



Universidad Nacional  
Autónoma de México



Practica No. 2

Alumno: Arroyo Llanes Miguel Alejandro.

No. Cuenta: 319218460

Materia: Lab. Computación Gráfica e Interacción Humano-Computadora

Grupo Laboratorio: 3

Grupo teoría: 6

Fecha de Entrega: 22/02/2025

Nombre profesor: Ing. Jose Roque Roman Guadarrama

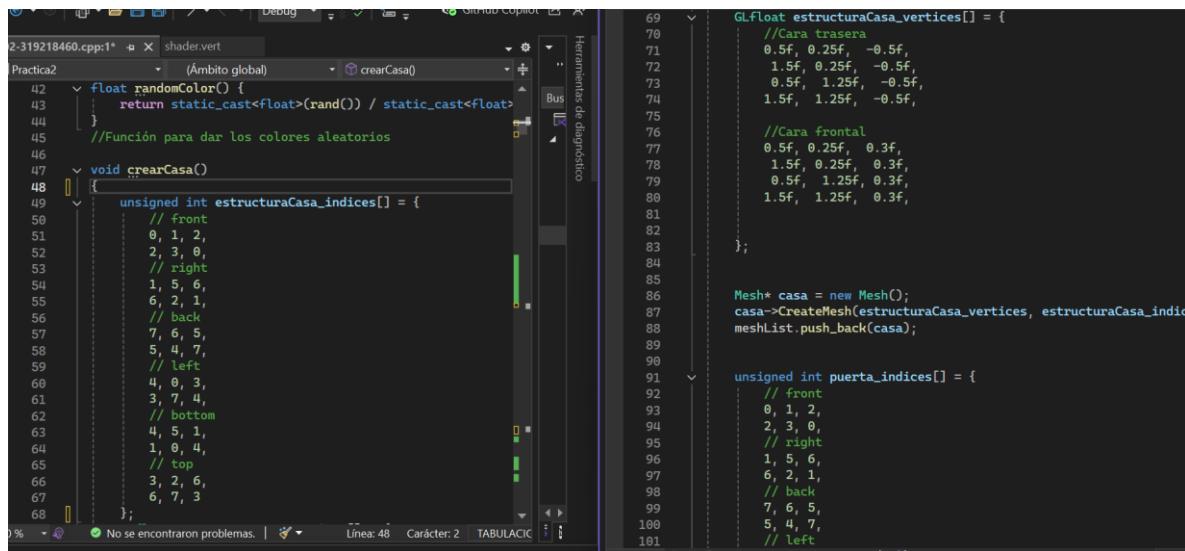
## Comentario:

Primeramente, se retomó el dibujado a mano a través de una Tablet, sin embargo, esta vez aquello solo sirvió para las letras, para la casa se tuvo que realizar una investigación para entender los modelos 3d de una forma más correcta a lo que se buscaba realizar en esta práctica, una vez entendido se tuvo que modificar todos los archivos de shader para la asignación de colores de una forma controlada dejando comentarios para poder ubicar bien cada bloque de código.

Además, para no tener que editar más allá de los colores a las letras se utilizó la función de traslación aprendida en el ejercicio anterior para así solo mover un poco abajo a la izquierda y que no se encimara con la casa.

## Bloque de código:

Líneas de código que muestran la función random para las letras y el dibujado de la casa.



