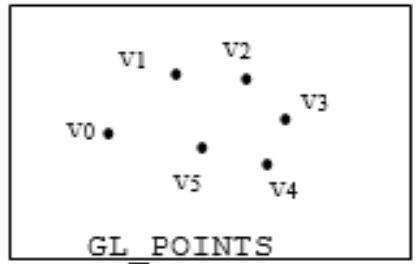
Curs
Grafica

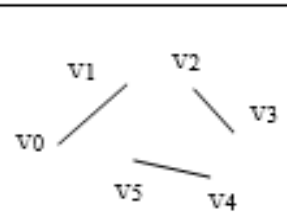
OpenGL

Tipurile de primitive geometrice

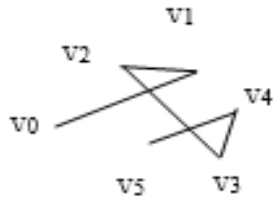
Proprietate	Primitivă
GL_POINTS	Desenează n puncte
GL_LINES	Desenează segmentele de dreaptă izolate (v_0,v_1), (v_2,v_3),… ş.a.m.d. Dacă n este impar ultimul vârf este ignorat
GL_LINE_STRIP	Desenează linia poligonală formată din segmentele (v_0,v_1), (v_1,v_2), .. . …(v_{n-2}, v_{n-1})
GL_LINE_LOOP	La fel ca primitiva GL_LINE_STRIP, dar se mai desenează segmentul (v_n, v_0) care închide o buclă.
GL_TRIANGLES	Desenează o serie de triunghiuri folosind vârfurile (v_0,v_1,v_2), (v_3,v_4,v_5), ş.a.m.d. Dacă n nu este multiplu de 3, atunci ultimele 1 sau 2 vârfuri sunt ignorate.
GL_TRIANGLE_STRIP	Desenează o serie de triunghiuri folosind vârfurile (v_0,v_1,v_2), (v_2,v_1,v_3),..ş.a.m.d. Ordinea este aleasă astfel ca triunghiurile să aibă aceeaşi orientare, deci să poată forma o suprafaţă închisă.
GL_TRIANGLE_FAN	Desenează triunghiurile (v_0,v_1,v_2), (v_0,v_2,v_3), ş.a.m.d.
GL_QUADS	Desenează o serie de patrulatere (v_0,v_1,v_2,v_3), (v_4,v_5,v_6,v_7), ş.a.m.d. Dacă n nu este multiplu de 4, ultimele 1, 2 sau 3 vârfuri sunt ignorate.
GL_QUADS_STRIP	Desenează o serie de patrulatere (v_0,v_1,v_3,v_2), (v_3,v_2,v_5,v_4), ş.a.m.d Dacă n < 4, nu se desenază nimic. Dacă n este impar, ultimul vârf este ignorat.
GL_POLYGON	Desenează un poligon cu n vârfuri, (v_0,v_1,…v_{n-1}). Dacă poligonul nu este convex, rezultatul este impredictibil.



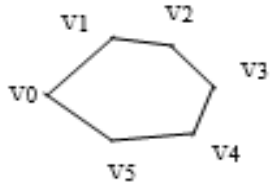
GL_POINTS



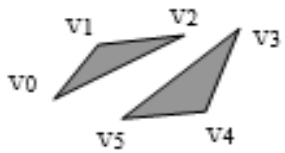
GL_LINES



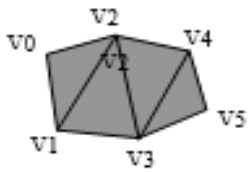
GL_LINE_STRIP



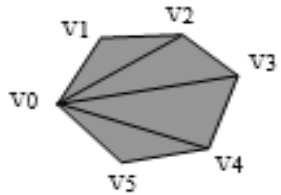
GL_LINE_LOOP



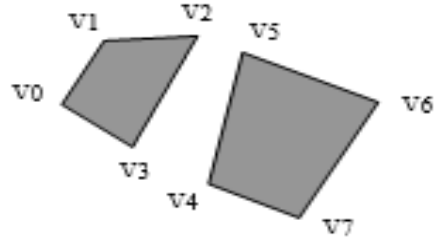
GL_TRIANGLES



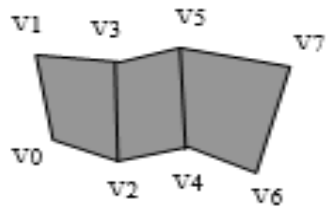
GL_TRIANGLE_STRIP



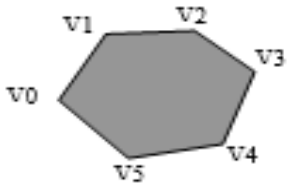
GL_TRIANGLE_FAN



GL_QUADS



GL_QUAD_STRIP



GL_POLYGON

Primitivele de tip suprafaţă (triunghiuri, patrulatere, poligoane) pot fi desenate în modul “cadru de sârmă” (*wireframe*), sau în modul plin (*fill*), prin setarea variabilei de stare GL_POLYGON_MODE folosind funcţia:

