



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

DIVISIÓN DE INGENIERÍA ELÉCTRICA

INGENIERÍA EN COMPUTACIÓN

LABORATORIO DE COMPUTACIÓN GRÁFICA e
INTERACCIÓN HUMANO COMPUTADORA



REPORTE DE PRÁCTICA N° 09

NOMBRE COMPLETO: LOPEZ BETANCOURT MICHELLE

N° de Cuenta: 318309028

GRUPO DE LABORATORIO: 02

GRUPO DE TEORÍA: 06

SEMESTRE 2025-1

FECHA DE ENTREGA LÍMITE: 19 DE OCTUBRE DE 2024

CALIFICACIÓN: _____

REPORTE DE PRÁCTICA:

1.- Ejecución de los ejercicios que se dejaron, comentar cada uno y capturas de pantalla de bloques de código generados y de ejecución del programa.

- Para esta práctica se tenían que realizar dos ejercicios, los cuales consistían en que mediante el dado de 10 caras realizado anteriormente se debía lograr que cayera al piso, girara y mostrara un numero "random", haciendo que la tirada del dado se repitiera al presionar alguna tecla en el teclado. El segundo ejercicio era que por integrante del equipo se debía elegir un tipo de vehículo ya fuera terrestre o aéreo, cada uno de nosotros debíamos crear un recorrido en donde el vehículo elegido se moviera alrededor de nuestro tablero de monopoly, se debía tomar en cuenta que cada uno iniciaría a partir de una esquina diferente y además el modelo escogido debía tener movimiento de llantas o de hélices. En caso de elegir un vehículo terrestre este no podía volver a ser un carro o similar a uno motorizado de 4 ruedas.

Código generado

```
8      Adicional.- ,Textura Animada
9      */
10     //para cargar imagen
11     #define STB_IMAGE_IMPLEMENTATION
12
13     #include <stdio.h>
14     #include <string.h>
15     #include <cmath>
16     #include <vector>
17     #include <math.h>
18     #include <stdlib.h>
19     #include <time.h>
20
21     #include <glew.h>
22     #include <glfw3.h>
23
24     #include <glm.hpp>
25     #include <glm/gtc/matrix_transform.hpp>
```

```

56 float offsetCartel;
57
58 //dado
59 float movimientoD;
60 float movimientoDOffset;
61 float girarX;
62 float girarY;
63 float girarZ;
64 bool cae;
65
66 //moto
67 bool avanzaMoto;
68 float movimientoMoto;
69
70 float movimientoMoto2;
71 float movimientoMotoOffset;
72 float rotllantamoto;
73 float rotllantamotoOffset;
74 float moviMoto;
75
76 int generarNumeroAleatorio;
77
78 Window mainWindow;
79 std::vector<Mesh*> meshList;

```

```
Texture pisoTexture;
```

```

v //Model arco_M;
  //Model cartel_M;
Model dado10_M;
Model cuerpomoto_M;
Model llantamoto_M;
Model mapa_M;

Skybox skybox;

//materiales
Material Material_brillante

```

```

247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1
```

```

348
349
350     GLfloat now = glfwGetTime();
351     deltaTime = now - lastTime;
352     deltaTime += (now - lastTime) / limitFPS;
353     lastTime = now;
354
355     //para los num aleatorios
356     if (mainWindow.getKeys()[GLFW_KEY_R]) {
357         generarNumeroAleatorio = rand() % 10 + 1;
358         girarX = girarY = girarZ = movimientoD = 0.0;
359         // Mostrar el número generado
360         printf("Número entre 1 y 10: %d\n", generarNumeroAleatorio);
361     }
362
363     //casos para caras del dado mediante numero aleatorio
364     if (glfwGetTime() > 25) {
365         if (movimientoD > -53.8) { //donde inicia dado (arriba)
366             cae = true;
367             movimientoD -= movimientoDOffset * deltaTime;
368         }
369     }
370     switch (generarNumeroAleatorio) {