The screenshot shows a code editor with two tabs: 'shader.vert' and '2-319218460.cpp'. The '2-319218460.cpp' tab contains C++ code for generating a house mesh. It includes a function 'randomColor()' and a 'crearCasa()' function. The 'crearCasa()' function defines vertex and index arrays for the house's faces. The GLSL code in 'shader.vert' defines vertex and fragment shaders, including color assignments and texture sampling logic.

```
2-319218460.cpp:1* shader.vert
Practica2
42     float randomColor() {
43         return static_cast<float>(rand()) / static_cast<float>(RAND_MAX);
44     }
45     //Función para dar los colores aleatorios
46
47     void crearCasa()
48     {
49         unsigned int estructuraCasa_indices[] = {
50             // front
51             0, 1, 2,
52             2, 3, 0,
53             // right
54             1, 5, 6,
55             6, 2, 1,
56             // back
57             7, 6, 5,
58             5, 4, 7,
59             // left
60             4, 0, 3,
61             3, 7, 4,
62             // bottom
63             4, 5, 1,
64             1, 0, 4,
65             // top
66             3, 2, 6,
67             6, 7, 3
68         };
69
70         GLfloat estructuraCasa_vertices[] = {
71             //Cara trasera
72             0.5f, 0.25f, -0.5f,
73             1.5f, 0.25f, -0.5f,
74             0.5f, 1.25f, -0.5f,
75             1.5f, 1.25f, -0.5f,
76             //Cara frontal
77             0.5f, 0.25f, 0.3f,
78             1.5f, 0.25f, 0.3f,
79             0.5f, 1.25f, 0.3f,
80             1.5f, 1.25f, 0.3f,
81
82         };
83
84
85         Mesh* casa = new Mesh();
86         casa->CreateMesh(estructuraCasa_vertices, estructuraCasa_indices);
87         meshList.push_back(casa);
88
89
90         unsigned int puerta_indices[] = {
91             // front
92             0, 1, 2,
93             2, 3, 0,
94             // right
95             1, 5, 6,
96             6, 2, 1,
97             // back
98             7, 6, 5,
99             5, 4, 7,
100            // left
101
102     };
103
104     //Cara frontal
105     0.5f, 0.25f, 0.3f,
106     1.5f, 0.25f, 0.3f,
107     0.5f, 1.25f, 0.3f,
108     1.5f, 1.25f, 0.3f,
109
110     //Cara trasera
111     0.5f, 0.25f, -0.5f,
112     1.5f, 0.25f, -0.5f,
113     0.5f, 1.25f, -0.5f,
114     1.5f, 1.25f, -0.5f,
115
116     //puerta
117     0.5f, 0.25f, 0.3f,
118     1.5f, 0.25f, 0.3f,
119     0.5f, 1.25f, 0.3f,
120     1.5f, 1.25f, 0.3f,
121
122     //Cara frontal
123     0.5f, 0.25f, 0.3f,
124     1.5f, 0.25f, 0.3f,
125     0.5f, 1.25f, 0.3f,
126     1.5f, 1.25f, 0.3f,
127
128     //Cara trasera
129     0.5f, 0.25f, -0.5f,
130     1.5f, 0.25f, -0.5f,
131     0.5f, 1.25f, -0.5f,
132     1.5f, 1.25f, -0.5f,
133
134     //puerta
135     0.5f, 0.25f, 0.3f,
136     1.5f, 0.25f, 0.3f,
137     0.5f, 1.25f, 0.3f,
138     1.5f, 1.25f, 0.3f,
139
140     //Cara frontal
141     0.5f, 0.25f, 0.3f,
142     1.5f, 0.25f, 0.3f,
143     0.5f, 1.25f, 0.3f,
144     1.5f, 1.25f, 0.3f,
145
146     //Cara trasera
147     0.5f, 0.25f, -0.5f,
148     1.5f, 0.25f, -0.5f,
149     0.5f, 1.25f, -0.5f,
150     1.5f, 1.25f, -0.5f,
151
152     //puerta
153     0.5f, 0.25f, 0.3f,
154     1.5f, 0.25f, 0.3f,
155     0.5f, 1.25f, 0.3f,
156     1.5f, 1.25f, 0.3f,
157
158     //Cara frontal
159     0.5f, 0.25f, 0.3f,
160     1.5f, 0.25f, 0.3f,
161     0.5f, 1.25f, 0.3f,
162     1.5f, 1.25f, 0.3f,
163
164     //Cara trasera
165     0.5f, 0.25f, -0.5f,
166     1.5f, 0.25f, -0.5f,
167     0.5f, 1.25f, -0.5f,
168     1.5f, 1.25f, -0.5f,
169
170     //puerta
171     0.5f, 0.25f, 0.3f,
172     1.5f, 0.25f, 0.3f,
173     0.5f, 1.25f, 0.3f,
174     1.5f, 1.25f, 0.3f,
175
176     //Cara frontal
177     0.5f, 0.25f, 0.3f,
178     1.5f, 0.25f, 0.3f,
179     0.5f, 1.25f, 0.3f,
180     1.5f, 1.25f, 0.3f,
181
182     //Cara trasera
183     0.5f, 0.25f, -0.5f,
184     1.5f, 0.25f, -0.5f,
185     0.5f, 1.25f, -0.5f,
186     1.5f, 1.25f, -0.5f,
187
188     //puerta
189     0.5f, 0.25f, 0.3f,
190     1.5f, 0.25f, 0.3f,
191     0.5f, 1.25f, 0.3f,
192     1.5f, 1.25f, 0.3f,
193
194     //Cara frontal
195     0.5f, 0.25f, 0.3f,
196     1.5f, 0.25f, 0.3f,
197     0.5f, 1.25f, 0.3f,
198     1.5f, 1.25f, 0.3f,
199
200     //Cara trasera
201     0.5f, 0.25f, -0.5f,
202     1.5f, 0.25f, -0.5f,
203     0.5f, 1.25f, -0.5f,
204     1.5f, 1.25f, -0.5f,
205
206     //puerta
207     0.5f, 0.25f, 0.3f,
208     1.5f, 0.25f, 0.3f,
209     0.5f, 1.25f, 0.3f,
210     1.5f, 1.25f, 0.3f,
211
212     //Cara frontal
213     0.5f, 0.25f, 0.3f,
214     1.5f, 0.25f, 0.3f,
215     0.5f, 1.25f, 0.3f,
216     1.5f, 1.25f, 0.3f,
217
218     //Cara trasera
219     0.5f, 0.25f, -0.5f,
220     1.5f, 0.25f, -0.5f,
221     0.5f, 1.25f, -0.5f,
222     1.5f, 1.25f, -0.5f,
223
224     //puerta
225     0.5f, 0.25f, 0.3f,
226     1.5f, 0.25f, 0.3f,
227     0.5f, 1.25f, 0.3f,
228     1.5f, 1.25f, 0.3f,
229
230     //Cara frontal
231     0.5f, 0.25f, 0.3f,
232     1.5f, 0.25f, 0.3f,
233     0.5f, 1.25f, 0.3f,
234     1.5f, 1.25f, 0.3f,
235
236     //Cara trasera
237     0.5f, 0.25f, -0.5f,
238     1.5f, 0.25f, -0.5f,
239     0.5f, 1.25f, -0.5f,
240     1.5f, 1.25f, -0.5f,
241
242     //puerta
243     0.5f, 0.25f, 0.3f,
244     1.5f, 0.25f, 0.3f,
245     0.5f, 1.25f, 0.3f,
246     1.5f, 1.25f, 0.3f,
247
248     //Cara frontal
249     0.5f, 0.25f, 0.3f,
250     1.5f, 0.25f, 0.3f,
251     0.5f, 1.25f, 0.3f,
252     1.5f, 1.25f, 0.3f,
253
254     //Cara trasera
255     0.5f, 0.25f, -0.5f,
256     1.5f, 0.25f, -0.5f,
257     0.5f, 1.25f, -0.5f,
258     1.5f, 1.25f, -0.5f,
259
260     //puerta
261     0.5f, 0.25f, 0.3f,
262     1.5f, 0.25f, 0.3f,
263     0.5f, 1.25f, 0.3f,
264     1.5f, 1.25f, 0.3f,
265
266     //Cara frontal
267     0.5f, 0.25f, 0.3f,
268     1.5f, 0.25f, 0.3f,
269     0.5f, 1.25f, 0.3f,
270     1.5f, 1.25f, 0.3f,
271
272     //Cara trasera
273     0.5f, 0.25f, -0.5f,
274     1.5f, 0.25f, -0.5f,
275     0.5f, 1.25f, -0.5f,
276     1.5f, 1.25f, -0.5f,
277
278     //puerta
279     0.5f, 0.25f, 0.3f,
280     1.5f, 0.25f, 0.3f,
281     0.5f, 1.25f, 0.3f,
282     1.5f, 1.25f, 0.3f,
283
284     //Cara frontal
285     0.5f, 0.25f, 0.3f,
286     1.5f, 0.25f, 0.3f,
287     0.5f, 1.25f, 0.3f,
288     1.5f, 1.25f, 0.3f,
289
290     //Cara trasera
291     0.5f, 0.25f, -0.5f,
292     1.5f, 0.25f, -0.5f,
293     0.5f, 1.25f, -0.5f,
294     1.5f, 1.25f, -0.5f,
295
296     //puerta
297     0.5f, 0.25f, 0.3f,
298     1.5f, 0.25f, 0.3f,
299     0.5f, 1.25f, 0.3f,
300     1.5f, 1.25f, 0.3f,
301
302     //Cara frontal
303     0.5f, 0.25f, 0.3f,
304     1.5f, 0.25f, 0.3f,
305     0.5f, 1.25f, 0.3f,
306     1.5f, 1.25f, 0.3f,
307
308     //Cara trasera
309     0.5f, 0.25f, -0.5f,
310     1.5f, 0.25f, -0.5f,
311     0.5f, 1.25f, -0.5f,
312     1.5f, 1.25f, -0.5f,
313
314     //puerta
315     0.5f, 0.25f, 0.3f,
316     1.5f, 0.25f, 0.3f,
317     0.5f, 1.25f, 0.3f,
318     1.5f, 1.25f, 0.3f,
319
320     //Cara frontal
321     0.5f, 0.25f, 0.3f,
322     1.5f, 0.25f, 0.3f,
323     0.5f, 1.25f, 0.3f,
324     1.5f, 1.25f, 0.3f,
325
326     //Cara trasera
327     0.5f, 0.25f, -0.5f,
328     1.5f, 0.25f, -0.5f,
329     0.5f, 1.25f, -0.5f,
330     1.5f, 1.25f, -0.5f,
331
332     //puerta
333     0.5f, 0.25f, 0.3f,
334     1.5f, 0.25f, 0.3f,
335     0.5f, 1.25f, 0.3f,
336     1.5f, 1.25f, 0.3f,
337
338     //Cara frontal
339     0.5f, 0.25f, 0.3f,
340     1.5f, 0.25f, 0.3f,
341     0.5f, 1.25f, 0.3f,
342     1.5f, 1.25f, 0.3f,
343
344     //Cara trasera
345     0.5f, 0.25f, -0.5f,
346     1.5f, 0.25f, -0.5f,
347     0.5f, 1.25f, -0.5f,
