insa cuprind si alte elemente ca textura, culoare, umbre, iluminare, necesare imbunatatirii realismului imaginii obiectului. Necesitatea utilizarii si altor elemente pentru model, in afara de modelul geometric, depinde de aplicatia pentru care a fost creat.

Obiectele alese pentru acest proiect au fost preluate de pe internet, urmand ca pozitionarea si texturarea lor sa fie realizate manual, cu ajutorul aplicatiei Blender.

Blender nu doar ca permite importul obiectelor si pozitionarea lor relativa, cat si crearea de modele grafice noi.

Un bun exemplu de obiect realizat 100% in Blender o reprezinta planulpe care e pozitionat castelul. Am creat acest plan pentru ca scena sa arate mai realist.

## **3.3 Structuri de date**

In cadrul proiectului, am utilizat structuri de date disponibile in biblioteca GLM, pentru functiile matematice folosite in cazul transformarilor geometrice, dar si structuri oferite de OpenGL, pentru rendering sau pentru comunicarea intre nivelul aplicatiei si cel al procesorului graphic (comunicarea cu shaderele).

Astfel, in functie de functionalitatea implementata, am utilizat urmatoarele structuri de date si tipuri speciale de variabile

- gps::SkyBox – crearea cubului infinit de jur imprejurul scenei

- gps::Camera – camera prin care se vizualizeaza scena

- GLuint – integer de tip uniform

- std::vector - vector pentru initializarea imaginilor din skybox

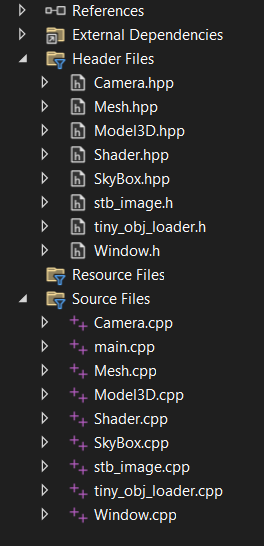
- glm::vec3 , mat 4 – pentru transformari geometrice

- gps::Shader – obiect de tip Shader pentru procesare in pipeline

- gps::Model3D – obiect pentru procesarea modelelor 3D

## **3.4 Ierarhia de clase**

Libraria de clasa este atasata mai jos:



# **4. Prezentarea interfeței grafice utilizator / manual de utilizare**

Controlul camerei se realizeaza cu tastele W,S,A,D,Q,E,9,0 astfel:

* Tasta W: inaintarea in scena;
* Tasta A: deplasare stanga;
* Tasta D: deplasare dreapta;
* Tasta S: departare in scena;
* Tasta Q: rotire dreapta
* Tasta E: rotire stanga
* Tasta 9: deplasare sus
* Tasta 0: deplasarein jos

Utilizatorul poate alege sa vizualizeze scena utilizand alt mod de prezentare. Aceste moduri se pot schimba cu ajutorul tastelor Z,X,C care au urmatorul efect:

* + Tasta Z: vizualizarea scenei in modul de reprezentare smooth;
  + Tasta X: vizualizarea scenei in modul de reprezentare wireframe;
  + Tasta C: vizualizarea scenei in modul de reprezentare polygonal;

De asemenea, directia de deplasare poate fi influentata si de pozitia mouse-ului.

In continuare se va prezenta o legend a tastelor de pare se activeaza animatiile pe diferite

obiecte si efecte:

* + Tasta L: rotire la stanga a cubului de lumina
  + Tasta J: rotire la dreapta a cubului de lumina
  + Tasta F: activarea cetii
  + Tasta G: dezactivarea cetii
  + Tasta N: activare spotlight
  + Tasta B: activare lumina directionala
  + Tasta T: activare turul camerei
  + Tasta R: resetare camera inpozitia initiala, sau oprirea ploii
  + Tasta P: activare ploaie
  + Tasta UP: deplasare caine in fata
  + Tasta DOWN:deplasare caine in spate
  + Tasta M: afisare harta de adancime

# **5. Concluzii si dezvoltari ulterioare**

In urma realizarii acestui proiect, am invatat sa creez o scena interactiva si sa modelez obiecte cu ajutorul software-urilor de proiectare. Am inteles mai aprofundat conceptele explicate la curs.

Proiectul a fost un mod bun s, i creativ de familiarizare cu pricipiile OpenGL, dar s, i cu softwareul grafic 3D ”Blender” ˆın care s-a dezvoltat scena init, ial. Opengl prezinta un pipeline grafic destul de puternic pentru dezvoltarea de aplicatii 2D/3D.

Ca dezvoltari ulterioare s-ar putea implementa intrarea in castel, mai multe animatii, lasarea noptii in tipm,succesiv.

# **6. Referințe**

* <https://learnopengl.com/>
* <https://free3d.com/>
* <https://sketchfab.com/feed?gclid=EAIaIQobChMIivjK_uTJ_AIVCPd3Ch1aBwicEAAYASAAEgLjTPD_BwE&utm_campaign=358341061&utm_content=414556469961&utm_medium=cpc&utm_source=googleads&utm_term=sketch%20fab>