```

```

62
63 //casos para caras del dado mediante numero aleatorio
64 if (glfwGetTime() > 25) {
65     if (movimientoD > -53.8) { //donde inicia dado (arriba)
66         cae = true;
67         movimientoD -= movimientoDOffset * deltaTime;
68     }
69 }
70 switch (generarNumeroAleatorio) {
71 case 1:
72     if (glfwGetTime() > 25) {
73         if (cae) {
74             if (girarX <= 46 && girarY <= 53) {
75                 girarX += 3.0 * deltaTime;
76                 girarY += 3.44 * deltaTime;
77             }
78         }
79     }
80     break;
81 case 2:
82     if (glfwGetTime() > 25) {
83         if (cae) {
84             if (girarX <= 46) {
85                 girarX += 2.0 * deltaTime;
86                 cae2 = true;
87                 if (cae2) {
88                     if (girarZ >= -20) {
89                         girarZ -= 2.0 * deltaTime;
90                     }
91                 }
92             }
93         }
94     } /*
95     girarX = 46.0;
96     girarY = -18.0;
97     girarZ = 0.0;*/
98     break;
99
100 case 3:
101     if (glfwGetTime() > 25) {

```

```

402         if (cae) {
403             if (girarY <= 35) {
404                 girarY += 2.0 * deltaTime;
405                 cae2 = true;
406                 if (cae2) {
407                     if (girarZ >= -48) {
408                         girarZ -= 8.0 * deltaTime;
409                     }
410                 }
411             }
412         }
413     }/*
414     girarY = 18.0;
415     girarZ = -48.0;*/
416     break;
417
418 case 4:
419     if (glfwGetTime() > 25) {
420         if (cae) {
421             if (girarX >= -47) {
422                 girarX -= 2.0 * deltaTime;
423                 cae2 = true;
424                 if (cae2) {
425                     if (girarY <= 20) {
426                         girarY += 2.0 * deltaTime;
427                     }
428                 }
429             }
430         }
431     }
432     /*girarX = -47.0;
433     girarY = 20.0;*/
434     break;
435
436 case 5:
437     if (glfwGetTime() > 25) {
438         if (cae) {
439             if (girarX >= -45) {
440                 girarX -= 2.0 * deltaTime;
441                 cae2 = true;

```

✓ No se encontraron problemas

Pratiche3 (Ambito global)

```
442     if (cae2) {
443         if (girarY >= -54) {
444             girarY -= 2.5 * deltaTime;
445         }
446     }
447 }
448 }
449 */
450 girarX = -45.0;
451 girarY = -54.0;*/
452 break;
453
454 case 6:
455     /*if (glfwGetTime() > 25) {
456         if (cae) {
457             if (girarX <= 134) {
458                 girarX += 6.0 * deltaTime;
459                 cae2 = true;
460                 if (cae2) {
461                     if (girarY <= 20) {
462                         girarY += 1.0 * deltaTime;
463                     }
464                 }
465             }
466         }
467     }*/
468     girarX = 134.0;
469     girarY = 20.0;
470     break;
471
472 case 7:
473     /*if (glfwGetTime() > 25) {
474         if (cae) {
475             if (girarX <= 137) {
476                 girarX += 6.0 * deltaTime;
477                 cae2 = true;
478                 if (cae2) {
479                     if (girarY >= -55) {
480                         girarY -= 2.0 * deltaTime;
481                     }

```



```

484     }
485     */
486     girarX = 137.0;
487     girarY = -55.0;
488     break;
489
490 case 8:
491     /*if (glfwGetTime() > 25) {
492         if (cae) {
493             if (girarZ >= -141) {
494                 girarZ -= 6.0 * deltaTime;
495                 cae2 = true;
496                 if (cae2) {
497                     if (girarX >= -55) {
498                         girarX -= 2.0 * deltaTime;
499                     }
500                 }
501             }
502         }
503     }*/
504     girarZ = -141.0;
505     girarX = -25.0;
506     break;
507
508 case 9:
509     /*if (glfwGetTime() > 25) {
510         if (cae) {
511             if (girarZ >= -135) {
512                 girarZ -= 6.0 * deltaTime;
513                 cae2 = true;
514                 if (cae2) {
515                     if (girarX >= -20) {
516                         girarX -= 2.0 * deltaTime;
517                     }
518                 }
519             }
520         }
521     }*/
522     girarX = -135.0;
523     girarZ = -20.0;