348     1.5f, 1.25f, -0.5f,
349
350     //puerta
351     0.5f, 0.25f, 0.3f,
352     1.5f, 0.25f, 0.3f,
353     0.5f, 1.25f, 0.3f,
354     1.5f, 1.25f, 0.3f,
355
356     //Cara frontal
357     0.5f, 0.25f, 0.3f,
358     1.5f, 0.25f, 0.3f,
359     0.5f, 1.25f, 0.3f,
360     1.5f, 1.25f, 0.3f,
361
362     //Cara trasera
363     0.5f, 0.25f, -0.5f,
364     1.5f, 0.25f, -0.5f,
365     0.5f, 1.25f, -0.5f,
366     1.5f, 1.25f, -0.5f,
367
368     //puerta
369     0.5f, 0.25f, 0.3f,
370     1.5f, 0.25f, 0.3f,
371     0.5f, 1.25f, 0.3f,
372     1.5f, 1.25f, 0.3f,
373
374     //Cara frontal
375     0.5f, 0.25f, 0.3f,
376     1.5f, 0.25f, 0.3f,
377     0.5f, 1.25f, 0.3f,
378     1.5f, 1.25f, 0.3f,
379
380     //Cara trasera
381     0.5f, 0.25f, -0.5f,
382     1.5f, 0.25f, -0.5f,
383     0.5f, 1.25f, -0.5f,
384     1.5f, 1.25f, -0.5f,
385
386     //puerta
387     0.5f, 0.25f, 0.3f,
388     1.5f, 0.25f, 0.3f,
389     0.5f, 1.25f, 0.3f,
390     1.5f, 1.25f, 0.3f,
391
392     //Cara frontal
393     0.5f, 0.25f, 0.3f,
394     1.5f, 0.25f, 0.3f,
395     0.5f, 1.25f, 0.3f,
396     1.5f, 1.25f, 0.3f,
397
398     //Cara trasera
399     0.5f, 0.25f, -0.5f,
400     1.5f, 0.25f, -0.5f,
401     0.5f, 1.25f, -0.5f,
402     1.5f, 1.25f, -0.5f,
403
404     //puerta
405     0.5f, 0.25f, 0.3f,
406     1.5f, 0.25f, 0.3f,
407     0.5f, 1.25f, 0.3f,
408     1.5f, 1.25f, 0.3f,
409
410     //Cara frontal
411     0.5f, 0.25f, 0.3f,
412     1.5f, 0.25f, 0.3f,
413     0.5f, 1.25f, 0.3f,
414     1.5f, 1.25f, 0.3f,
415
416     //Cara trasera
417     0.5f, 0.25f, -0.5f,
418     1.5f, 0.25f, -0.5f,
419     0.5f, 1.25f, -0.5f,
420     1.5f, 1.25f, -0.5f,
421
422     //puerta
423     0.5f, 0.25f, 0.3f,
424     1.5f, 0.25f, 0.3f,
425     0.5f, 1.25f, 0.3f,
426     1.5f, 1.25f, 0.3f,
427
428     //Cara frontal
429     0.5f, 0.25f, 0.3f,
430     1.5f, 0.25f, 0.3f,
431     0.5f, 1.25f, 0.3f,
432     1.5f, 1.25f, 0.3f,
433
434     //Cara trasera
435     0.5f, 0.25f, -0.5f,
436     1.5f, 0.25f, -0.5f,
437     0.5f, 1.25f, -0.5f,
438     1.5f, 1.25f, -0.5f,
439
440     //puerta
441     0.5f, 0.25f, 0.3f,
442     1.5f, 0.25f, 0.3f,
443     0.5f, 1.25f, 0.3f,
444     1.5f, 1.25f, 0.3f,
445
446     //Cara frontal
447     0.5f, 0.25f, 0.3f,
448     1.5f, 0.25f, 0.3f,
449     0.5f, 1.25f, 0.3f,
450     1.5f, 1.25f, 0.3f,
451
452     //Cara trasera
453     0.5f, 0.25f, -0.5f,
454     1.5f, 0.25f, -0.5f,
455     0.5f, 1.25f, -0.5f,
456     1.5f, 1.25f, -0.5f,
457
458     //puerta
459     0.5f, 0.25f, 0.3f,
460     1.5f, 0.25f, 0.3f,
461     0.5f, 1.25f, 0.3f,
462     1.5f, 1.25f, 0.3f,
463
464     //Cara frontal
465     0.5f, 0.25f, 0.3f,
466     1.5f, 0.25f, 0.3f,
467     0.5f, 1.25f, 0.3f,
468     1.5f, 1.25f, 0.3f,
469
470     //Cara trasera
471     0.5f, 0.25f, -0.5f,
472     1.5f, 0.25f, -0.5f,
473     0.5f, 1.25f, -0.5f,
474     1.5f, 1.25f, -0.5f,
475
476     //puerta
477     0.5f, 0.25f, 0.3f,
478     1.5f, 0.25f, 0.3f,
479     0.5f, 1.25f, 0.3f,
480     1.5f, 1.25f, 0.3f,
481
482     //Cara frontal
483     0.5f, 0.25f, 0.3f,
484     1.5f, 0.25f, 0.3f,
485     0.5f, 1.25f, 0.3f,
486     1.5f, 1.25f, 0.3f,
487
488     //Cara trasera
489     0.5f, 0.25f, -0.5f,
490     1.5f, 0.25f, -0.5f,
491     0.5f, 1.25f, -0.5f,
492     1.5f, 1.25f, -0.5f,
493
494     //puerta
495     0.5f, 0.25f, 0.3f,
496     1.5f, 0.25f, 0.3f,
497     0.5f, 1.25f, 0.3f,
498     1.5f, 1.25f, 0.3f,
499
500     //Cara frontal
501     0.5f, 0.25f, 0.3f,
502     1.5f, 0.25f, 0.3f,
503     0.5f, 1.25f, 0.3f,
504     1.5f, 1.25f, 0.3f,
505
506     //Cara trasera
507     0.5f, 0.25f, -0.5f,
508     1.5f, 0.25f, -0.5f,
509     0.5f, 1.25f, -0.5f,
510     1.5f, 1.25f, -0.5f,
511
512     //puerta
513     0.5f, 0.25f, 0.3f,
514     1.5f, 0.25f, 0.3f,
515     0.5f, 1.25f, 0.3f,
516     1.5f, 1.25f, 0.3f,
517
518     //Cara frontal
519     0.5f, 0.25f, 0.3f,
520     1.5f, 0.25f, 0.3f,
521     0.5f, 1.25f, 0.3f,
522     1.5f, 1.25f, 0.3f,
523
524     //Cara trasera
525     0.5f, 0.25f, -0.5f,
526     1.5f, 0.25f, -0.5f,
527     0.5f, 1.25f, -0.5f,
528     1.5f, 1.25f, -0.5f,
529
530     //puerta
531     0.5f, 0.25f, 0.3f,
532     1.5f, 0.25f, 0.3f,
533     0.5f, 1.25f, 0.3f,
534     1.5f, 1.25f, 0.3f,
535
536     //Cara frontal
537     0.5f, 0.25f, 0.3f,
538     1.5f, 0.25f, 0.3f,
539     0.5f, 1.25f, 0.3f,
540     1.5f, 1.25f, 0.3f,
541
542     //Cara trasera
543     0.5f, 0.25f, -0.5f,
544     1.5f, 0.25f, -0.5f,
545     0.5f, 1.25f, -0.5f,
546     1.5f, 1.25f, -0.5f,
547
548     //puerta
549     0.5f, 0.25f, 0.3f,
550     1.5f, 0.25f, 0.3f,
551     0.5f, 1.25f, 0.3f,
552     1.5f, 1.25f, 0.3f,
553
554     //Cara frontal
555     0.5f, 0.25f, 0.3f,
556     1.5f, 0.25f, 0.3f,
557     0.5f, 1.25f, 0.3f,
558     1.5f, 1.25f, 0.3f,
559
560     //Cara trasera
561     0.5f, 0.25f, -0.5f,
562     1.5f, 0.25f, -0.5f,
563     0.5f, 1.25f, -0.5f,
564     1.5f, 1.25f, -0.5f,
565
566     //puerta
567     0.5f, 0.25f, 0.3f,
568     1.5f, 0.25f, 0.3f,
569     0.5f, 1.25f, 0.3f,
570     1.5f, 1.25f, 0.3f,
571
572     //Cara frontal
573     0.5f, 0.25f, 0.3f,
574     1.5f, 0.25f, 0.3f,
575     0.5f, 1.25f, 0.3f,
576     1.5f, 1.25f, 0.3f,
577
578     //Cara trasera
579     0.5f, 0.25f, -0.5f,
580     1.5f, 0.25f, -0.5f,
581     0.5f, 1.25f, -0.5f,
582     1.5f, 1.25f, -0.5f,
583
584     //puerta
585     0.5f, 0.25f, 0.3f,
586     1.5f, 0.25f, 0.3f,
587     0.5f, 1.25f, 0.3f,
588     1.5f, 1.25f, 0.3f,
589
590     //Cara frontal
591     0.5f, 0.25f, 0.3f,
592     1.5f, 0.25f, 0.3f,
593     0.5f, 1.25f, 0.3f,
594     1.5f, 1.25f, 0.3f,
595
596     //Cara trasera
597     0.5f, 0.25f, -0.5f,
598     1.5f, 0.25f, -0.5f,
599     0.5f, 1.25f, -0.5f,
600     1.5f, 1.25f, -0.5f,
601
602     //puerta
603     0.5f, 0.25f, 0.3f,
604     1.5f, 0.25f, 0.3f,
605     0.5f, 1.25f, 0.3f,
606     1.5f, 1.25f, 0.3f,
607
608     //Cara frontal
609     0.5f, 0.25f, 0.3f,
610     1.5f, 0.25f, 0.3f,
611     0.5f, 1.25f, 0.3f,
612     1.5f, 1.25f, 0.3f,
613
614     //Cara trasera
615     0.5f, 0.25f, -0.5f,
616     1.5f, 0.25f, -0.5f,
617     0.5f, 1.25f, -0.5f,
618     1.5f, 1.25f, -0.5f,
619
620     //puerta
621     0.5f, 0.25f, 0.3f,
622     1.5f, 0.25f, 0.3f,
623     0.5f, 1.25f, 0.3f,
624     1.5f, 1.25f, 0.3f,
625
626     //Cara frontal
627     0.5f, 0.25f, 0.3f,
628     1.5f, 0.25f, 0.3f,
629     0.5f, 1.25f, 0.3f,
630     1.5f, 1.25f, 0.3f,
631
632     //Cara trasera
633     0.5f, 0.25f, -0.5f,
634     1.5f, 0.25f, -0.5f,
635     0.5f, 1.25f, -0.5f,
636     1.5f, 1.25f, -0.5f,
637
638     //puerta
639     0.5f, 0.25f, 0.3f,
640     1.5f, 0.25f, 0.3f,
641     0.5f, 1.25f, 0.3f,
642     1.5f, 1.25f, 0.3f,
643
644     //Cara frontal
645     0.5f, 0.25f, 0.3f,
646     1.5f, 0.25f, 0.3f,
647     0.5f, 1.25f, 0.3f,
648     1.5f, 1.25f, 0.3f,
649
650     //Cara trasera
651     0.5f, 0.25f, -0.5f,
652     1.5f, 0.25f, -0.5f,
653     0.5f, 1.25f, -0.5f,
654     1.5f, 1.25f, -0.5f,
655
656     //puerta
657     0.5f, 0.25f, 0.3f,
658     1.5f, 0.25f, 0.3f,
659     0.5f, 1.25f, 0.3f,
660     1.5f, 1.25f, 0.3f,
661
662     //Cara frontal
663     0.5f, 0.25f, 0.3f,
664     1.5f, 0.25f, 0.3f,
665     0.5f, 1.25f, 0.3f,
666     1.5f, 1.25f, 0.3f,
667
668     //Cara trasera
669     0.5f, 0.25f, -0.5f,
670     1.5f, 0.25f, -0.5f,
671     0.5f, 1.25f, -0.5f,
672     1.5f, 1.25f, -0.5f,
673
674     //puerta
675     0.5f, 0.25f, 0.3f,
676     1.5f, 0.25f, 0.3f,
677     0.5f, 1.25f, 0.3f,
678     1.5f, 1.25f, 0.3f,
679
680     //Cara frontal
681     0.5f, 0.25f, 0.3f,
682     1.5f, 0.25f, 0.3f,
683     0.5f, 1.25f, 0.3f,
684     1.5f
```