- void glPolygonMode(GLenum face, GLenum mode);
- unde argumentul mode poate lua una din valorile:
- GL_POINT : se desenează numai vârfurile primitivei, ca puncte în spaţiu, indiferent de tipul acesteia.
- GL_LINE: muchiile poligoanelor se desenează ca segmente de dreaptă.
- GL_FILL: se desenează poligonul plin.

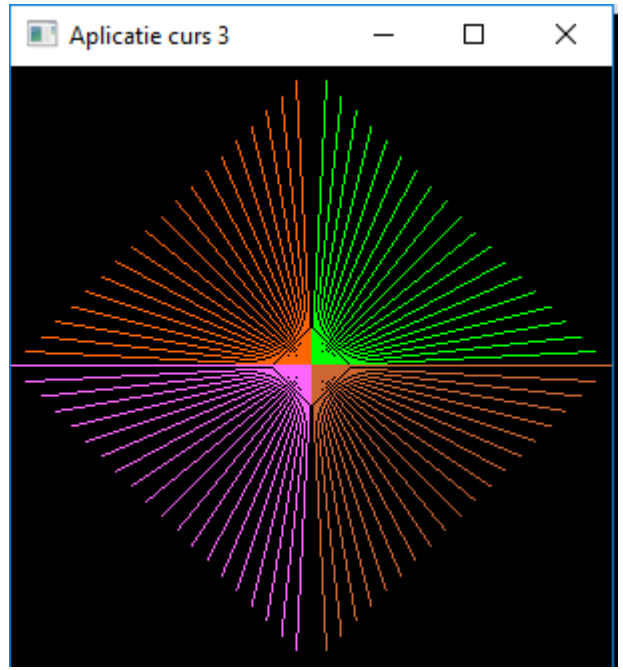
Argumentul face se referă la tipul primitivei geometrice din punct de vedere al orientării căreia i se aplică modul de redare mode.

OpenGL admite primitive orientate direct (frontface) şi primitive orientate invers (backface). Argumentul face poate lua una din valorile: GL_FRONT, GL_BACK sau GL_FRONT_AND_BACK, pentru a se specifica primitive orientate direct, primitive orientate invers şi, respectiv ambele tipuri de primitive.

Tipuri de date OpenGL

Sufix	Tipul de date	Corespondentul în C	Tipul definit în OpenGL
b	8-bit integer	signed char	GLbyte
s	16-bit integer	Short	GLshort
i	32-bit integer	int sau long	GLint, GLsizei
f	32-bit floating-point	Float	GLfloat, GLclampf
d	64-bit floating-point	Double	GLdouble, GLclampd
ub	8-bitunsigned integer	unsigned char	GLubyte, GLboolean
us	16-bit unsigned integer	unsigned short	GLushort
ui	32-bit unsigned integer	unsigned int sau unsigned long	GLuint,GLenum, GLbitfield

```
#include "stdafx.h"
#include <gl/freeglut.h>
void Display(void)
{
glClear(GL_COLOR_BUFFER_BIT);
glColor3f(0, 1, 0);
glBegin(GL_LINES);
// Cadran 1
for (int i = 0; i<20; i++)
{
glVertex3f(0, 0, 0);
glVertex3f(1 - i / 20.0, i / 20.0, 0);
} // Cadran 2
glColor3f(1, 0.4, 0);
for (int i = 0; i<20; i++)
{
glVertex3f(0, 0, 0);
glVertex3f(-1 + i / 20.0, i / 20.0, 0);
} // Cadran 3
glColor3f(1, 0.4, 1);
for (int i = 0; i<20; i++)
{
glVertex3f(0, 0, 0);
glVertex3f(-1 + i / 20.0, -i / 20.0, 0);
}
// Cadran 4
glColor3f(0.8, 0.4, 0.2);
for (int i = 0; i<20; i++)
{
glVertex3f(0, 0, 0);
glVertex3f(1 - i / 20.0, -i / 20.0, 0);
}
glEnd();   glFlush();  glColor3f(0, 0, 1);
}
int main(int argc, char** argv) {
glutInit(&argc, argv);
glutInitDisplayMode(GLUT_SINGLE |
GLUT_RGB);//se specifica modelul de
culoare al ferestrei: un singur buffer si
culoare RGB
glutCreateWindow("Problema 4 varianta
cu cadrame");
glutDisplayFunc(Display);
glutMainLoop();
return 0;
}
```



glutPostRedisplay

glutPostRedisplay marks the current window as needing to be redisplayed.

