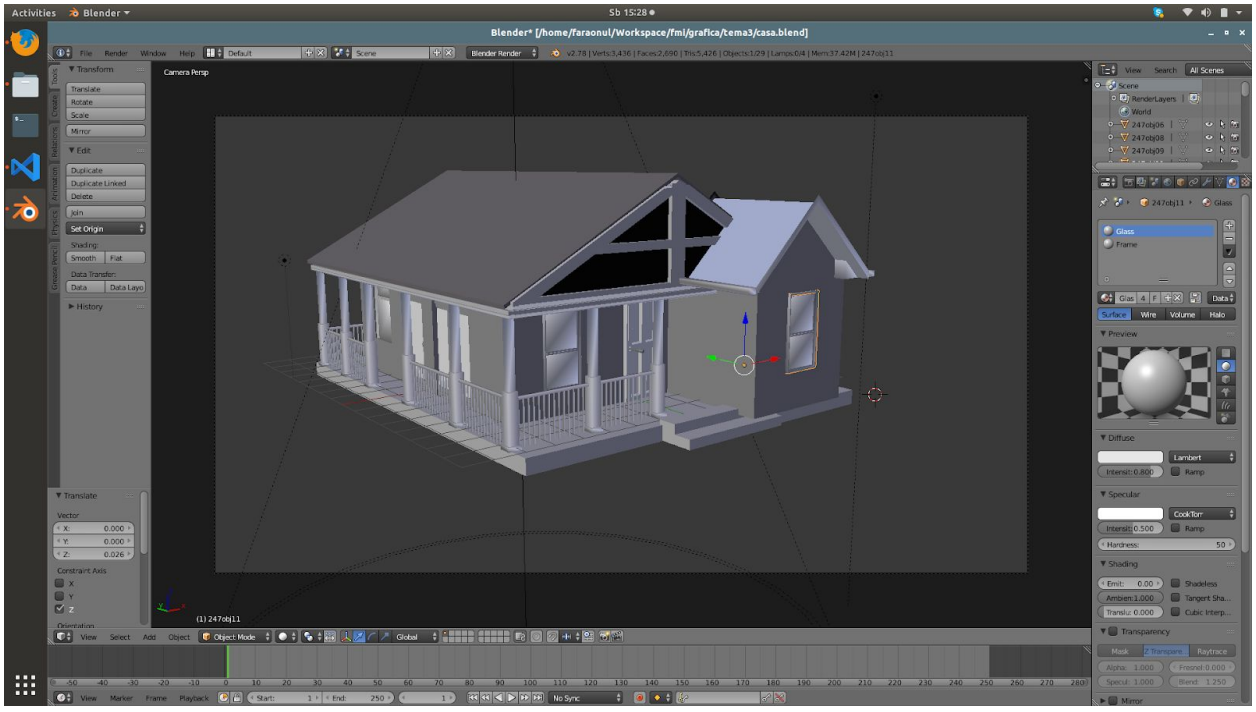
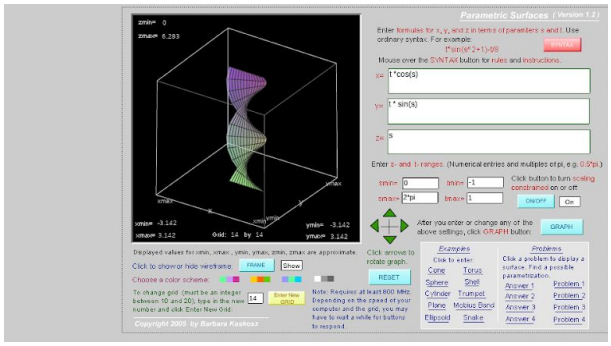
Tema 3 - Dezvoltarea aplicatiilor grafice