P02-319218460.cpp:1\* shader.vert

```

102     4, 0, 3,
103     3, 7, 4,
104     // bottom
105     4, 5, 1,
106     1, 0, 4,
107     // top
108     3, 2, 6,
109     6, 7, 3
110 };
111
112 GLfloat puerta_vertices[] = {
113     //Cara trasera puerta
114     0.875f, 0.25f, 0.3f,
115     1.125f, 0.25f, 0.3f,
116     0.875f, 0.375f, 0.3f,
117     1.125f, 0.375f, 0.3f,
118     //Cara frontal puerta
119     0.875f, 0.25f, 0.5f,
120     1.125f, 0.25f, 0.5f,
121     0.875f, 0.375f, 0.5f,
122     1.125f, 0.375f, 0.5f,
123 };
124
125 Mesh* puerta = new Mesh();
126 puerta->CreateMesh(puerta_vertices, puerta_indices, 24);
127 meshList.push_back(puerta);
128
129 unsigned int ventanaIzq_indices[] = {
130     // front
131     0, 1, 2,
132     2, 3, 0,
133     // right
134     1, 5, 6,
135     6, 2, 1,
136     // back
137     7, 6, 5,
138     5, 4, 7,
139     // left
140     4, 0, 3,
141     3, 7, 4,
142     // bottom
143     4, 5, 1,
144     1, 0, 4,
145     // top
146     3, 2, 6,
147     6, 7, 3
148 };
149
150 GLfloat ventanaIzq_vertices[] = {
151     //Cara trasera ventana izquierda
152     0.625f, 0.875f, 0.3f,
153     0.875f, 0.875f, 0.3f,
154     0.625f, 1.125f, 0.3f,
155     0.875f, 1.125f, 0.3f,
156     //Cara frontal ventana izquierda
157     0.625f, 0.875f, 0.5f,
158     0.875f, 0.875f, 0.5f,
159     0.625f, 1.125f, 0.5f,
160     0.875f, 1.125f, 0.5f,
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223