Usage

void glutPostRedisplay(void);

Description

Mark the normal plane of current window as needing to be redisplayed. The next iteration through glutMainLoop, the window's display callback will be called to redisplay the window's normal plane. Multiple calls to glutPostRedisplay before the next display callback opportunity generates only a single redisplay callback. glutPostRedisplay may be called within a window's display or overlay display callback to re-mark that window for redisplay.

Logically, normal plane damage notification for a window is treated as a glutPostRedisplay on the damaged window. Unlike damage reported by the window system, glutPostRedisplay will not set to true the normal plane's damaged status (returned by glutLayerGet(GLUT_NORMAL_DAMAGED).

https://www.opengl.org/resources/libraries/glut/spec3/node20.html

glRotatef

- The first argument specifies how many degree you want to rotate around an axis. The following three argument specifiy the x, y, z axis to rotate, respectively. In that statement, you're rotating the y and z axis 0 degrees. Thats why you don't see any change. Also, the last three parameters are clamped to the range of [0, 1].

Utilizarea ferestrelor in OpenGl/ Creare fereastră

```
int glutCreateWindow(char *string) ;
```

Funcţia **glutCreateWindow** creează o fereastra cu un context OpenGL. Ea intoarce un identificator unic pentru fereastra nou creată. Fereastra nu va fi afişată înainte de apelarea funcţiei **glutMainLoop**. Valoarea întoarsă de funcţie reprezintă identificatorul ferestrei, care este unic.

Creare ferestra copil

```
int glutCreateSubWindow(int win, int x, int y,
int width, int height);
```

Funcţia creează o fereastră având ca părinte fereastra identificată de **win**, unde :

- **win** - reprezintă identificatorul ferestrei părinte;
- (**x**, **y**) - reprezintă colţul stânga sus al ferestrei (x şi y sunt exprimate în pixeli şi sunt relative la originea ferestrei părinte);
- **width** - reprezintă lăţimea ferestrei (exprimată în pixeli);
- **height** - reprezintă înălţimea ferestrei (exprimată în pixeli);

Fereastra nou creată devine fereastra curentă. Funcţia întoarce identificatorul ferestrei create. Exemplu :

```
int winMain, winSub ;
winMain= glutCreateWindow("My Main window");
winSub = glutCreateSubWindow(winMain, xtop, ytop, width, height);
```

The first parameter of glutCreatesubWindow is just the index of the parent window. As you know glutCreateWindow returns an integer. You must use that integer in glutCreatesubWindow to tell who is the parent window.

http://www.opengl.org/resources/libraries/glut/spec3/node17.html

Distrugere fereastră

```
void glutDestroyWindow(int win) ;
```

Funcţia distruge fereastra specificată de **win**. De asemenea, este distrus şi contextul OpenGL asociat ferestrei. Orice subfereastră a ferestrei distruse va fi de asemenea distrusă. Dacă **win** identifică fereastra curentă, atunci ea va deveni invalidă.

Selectarea ferestrei curente

```
void glutSetWindow(int win) ;
```

Funcţia selectează fereastra curentă ca fiind cea identificată de parametrul **win**.

Aflarea ferestrei curente

```
int glutGetWindow(void) ;
```

Funcţia întoarce identificatorul ferestrei curente. Funcţia întoarce 0 dacă nu există nici o fereastră curentă sau fereastra curentă a fost distrusă.

Selectarea cursorului asociat ferestrei curente

```
void glutSetCursor(int cursor) ;
```

parametrul **cursor,** care poate avea una din următoarele valori:

void glutSetCursor(int cursor); cursor
Name of cursor image to change to.

GLUT_CURSOR_RIGHT_ARROW
Arrow pointing up and to the right.
GLUT_CURSOR_LEFT_ARROW
Arrow pointing up and to the left.
GLUT_CURSOR_INFO
Pointing hand.
GLUT_CURSOR_DESTROY
Skull & cross bones.
GLUT_CURSOR_HELP
Question mark.
GLUT_CURSOR_CYCLE
Arrows rotating in a circle.
GLUT_CURSOR_SPRAY
Spray can.
GLUT_CURSOR_WAIT
Wrist watch.
GLUT_CURSOR_TEXT
Insertion point cursor for text.
GLUT_CURSOR_CROSSHAIR
Simple cross-hair.
GLUT_CURSOR_UP_DOWN
Bi-directional pointing up & down.
GLUT_CURSOR_LEFT_RIGHT
Bi-directional pointing left & right.