1. Utilizati functionalitati de baza ale Blender pentru a crea un model cat mai complex.

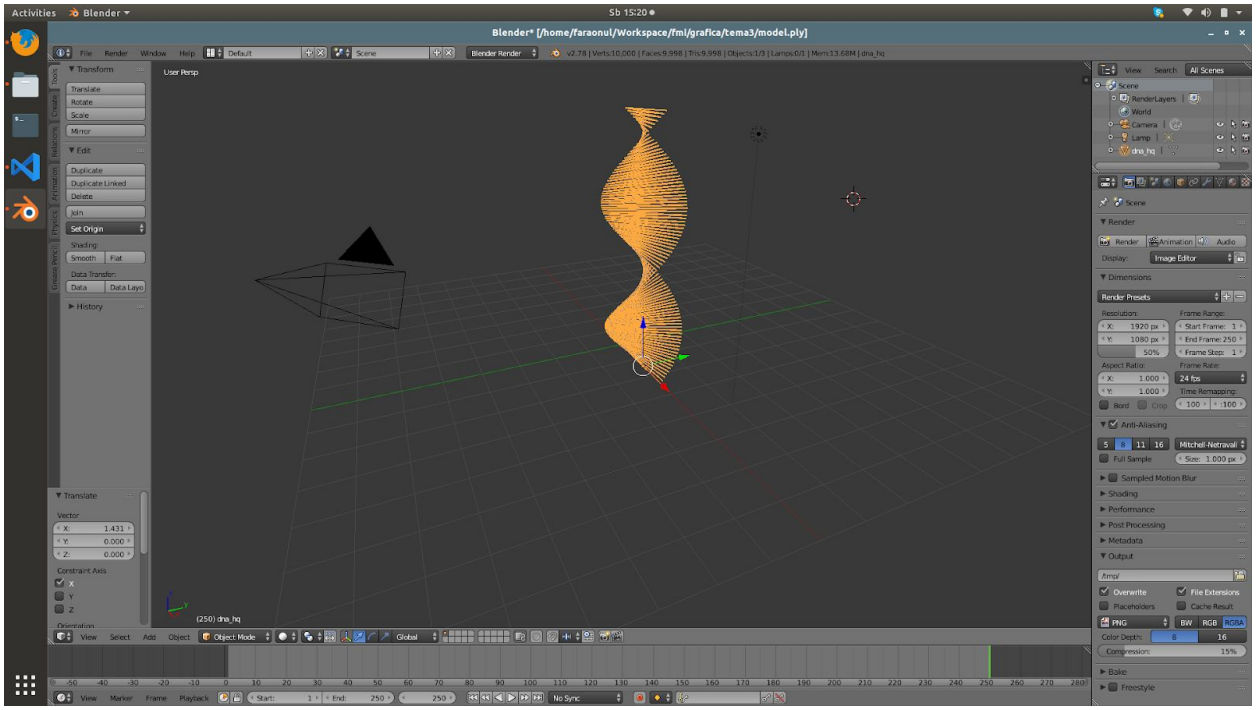


La aceasta tema am ales sa construiesc exteriorul unei case. Modelul a fost realizat din forme de baza: cilindrii, sfere, conuri, cuburi folosind scalari, rotatii. Pentru realizarea clantelor am plecat de la o sfera si am editat nodurile (punctele cheie) din edit mode.

2. Alegeti o suprafata data intr-o forma parametrica explicita si scrieti un program/script care sa exporte un model PLY. Importati acest model in Blender, adaugandu-l modelului creat la (i). Exemple de ecuatii parametrice pe care le puteti folosi: http://www.math.uri.edu/~bkaskosz/flashmo/tools/parsur/ Intocmiti un scurt raport comun pentru (i) si (ii).



La aceasta cerinta ne propunem sa exportam un model similar cu ADN. In forma parametrica avem ecuatiile: x = t * cos(s), y = t * sin(s), z = s.



```
from math import *
import numpy as np

t_min = -1
t_max = 1
s_min = 0
s_max = 2 * pi

polygons = 100
verticies = [
       (t * cos(s), t * sin(s), s)
           for s in np.linspace(s_min, s_max, polygons)
           for t in np.linspace(t_min, t_max, polygons)
   ]
print("ply")
print("format ascii 1.0")
print("element vertex {0}".format(len(verticies)))
print("property float32 x")
print("property float32 y")
print("property float32 z")
print("element face {0}".format(len(verticies) - 2))
print("property list uint8 int32 vertex_index")
print("end_header")
for vertex in verticies:
   print("{0} {1} {2}".format(round(vertex[0], 2), round(vertex[1], 2), round(vertex[2], 2)))
for i in range(len(verticies) - 2):
   print("3 {0} {1} {2}".format(i, i + 1, i + 2))
```

Generarea de fisier PLY o facem luan un numar fixat de puncte (1000) si apoi unim punctele alaturate cu triunghiuri, reprezentand multimea fetelor.

3. Identificati alte formate de date care pot fi utilizate in Blender. Intocmiti o scurta documentatie, exemplificati si implementati.

Blender mai suporta import pentru: Collada (.dae), 3D Studio (.3ds), FBX (.fbx), MotionCapture (.bvh), Wavefront (.obj), X3D (.x3d/.wrl), Stl (.stl) si SVG (.svg).