```

P02-319218460.cpp:1\* shader.vert

```

164 Mesh* ventanaIzq = new Mesh();
165 ventanaIzq->CreateMesh(ventanaIzq_vertices, ventanaIzq_indices, 24);
166 meshList.push_back(ventanaIzq);
167
168
169 unsigned int ventanaDer_indices[] = {
170     // front
171     0, 1, 2,
172     2, 3, 0,
173     // right
174     1, 5, 6,
175     6, 2, 1,
176     // back
177     7, 6, 5,
178     5, 4, 7,
179     // left
180     4, 0, 3,
181     3, 7, 4,
182     // bottom
183     4, 5, 1,
184     1, 0, 4,
185     // top
186     3, 2, 6,
187     6, 7, 3
188 };
189
190 GLfloat ventanaDer_vertices[] = {
191     //Cara trasera ventana derecha
192     1.125f, 0.875f, 0.3f,
193     1.375f, 0.875f, 0.3f,
194     1.125f, 1.125f, 0.3f,
195     1.375f, 1.125f, 0.3f,
196     //Cara frontal ventana derecha
197     1.125f, 0.875f, 0.5f,
198     1.375f, 0.875f, 0.5f,
199     1.125f, 1.125f, 0.5f,
200     1.375f, 1.125f, 0.5f,
201     //Cara trasera tronco izquierdo
202     0.625f, 0.25f, -0.1f,
203     0.2f, 0.25f, -0.1f,
204     0.05f, 0.5f, -0.1f,
205     0.2f, 0.5f, -0.1f,
206     //Cara frontal tronco izquierdo
207     0.05f, 0.25f, 0.1f,
208     0.2f, 0.25f, 0.1f,
209     0.05f, 0.5f, 0.1f,
210     0.2f, 0.5f, 0.1f,
211     //Cara trasera tronco derecho
212     1.8f, 0.25f, -0.1f,
213     1.95f, 0.25f, -0.1f,
214     1.8f, 0.5f, -0.1f,
215     1.95f, 0.5f, -0.1f,
216     //Cara frontal tronco derecho
217     1.8f, 0.25f, 0.1f,
218     1.95f, 0.25f, 0.1f,
219     1.8f, 0.5f, 0.1f,
220     1.95f, 0.5f, 0.1f,
221     //Cara trasera casa
222     0.625f, 1.125f, 0.3f,
223     0.875f, 1.125f, 0.3f,
224     0.625f, 1.125f, 0.5f,
225     0.875f, 1.125f, 0.5f,
226     0.625f, 1.125f, 0.7f,
227     0.875f, 1.125f, 0.7f,
228     //Cara frontal casa
229     0.625f, 1.125f, 0.3f,
230     0.875f, 1.125f, 0.3f,
231     0.625f, 1.125f, 0.5f,
232     0.875f, 1.125f, 0.5f,
233     0.625f, 1.125f, 0.7f,
234     0.875f, 1.125f, 0.7f
235     //Cara trasera casa
236     0.625f, 1.125f, 0.3f,
237     0.875f, 1.125f, 0.3f,
238     0.625f, 1.125f, 0.5f,
239     0.875f, 1.125f, 0.5f,
240     0.625f, 1.125f, 0.7f,
241     0.875f, 1.125f, 0.7f
242     //Cara frontal casa
243     0.625f, 1.125f, 0.3f,
244     0.875f, 1.125f, 0.3f,
245     0.625f, 1.125f, 0.5f,
246     0.875f, 1.125f, 0.5f,
247     0.625f, 1.125f, 0.7f,
248     0.875f, 1.125f, 0.7f
249     //Cara trasera casa
250     0.625f, 1.125f, 0.3f,
251     0.875f, 1.125f, 0.3f,
252     0.625f, 1.125f, 0.5f,
253     0.875f, 1.125f, 0.5f,
254     0.625f, 1.125f, 0.7f,
255     0.875f, 1.125f, 0.7f
256     //Cara frontal casa
257     0.625f, 1.125f, 0.3f,
258     0.875f, 1.125f, 0.3f,
259     0.625f, 1.125f, 0.5f,
260     0.875f, 1.125f, 0.5f,
261     0.625f, 1.125f, 0.7f,
262     0.875f, 1.125f, 0.7f
263     //Cara trasera casa
264     0.625f, 1.125f, 0.3f,
265     0.875f, 1.125f, 0.3f,
266     0.625f, 1.125f, 0.5f,
267     0.875f, 1.125f, 0.5f,
268     0.625f, 1.125f, 0.7f,
269     0.875f, 1.125f, 0.7f
270     //Cara frontal casa
271     0.625f, 1.125f, 0.3f,
272     0.875f, 1.125f, 0.3f,
273     0.625f, 1.125f, 0.5f,
274     0.875f, 1.125f, 0.5f,
275     0.625f, 1.125f, 0.7f,
276     0.875f, 1.125f, 0.7f
277     //Cara trasera casa
278     0.625f, 1.125f, 0.3f,
279     0.875f, 1.125f, 0.3f,
280     0.625f, 1.125f, 0.5f,
281     0.875f, 1.125f, 0.5f

```

P02-319218460.cpp:1\* shader.vert

```

223     4, 5, 1,
224     1, 0, 4,
225     // top
226     3, 2, 6,
227     6, 7, 3
228 };
229
230 GLfloat troncoIzq_vertices[] = {
231     //Cara trasera tronco izquierdo
232     0.05f, 0.25f, -0.1f,
233     0.2f, 0.25f, -0.1f,
234     0.05f, 0.5f, -0.1f,
235     0.2f, 0.5f, -0.1f,
236     //Cara frontal tronco izquierdo
237     0.05f, 0.25f, 0.1f,
238     0.2f, 0.25f, 0.1f,
239     0.05f, 0.5f, 0.1f,
240     0.2f, 0.5f, 0.1f,
241 };
242
243 Mesh* troncoIzq = new Mesh();
244 troncoIzq->CreateMesh(troncoIzq_vertices, troncoIzq_indices, 24);
245 meshList.push_back(troncoIzq);
246
247
248 unsigned int troncoDer_indices[] = {
249     //Cara trasera tronco derecho
250     1.8f, 0.25f, -0.1f,
251     1.95f, 0.25f, -0.1f,
252     1.8f, 0.5f, -0.1f,
253     1.95f, 0.5f, -0.1f,
254     //Cara frontal tronco derecho
255     1.8f, 0.25f, 0.1f,
256     1.95f, 0.25f, 0.1f,
257     1.8f, 0.5f, 0.1f,
258     1.95f, 0.5f, 0.1f,
259     //Cara trasera casa
260     0.625f, 1.125f, 0.3f,
261     0.875f, 1.125f, 0.3f,
262     0.625f, 1.125f, 0.5f,
263     0.875f, 1.125f, 0.5f,
264     0.625f, 1.125f, 0.7f,
265     0.875f, 1.125f, 0.7f
266     //Cara frontal casa
267     0.625f, 1.125f, 0.3f,
268     0.875f, 1.125f, 0.3f,
269     0.625f, 1.125f, 0.5f,
270     0.875f, 1.125f, 0.5f,
271     0.625f, 1.125f, 0.7f,
272     0.875f, 1.125f, 0.7f
273     //Cara trasera casa
274     0.625f, 1.125f, 0.3f,
275     0.875f, 1.125f, 0.3f,
276     0.625f, 1.125f, 0.5f,
277     0.875f, 1.125f, 0.5f,
278     0.625f, 1.125f, 0.7f,
279     0.875f, 1.125f, 0.7f
280     //Cara frontal casa
281     0.625f, 1.125f, 0.3f,
282     0.875f, 1.125f, 0.3f,
283     0.625f, 1.125f, 0.5f,
284     0.875f, 1.125f, 0.5f,
285     0.625f, 1.125f, 0.7f,
286     0.875f, 1.125f, 0.7f