```

```

16         girarX += 2.0 * deltaTime;
17     }
18 }
19 }
20 }
21 */
22 girarX = -135.0;
23 girarZ = -20.0;
24 break;
25
26 case 10:
27     if (glfwGetTime() > 25) {
28         if (cae) {
29             if (girarZ <= 134) {
30                 girarZ += 6.0 * deltaTime;
31                 cae2 = true;
32                 if (cae2) {
33                     if (girarY <= 25) {
34                         girarY += 2.0 * deltaTime;
35                     }
36                 }
37             }
38         }
39     }
40     /*girarZ = 134.0;
41     girarY = 25.0;*/
42     break;
43 }
44

```

```

//para movimiento de la moto en el tablero empezando desde -x, -z
if (glfwGetTime() > 20) {
    if (avanzaMoto)
    {
        if (movimientoMoto < 140.0f)
        {
            movimientoMoto += movimientoMotoOffset * deltaTime;
            rotllantamoto += rotllantamotoOffset * deltaTime;
        }

        else
        {
            avanzaMoto = !avanzaMoto;
        }
    }

    else
    {
        avanzaMoto = !avanzaMoto;
    }
}

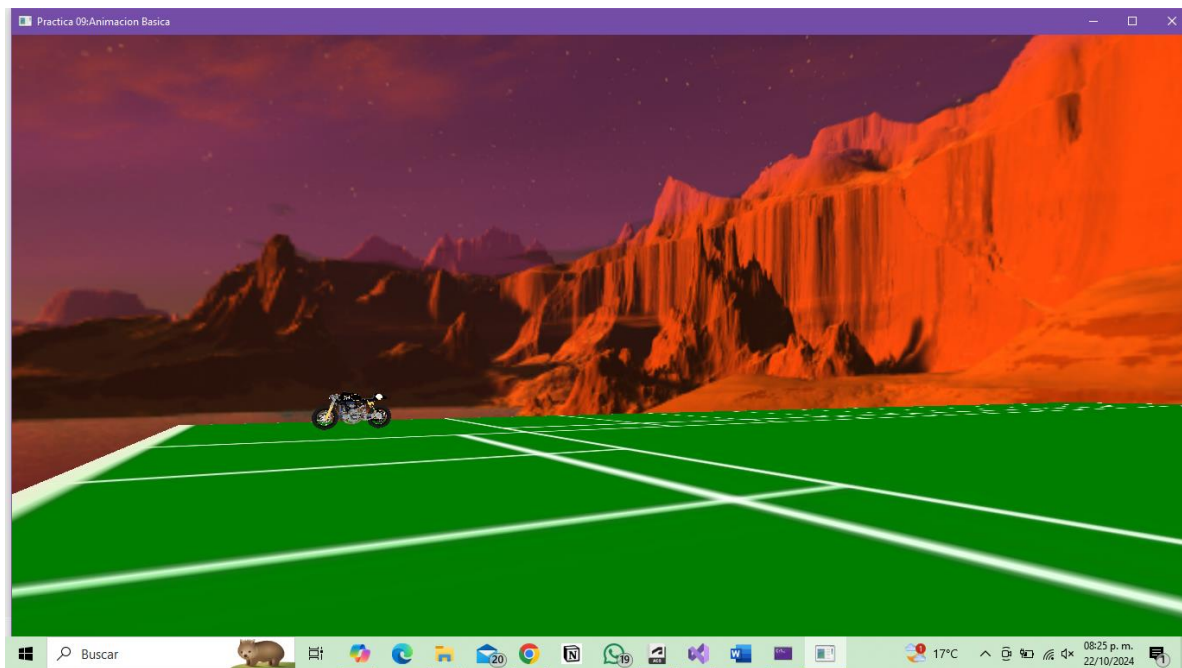
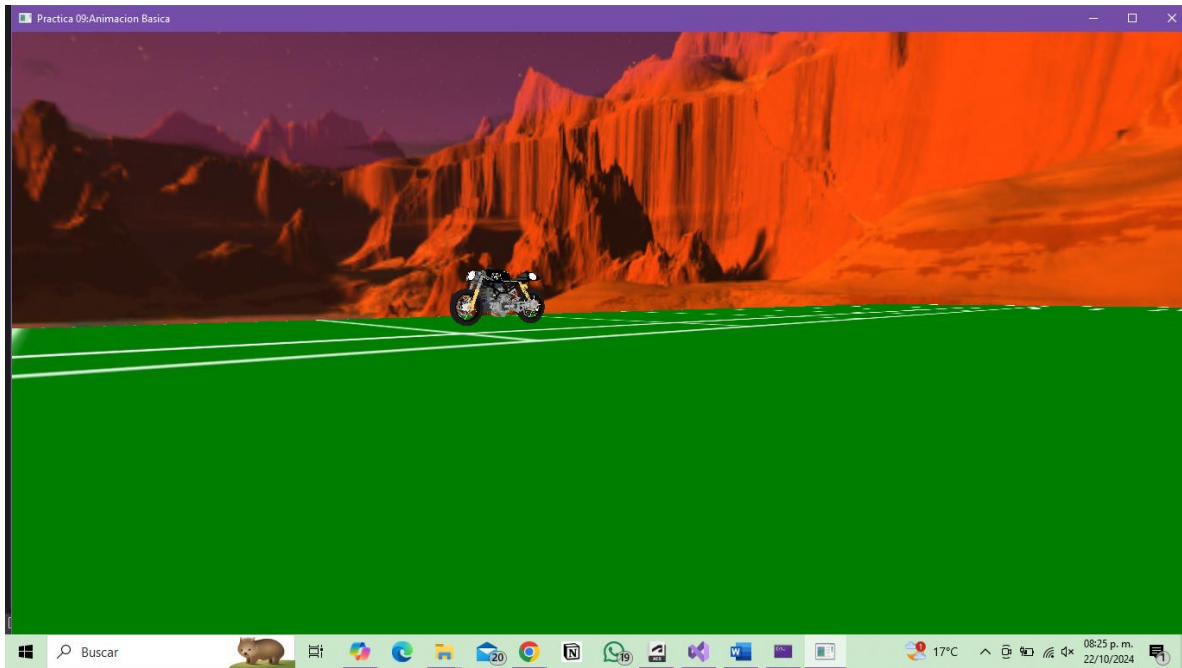
```