Collada: Similar cu .PLY, este sub forma de fisier XML care descrie obiectul, si tine referinta si a resurselor folosite de obiect.

```
<geometry id="Cube-mesh">
```

```
    <mesh>
      <source id="Cube-mesh-positions">
        <float_array id="Cube-mesh-positions-array" count="24">
          1 1 -1 1 -1 -1 -1 -1 -1 -1 1 -1 1 1 1 1 -1 1 -1 -1 1 -1 1 1
        </float_array>
      </source>
      <vertices id="Cube-mesh-vertices">
        <input semantic="POSITION" source="#Cube-mesh-positions" />
      </vertices>
      <polylist material="Material1" count="6">
        <input semantic="VERTEX" source="#Cube-mesh-vertices" offset="0" />
        <vcount>4 4 4 4 4 4 </vcount>
        <p>0 1 2 3 4 7 6 5 0 4 5 1 1 5 6 2 2 6 7 3 4 0 3 7</p>
      </polylist>
    </mesh>
  </geometry>
```

3D Studio: este un format binar, ce se citeste pe chunkuri, ca structura e arborescent.
Este similar cu celelalte insa are cateva limitari: toate mesele sunt facute din triunghiuri, numarul de noduri este limitat la 65536.

```
0x4D4D // Main Chunk
├── 0x0002 // M3D Version
├── 0x3D3D // 3D Editor Chunk
│   ├── 0x4000 // Object Block
│   │   ├── 0x4100 // Triangular Mesh
│   │   │   ├── 0x4110 // Vertices List
│   │   │   ├── 0x4120 // Faces Description
│   │   │   │   ├── 0x4130 // Faces Material
│   │   │   │   └── 0x4150 // Smoothing Group List
│   │   │   ├── 0x4140 // Mapping Coordinates List
│   │   │   └── 0x4160 // Local Coordinates System
│   │   ├── 0x4600 // Light
│   │   │   └── 0x4610 // Spotlight
│   │   └── 0x4700 // Camera
│   └── 0xAFFF // Material Block
```

FBX: Este un format proprietar Autodesk, este tot intr-o structura arborescenta, insa aceasta permite definirea animatiilor asupra obiectelor.

```
FBXHeaderExtension:  { <---- beginning of node
	FBXHeaderVersion: 1003 <---- node property
	FBXVersion: 6100
	CreationTimeStamp:  { <---- beginning of subnode (1)
		Version: 1000  <---- subnode property
		Year: 2014
		Month: 03
		Day: 20
		Hour: 17
		Minute: 38
		Second: 29
		Millisecond: 0
	} <---- end of subnode (1)
	Creator: "FBX SDK/FBX Plugins build 20070228"
	OtherFlags:  { <---- beginning of subnode (2)
		FlagPLE: 0
	} <---- end of subnode (2)
}
```

Wavefront: permite definirea nodurilor si a texturilor respective, dar pe langa astea mai poate contine curbe, forme libere.

```
# List of geometric vertices, with (x,y,z[,w]) coordinates, w is optional and defaults to 1.0.
  v 0.123 0.234 0.345 1.0
  v ...
  ...
  # List of texture coordinates, in (u, v [,w]) coordinates, these will vary between 0 and 1,
w is optional and defaults to 0.
  vt 0.500 1 [0]
  vt ...
  ...
  # List of vertex normals in (x,y,z) form; normals might not be unit vectors.
  vn 0.707 0.000 0.707
```

Surse:

http://www.real3dtutorials.com/tut00021.php
https://en.wikipedia.org/wiki/.3ds
https://banexdevblog.wordpress.com/2014/06/23/a-quick-tutorial-about-the-fbx-ascii-format/
https://en.wikipedia.org/wiki/Wavefront_.obj_file

4. Identificati (dezvoltati) coduri sursa / articole care sa implementeze / detalieze temele discutate la curs. Intocmiti o scurta documentatie.

https://www.youtube.com/watch?v=9wuuLbwr0vQ -- tutorial video de bleder pentru a realiza arhitectura de baza a unei case
https://www.youtube.com/watch?v=Ay23U9t4ja8 -- tutorial video de blender pentru a realiza usi cu detalii
http://people.sc.fsu.edu/~jburkardt/data/ply/ply.html -- formatul PLY explicat
http://paulbourke.net/dataformats/ply/ -- formatul PLY explicat
https://github.com/jimwise/ply -- implementare in Ruby pentru a citi fisiere PLY
https://github.com/dranjan/python-plyfile -- implementare in Python de citire / exportare PLY folosind la baza vectori din numpy. Exemplu de parsare:

```
>>> plydata.elements[0].name
'vertex'
>>> plydata.elements[0].data[0]
(0.0, 0.0, 0.0)
>>> plydata.elements[0].data['x']
array([ 0.,  0.,  1.,  1.], dtype=float32)
>>> plydata['face'].data['vertex_indices'][0]
array([0, 1, 2], dtype=int32
```

http://web.cse.ohio-state.edu/~shen.94/5542/Site/Ply.html -- implementare in C de citire a unui fisier PLY