```

P02-319218460.cpp:1\* shader.vert GitHub Copilot

```
Practica2 (Ámbito global) crearCasa()
284     Mesh* troncoDer = new Mesh();
285     troncoDer->CreateMesh(troncoDer_vertices, troncoDer_indices);
286     meshList.push_back(troncoDer);
287
288     unsigned int techo_indices[] = {
289         // Triángulo frontal
290         0, 1, 2,
291
292         // Triángulo trasero
293         3, 5, 4,
294
295         // Lado izquierdo
296         0, 2, 3,
297         3, 2, 5,
298
299         // Lado derecho
300         1, 4, 2,
301         2, 4, 5,
302
303         // Base
304         0, 3, 1,
305         1, 3, 4
306     };
307     GLfloat techo_vertices[] = {
308         // Base frontal
309         0.25f, 1.25f, 0.5f,
310
311         1.75f, 1.25f, 0.5f,
312         1.00f, 2.00f, 0.5f,
313
314         // Base trasera
315         0.25f, 1.25f, -0.5f,
316         1.75f, 1.25f, -0.5f,
317         1.00f, 2.00f, -0.5f
318     };
319
320     Mesh* techo = new Mesh();
321     techo->CreateMesh(techo_vertices, techo_indices, 36, 18);
322     meshList.push_back(techo);
323
324
325     unsigned int copaIzq_indices[] = {
326         0, 1, 2, // Triángulo frontal
327         3, 5, 4, // Triángulo trasero
328         0, 2, 3, 3, 2, 5, // Lado izquierdo
329         1, 4, 2, 2, 4, 5, // Lado derecho
330         0, 3, 1, 1, 3, 4 // Base cuadrada
331     };
332
333     GLfloat copaIzq_vertices[] = {
334         0.0f, 0.5f, 0.3f,
335         0.25f, 0.5f, 0.3f,
336         0.125f, 1.0f, 0.3f,
337
338         0.0f, 0.5f, -0.3f,
339         0.25f, 0.5f, -0.3f,
340         0.125f, 1.0f, -0.3f
341     };
342
343     Mesh* copaIzq = new Mesh();
344     copaIzq->CreateMesh(copaIzq_vertices, copaIzq_indices,
345     meshList.push_back(copaIzq);
346
347
348     unsigned int copaDer_indices[] = {
349         0, 1, 2,
350         3, 5, 4,
351         0, 2, 3, 3, 2, 5,
352         1, 4, 2, 2, 4, 5,
353         0, 3, 1, 1, 3, 4
354     };
355
356     GLfloat copaDer_vertices[] = {
357         1.75f, 0.5f, 0.3f,
358         2.00f, 0.5f, 0.3f,
359         1.875f, 1.0f, 0.3f,
360
361         1.75f, 0.5f, -0.3f,
362         2.00f, 0.5f, -0.3f,
363         1.875f, 1.0f, -0.3f
364     };
365
366     Mesh* copaDer = new Mesh();
367     copaDer->CreateMesh(copaDer_vertices, copaDer_indices,
368     meshList.push_back(copaDer);
369 }
```

```
343     Mesh* copaIzq = new Mesh();
344     copaIzq->CreateMesh(copaIzq_vertices, copaIzq_indices,
345     meshList.push_back(copaIzq);
346
347
348     unsigned int copaDer_indices[] = {
349         0, 1, 2,
350         3, 5, 4,
351         0, 2, 3, 3, 2, 5,
352         1, 4, 2, 2, 4, 5,
353         0, 3, 1, 1, 3, 4
354     };
355
356     GLfloat copaDer_vertices[] = {
357         1.75f, 0.5f, 0.3f,
358         2.00f, 0.5f, 0.3f,
359         1.875f, 1.0f, 0.3f,
360
361         1.75f, 0.5f, -0.3f,
362         2.00f, 0.5f, -0.3f,
363         1.875f, 1.0f, -0.3f
364     };
365
366     Mesh* copaDer = new Mesh();
367     copaDer->CreateMesh(copaDer_vertices, copaDer_indices,
368     meshList.push_back(copaDer);
369 }
```

## Líneas de código que muestran la función del dibujado de las letras.

This screenshot shows a code editor with three panes. The left pane displays C++ code for generating vertex positions for a letter 'A'. The middle pane shows the vertex coordinates themselves. The right pane shows the continuation of the vertex list. The code uses GLfloat arrays to store vertex coordinates, separated by sections of text.

```

374 void ClearFiguras()
375 {
376     srand(static_cast<unsigned int>(time(0)));
377     GLfloat pos_letra_A[] = {
378         // Triángulo lateral izquierdo inferior izquierdo
379         -1.0f, -0.5f, 0.0f,
380         -0.9f, 0.0f, 0.0f,
381         -0.76f, -0.0f, 0.0f,
382
383         // Triángulo lateral izquierdo inferior izquierdo
384         -1.0f, -0.5f, 0.0f,
385         -0.8f, -0.5f, 0.0f,
386         -0.76f, -0.0f, 0.0f,
387
388         // Triángulo lateral derecho inferior derecho
389         -0.7f, -0.5f, 0.0f,
390         -0.74f, 0.0f, 0.0f,
391         -0.5f, -0.5f, 0.0f,
392
393         // Triángulo lateral derecho inferior derecho izq
394         -0.6f, -0.0f, 0.0f,
395         -0.74f, 0.0f, 0.0f,
396         -0.5f, -0.5f, 0.0f,
397
398         //Partes superiores del triangulo de la sección 2
399         -0.9f, -0.0f, 0.0f,
400     };

```

```

401         -0.85f, 0.25f, 0.0f,
402         -0.75f, 0.25f, 0.0f,
403
404         //texto para separar las secciones
405         -0.9f, -0.0f, 0.0f,
406         -0.75f, 0.0f, 0.0f,
407         -0.75f, 0.25f, 0.0f,
408
409         //Partes superiores del triangulo de la sección 2
410         -0.6f, -0.0f, 0.0f,
411         -0.6f, 0.25f, 0.0f,
412         -0.75f, 0.25f, 0.0f,
413
414         //texto para separar las secciones
415         -0.6f, -0.0f, 0.0f,
416         -0.75f, 0.0f, 0.0f,
417         -0.75f, 0.25f, 0.0f,
418
419         //Partes superiores del triangulo de la sección 3
420         -0.85f, 0.25f, 0.0f,
421         -0.76f, 0.25f, 0.0f,
422         -0.75f, 0.3f, 0.0f,
423
424         //texto para separar las secciones
425         -0.65f, 0.25f, 0.0f,
426         -0.74f, 0.25f, 0.0f,
427         -0.75f, 0.3f, 0.0f,
428
429         //Partes superiores del triangulo de la sección 4
430         -0.85f, 0.25f, 0.0f,
431         -0.8f, 0.5f, 0.0f,
432
433     };