GLUT_CURSOR_TOP_SIDE
Arrow pointing to top side.
GLUT_CURSOR_BOTTOM_SIDE
Arrow pointing to bottom side.
GLUT_CURSOR_LEFT_SIDE
Arrow pointing to left side.
GLUT_CURSOR_RIGHT_SIDE
Arrow pointing to right side.
GLUT_CURSOR_TOP_LEFT_CORNER
Arrow pointing to top-left corner.
GLUT_CURSOR_TOP_RIGHT_CORNER
Arrow pointing to top-right corner.
GLUT_CURSOR_BOTTOM_RIGHT_CORNER
Arrow pointing to bottom-left corner.
GLUT_CURSOR_BOTTOM_LEFT_CORNER
Arrow pointing to bottom-right corner.
GLUT_CURSOR_FULL_CROSSHAIR
Full-screen cross-hair cursor (if possible, otherwise GLUT_CURSOR_CROSSHAIR).
GLUT_CURSOR_NONE
Invisible cursor.
GLUT_CURSOR_INHERIT
Use parent's cursor.

Gestiunea meniurilor

Menu Management

http://www.opengl.org/resources/libraries/glut/spec3/node35.html

- GLUT supports simple cascading pop-up menus. They are designed to let a user select various modes within a program. The functionality is simple and minimalistic and is meant to be that way. Do not mistake GLUT's pop-up menu facility with an attempt to create a full-featured user interface.
- It is illegal to create or destroy menus, or change, add, or remove menu items while a menu (and any cascaded sub-menus) are in use (that is, popped up).

1 glutCreateMenu

2 glutSetMenu, glutGetMenu

3 glutDestroyMenu

4 glutAddMenuEntry

5 glutAddSubMenu

6 glutChangeToMenuEntry

7 glutChangeToSubMenu

8 glutRemoveMenuItem

9 glutAttachMenu, glutDetachMenu

Crearea meniurilor

Meniurile create cu ajutorul GLUT-ului sunt meniuri simple, de tip pop-up în cascadă.

int **glutCreateMenu**(void (*func)(int value)) ;

Funcţia creează un nou meniu pop-up şi întoarce identificatorul corespunzator, de tip întreg. Identificatorii meniurilor încep de la valoarea 1. Valorile lor sunt separate de identificatorii ferestrelor.

- parametrul *func* specifică o **funcţie callback** a aplicaţiei, care va fi apelată de biblioteca GLUT la selectarea unei opţiuni din meniu.

Parametrul transmis funcţiei callback (*value*) specifică opţiunea de meniu selectată.

Adăugarea unei opţiuni într-un meniu

void **glutAddMenuEntry**(char* name, int value) ;

Funcţia adaugă o nouă opţiune meniului curent, la sfârşitul listei de articole a meniului.

•*name* - specifică textul prin care va fi reprezentată opţiunea introdusă

•*value* - reprezintă valoarea transmisă funcţiei callback asociată meniului la selectarea opţiunii respective

glutAddMenuEntry adds a menu entry to the bottom of the *current menu*.

The string name will be displayed for the newly added menu entry.

If the menu entry is selected by the user, the menu's callback will be called passing value as the callback's parameter.

Adăugarea unui submeniu într-un meniu

void **glutAddSubMenu**(char* name, int menu) ;

Funcţia adaugă un submeniu la sfârşitul meniului curent.

name - reprezintă numele submeniului introdus (textul prin care va fi afişat în meniu)

menu - reprezintă identificatorul submeniului.

Ştergerea unei opţiuni sau a unui submeniu

void **glutRemoveMenuItem**(int entry) ;

Funcţia şterge opţiunea sau submeniul identificat de parametrul entry. Opţiunile din meniu de sub opţiunea ştearsă sunt renumerotate.

Distrugerea unui meniu

void **glutDestroyMenu**(int menu) ;

Funcţia distruge meniul specificat prin parametru.

Observatie: dacă meniul distrus este cel curent, atunci meniul curent fi deveni invalid.

Setarea meniului curent

void **glutSetMenu**(int menu) ;

- Funcţia setează meniul curent ca fiind cel identificat de parametrul ***menu***.

Aflarea meniului curent

int **glutGetMenu**(void) ;

- Funcţia întoarce identificatorul meniului curent. Funcţia întoarce 0 dacă nu există meniu curent sau meniul curent a fost distrus.

Ataşare / detaşare meniu unui buton al mouse-ului

void **glutAttachMenu**(int button) ;

void **glutDetachMenu**(int button) ;

Funcţia **glutAttachMenu** ataşează meniul curent butonului mouse-ului specificat de parametrul *button*. Funcţia **glutDetachMenu** detaşează butonul asociat meniului curent. Prin ataşarea unui buton al mouse-ului meniului curent, meniul va fi derulat la apăsarea butonului respectiv în poziţia curentă a cursorului.