```

This screenshot shows a code editor with three panes. The left pane displays C++ code for generating vertex positions for a letter 'L'. The middle pane shows the vertex coordinates themselves. The right pane shows the continuation of the vertex list. The code uses GLfloat arrays to store vertex coordinates, separated by sections of text.

```

434
435     -0.75f, 0.3f, 0.0f,
436
437         //texto para separar las secciones
438         -0.65f, 0.25f, 0.0f,
439         -0.7f, 0.5f, 0.0f,
440         -0.75f, 0.3f, 0.0f,
441
442         //Partes superiores del triangulo de la sección 5
443         -0.7f, 0.5f, 0.0f,
444         -0.8f, 0.5f, 0.0f,
445         -0.75f, 0.3f, 0.0f,
446
447     int numVerticesA = sizeof(pos_letra_A) / (3 * sizeof(GLfloat));
448     GLfloat* letra_A = new GLfloat[numVerticesA * 6];
449     for (int i = 0; i < numVerticesA; i++) {
450         // Copiamos las coordenadas de posición
451         letra_A[i * 6 + 0] = pos_letra_A[i * 3 + 0];
452         letra_A[i * 6 + 1] = pos_letra_A[i * 3 + 1];
453         letra_A[i * 6 + 2] = pos_letra_A[i * 3 + 2];
454         // Asignamos colores aleatorios
455         letra_A[i * 6 + 3] = randomColor();
456         letra_A[i * 6 + 4] = randomColor();
457         letra_A[i * 6 + 5] = randomColor();
458
459     MeshColor* meshLetra_A = new MeshColor();
460     meshLetra_A->CreateMeshColor(letra_A, numVerticesA * 6);
461
462     meshColorList.push_back(meshLetra_A);

```

```

463
464
465     // Vértices de la letra L
466     GLfloat pos_letra_L[] = {
467         //Partes del triangulo de la sección 1
468         -0.25f, -0.5f, 0.0f,
469         -0.0f, 0.5f, 0.0f,
470         -0.25f, 0.5f, 0.0f,
471
472         //texto para separar las secciones
473         -0.25f, -0.5f, 0.0f,
474         0.0f, 0.5f, 0.0f,
475         -0.0f, -0.5f, 0.0f,
476
477         //Partes del triangulo de la sección 2
478         -0.0f, -0.25f, 0.0f,
479         0.25f, -0.5f, 0.0f,
480         -0.0f, -0.5f, 0.0f,
481
482         //texto para separar las secciones
483         -0.0f, -0.25f, 0.0f,
484         0.25f, -0.5f, 0.0f,
485         0.25f, -0.25f, 0.0f,
486
487     int numVerticesL = sizeof(pos_letra_L) / (3 * sizeof(GLfloat));
488     GLfloat* letra_L = new GLfloat[numVerticesL * 6];
489     for (int i = 0; i < numVerticesL; i++) {
490         // Copiamos las coordenadas de posición
491         letra_L[i * 6 + 0] = pos_letra_L[i * 3 + 0];
492         letra_L[i * 6 + 1] = pos_letra_L[i * 3 + 1];
493         letra_L[i * 6 + 2] = pos_letra_L[i * 3 + 2];

```

This screenshot shows a code editor with three panes. The left pane displays C++ code for generating vertex positions for a letter 'M'. The middle pane shows the vertex coordinates themselves. The right pane shows the continuation of the vertex list. The code uses GLfloat arrays to store vertex coordinates, separated by sections of text.

```

494
495     // Asignamos colores aleatorios
496     letra_L[i * 6 + 3] = randomColor();
497     letra_L[i * 6 + 4] = randomColor();
498     letra_L[i * 6 + 5] = randomColor();
499
500     MeshColor* meshLetra_L = new MeshColor();
501     meshLetra_L->CreateMeshColor(letra_L, numVerticesL * 6);
502     meshColorList.push_back(meshLetra_L);
503
504     // Vértices de la letra M
505     GLfloat pos_letra_M[] = {
506         //Partes del triangulo de la sección 1 izquierda
507         0.5f, -0.5f, 0.0f,
508         0.5f, 0.5f, 0.0f,
509         0.625f, -0.5f, 0.0f,
510
511         //texto para separar las secciones
512         0.625f, 0.5f, 0.0f,
513         0.5f, 0.5f, 0.0f,
514         0.625f, -0.5f, 0.0f,
515
516         //Partes del triangulo de la sección 1 derecha
517         0.875f, 0.5f, 0.0f,
518         0.875f, -0.5f, 0.0f,
519         1.0f, 0.5f, 0.0f,
520
521         //texto para separar las secciones
522         1.0f, -0.5f, 0.0f,
523         0.875f, -0.5f, 0.0f,
524         1.0f, 0.5f, 0.0f,
525
526         //Partes del triangulo de la sección central inferior
527         0.625f, 0.125f, 0.0f,
528         0.7f, 0.0f, 0.0f,
529         0.7f, 0.125f, 0.0f,
530
531         //texto para separar las secciones
532         0.75f, 0.0625f, 0.0f,
533         0.7f, -0.0f, 0.0f,
534         0.7f, 0.125f, 0.0f,
535
536         //texto para separar las secciones
537         0.75f, 0.0625f, 0.0f,
538         0.7f, 0.0f, 0.0f,
539         0.8f, 0.0f, 0.0f,
540
541         //texto para separar las secciones
542         0.75f, 0.0625f, 0.0f,
543         0.8f, 0.125f, 0.0f,
544         0.7f, 0.125f, 0.0f,
545
546         //texto para separar las secciones
547         0.75f, 0.0625f, 0.0f,
548         0.8f, 0.125f, 0.0f,
549         0.8f, 0.0f, 0.0f,
550
551         //texto para separar las secciones

```

```

553     0.875f, 0.125f, 0.0f,
554     0.8f, 0.125f, 0.0f,
555     0.8f, 0.0f, 0.0f,
556
557     //Partes del triángulo de la sección central superior
558     0.625f, 0.125f, 0.0f,
559     0.875f, 0.125f, 0.0f,
560     0.75f, 0.25f, 0.0f,
561
562     //Texto para separar las secciones
563     0.625f, 0.125f, 0.0f,
564     0.625f, 0.5f, 0.0f,
565     0.75f, 0.25f, 0.0f,
566
567     //Texto para separar las secciones
568     0.875f, 0.5f, 0.0f,
569     0.875f, 0.125f, 0.0f,
570     0.75f, 0.25f, 0.0f,
571
572 };
573 int numVerticesM = sizeof(pos_letra_M) / (3 * sizeof(GLfloat));
574 GLfloat* letra_M = new GLfloat[numVerticesM * 6];
575 for (int i = 0; i < numVerticesM; i++) {
576     // Copiamos las coordenadas de posición
577     letra_M[i * 6 + 0] = pos_letra_M[i * 3 + 0];
578     letra_M[i * 6 + 1] = pos_letra_M[i * 3 + 1];
579     letra_M[i * 6 + 2] = pos_letra_M[i * 3 + 2];
580
581     // Asignamos colores aleatorios
582     letra_M[i * 6 + 3] = randomColor();
583     letra_M[i * 6 + 4] = randomColor();
584     letra_M[i * 6 + 5] = randomColor();
585
586     MeshColor* meshLetra_M = new MeshColor();
587     meshLetra_M->CreateMeshColor(letra_M, numVerticesM * 6);
588     meshColorList.push_back(meshLetra_M);
589
590 }
591
592 void id CreateShaders()
593 {
594     Shader* shader1 = new Shader(); // shader para usar índices: objetos
595     shader1->CreateFromFiles(vShader, fShader);
596     shaderList.push_back(*shader1);
597
598     Shader* shader2 = new Shader(); // shader para usar color como parámetro
599     shader2->CreateFromFiles(vShaderColor, fShaderColor);
600     shaderList.push_back(*shader2);
601
602     t main()
603 }

```

Modificación del Main para la inclusión de el dibujado de letras y creación de la casa, así como su impresión.

En el caso de la casa se imprimió con el index de color asignado en el shader.vert.

```

610     int main()
611     {
612         mainWindow = Window(800, 600);
613         mainWindow.Initialise();
614
615         CreateShaders();
616
617         crearCasa();
618
619         CrearFiguras();
620
621         GLEventLoop();
622
623         GLEventLoop();
624
625         GLuint uniformProjection = 0;
626         GLuint uniformModel = 0;
627         // Projection: Matriz de Dimensión 4x4 para indicar si
628         // glm::mat4 projection = glm::ortho(-1.0f, 1.0f, -1.0f
629         glm::mat4 projection = glm::perspective(glm::radians(45.0f),
630             1.0f, 10.0f);
631         // Model: Matriz de Dimensión 4x4 en la cual se almacena la
632         // glm::mat4 model(1.0); // fuera del while se usa para ir
633         // Loop mientras no se cierra la ventana
634         while (!mainWindow.getShouldClose())
635         {
636
637             // Recibir eventos del usuario
638             glfwPollEvents();
639             // Limpiar la ventana
640             glClearColor(0.0f, 0.0f, 0.0f, 1.0f);
641             glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT); // Se agraga
642
643             // Se puede usar una transformación común para estos objetos
644             model = glm::mat4(1.0);
645             model = glm::translate(model, glm::vec3(0.0f, 0.0f, -4.0f));
646             glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
647             glUniformMatrix4fv(uniformProjection, 1, GL_FALSE, glm::value_ptr(projection));
648
649             // Rendizar cada objeto con un 'colorIndex' específico
650             for (size_t i = 0; i < meshList.size(); i++) {
651                 if (i == 0) glUniform1i(uniformColorIndex, 0); // Casa
652                 else if (i == 1) glUniform1i(uniformColorIndex, 1); // Objeto
653                 else if (i == 2) glUniform1i(uniformColorIndex, 1); // Objeto
654                 else if (i == 3) glUniform1i(uniformColorIndex, 1); // Objeto
655                 else if (i == 4) glUniform1i(uniformColorIndex, 4); // Objeto
656                 else if (i == 5) glUniform1i(uniformColorIndex, 4); // Objeto
657                 else if (i == 6) glUniform1i(uniformColorIndex, 2); // Objeto
658                 else if (i == 7) glUniform1i(uniformColorIndex, 2); // Objeto
659                 else if (i == 8) glUniform1i(uniformColorIndex, 2); // Objeto
660                 else if (i == 9) glUniform1i(uniformColorIndex, 3); // Objeto
661             }
662
663             // Dibujar la casa
664             crearCasa();
665
666             // Dibujar las figuras
667             CrearFiguras();
668
669         }
670     }

```

```

        else if (i == 2) glUniform1i(uniformColorIndex, 1); // Ventana Izq (verde claro)
        else if (i == 3) glUniform1i(uniformColorIndex, 1); // Ventana Der (verde claro)
        else if (i == 4) glUniform1i(uniformColorIndex, 4); // Tronco Izq (café)
        else if (i == 5) glUniform1i(uniformColorIndex, 4); // Tronco Der (café)
        else if (i == 7) glUniform1i(uniformColorIndex, 2); // Copa Izq (verde oscuro)
        else if (i == 8) glUniform1i(uniformColorIndex, 2); // Copa Der (verde oscuro)
        else if (i == 6) glUniform1i(uniformColorIndex, 3); // Techo (azul)

    meshList[i]->RenderMesh();
}

//Para las letras hay que usar el segundo set de shaders con indice 1 en ShaderList
shaderList[1].useShader();
uniformModel = shaderList[1].getModelLocation();
uniformProjection = shaderList[1].getProjectLocation();

//Inicializar matriz de dimensión 4x4 que servirá como matriz de modelo para almacenar las transformaciones geométricas
model = glm::mat4(1.0);
model = glm::translate(model, glm::vec3(-1.0f, -1.0f, -4.0f));
//
glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));//FALSE ES PARA QUE NO SEA TRANSPUESTA y se envían al shader
glUniformMatrix4fv(uniformProjection, 1, GL_FALSE, glm::value_ptr(projection));
meshColorList[0]->RenderMeshColor();
for (size_t i = 1; i < meshColorList.size(); i++) {
    meshColorList[i]->RenderMeshColor();
}

```

## Ejecución:



## **Conclusión:**

Con lo aprendido se ha adquirido el conocimiento para hacer modelado 3D, aunque se tuvo que hacer más investigación de la esperada, los conceptos aprendidos como traslación y rotación (aunque no se puso en práctica en este ejercicio) han sido conceptos útiles junto con la comprensión de figuras en 3d, aún se debe de hacer énfasis que se debe de buscar alguna alternativa para facilitar este tipo de trabajos, esto debido a que la práctica que había hecho antes en trabajos anteriores no ha sido la mejor en este caso.

Aún hay mucho que aprender del tema para encontrar una forma en la que la forma visual sea más agradable a la vista y el proceso de trabajo sea más eficiente,

## **Bibliografía:**

1. Tutorial 4 : Un cubo con color. (s. f.). [https://www.opengl-tutorial.org/es/beginners-tutorials/tutorial-4-a-colored-cube/?utm\\_source=chatgpt.com](https://www.opengl-tutorial.org/es/beginners-tutorials/tutorial-4-a-colored-cube/?utm_source=chatgpt.com)
2. Kurisuto. (2017, 24 noviembre).  -♦-Representación de objetos 3D y texturizado en OpenGL [Vídeo]. YouTube.  
<https://www.youtube.com/watch?v=EkXIsI-CSlc>
3. Curso de introducción a OpenGL (v1.0). (1999).  
[https://www.cs.buap.mx/~hilario\\_sm/slides/graficacion/Manual-opengl.pdf?utm\\_source=chatgpt.com](https://www.cs.buap.mx/~hilario_sm/slides/graficacion/Manual-opengl.pdf?utm_source=chatgpt.com)