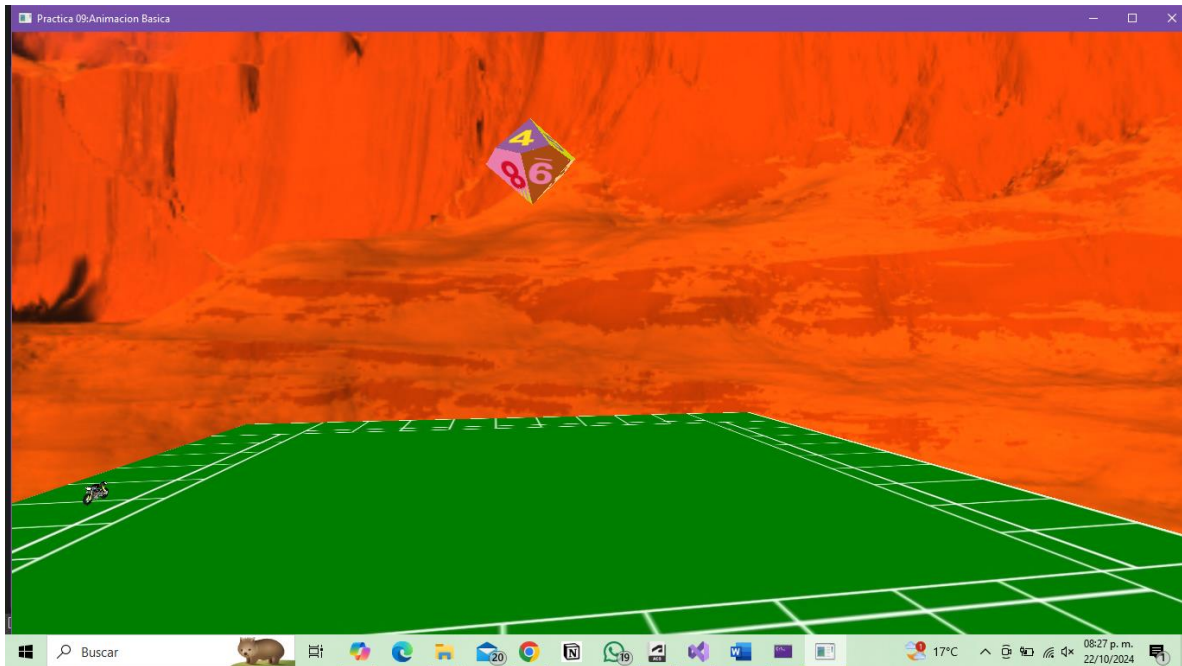
```

656
657 //Practica 9
658 //1. Su dado de 10 caras cae sobre el piso, gira y cae en un número "random", se repite la tirada al presionar una tecla
659 model = glm::mat4(1.0);
660 model = glm::translate(model, glm::vec3(0.0f, movimientoD + 60.5f, 0.0f));
661 model = glm::rotate(model, girarX * toRadians, glm::vec3(1.0f, 0.0f, 0.0f));
662 model = glm::rotate(model, girarY * toRadians, glm::vec3(0.0f, 1.0f, 0.0f));
663 model = glm::rotate(model, girarZ * toRadians, glm::vec3(0.0f, 0.0f, 1.0f));
664 glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
665
666 dado10_M.RenderModel();
667
668
669
670 //2. Por integrante del equipo elegirán un tipo de vehículo: terrestre o aéreo . Cada integrante del equipo creará un
671 recorrido en el cual el vehículo se moverá alrededor de su tablero de Monopoly. Cada vehículo iniciará su recorrido a
672 partir de una esquina diferente. (el vehículo terrestre no puede ser un carro o vehículo similar
673 motorizado de 4 ruedas, se debe de tener movimiento de llantas o de hélices en sus vehículos.)
674 */
675
676 //tablero monopoly
677 model = glm::mat4(1.0);
678 model = glm::translate(model, glm::vec3(0.0f, 0.0f, 0.0f));
679 modelaux = model;
680 model = glm::scale(model, glm::vec3(5.5f, 5.5f, 5.5f));
681 glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
682 mapa_M.RenderModel();
683
684 //moto
685 model = glm::mat4(1.0);
686 model = glm::translate(model, glm::vec3(movimientoMoto2 - 70.0f, 1.5f, movimientoMoto - 70.0f));
687 modelaux = model;
688 model = glm::scale(model, glm::vec3(0.9f, 0.9f, 0.9f));
689
690
691 //moto
692 model = glm::mat4(1.0);
693 model = glm::translate(model, glm::vec3(movimientoMoto2 - 70.0f, 1.5f, movimientoMoto - 70.0f));
694 modelaux = model;
695 model = glm::scale(model, glm::vec3(0.9f, 0.9f, 0.9f));
696 glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
697 cuerpomoto_M.RenderModel();
698
699 //llanta moto delante
700 model = modelaux;
701 model = glm::translate(model, glm::vec3(0.0f, 0.0f, 2.0f));
702 model = glm::rotate(model, rotllantamoto * toRadians, glm::vec3(1.0f, 0.0f, 0.0f));
703 model = glm::scale(model, glm::vec3(0.9f, 0.9f, 0.9f));
704 //color = glm::vec3(0.5f, 0.5f, 0.5f); //llanta con color gris
705 glUniform3fv(uniformColor, 1, glm::value_ptr(color));
706 glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
707 llantamoto_M.RenderModel();
708
709 //llanta moto delante
710 model = modelaux;
711 model = glm::translate(model, glm::vec3(0.0f, 0.0f, -1.9f));
712 model = glm::rotate(model, rotllantamoto * toRadians, glm::vec3(1.0f, 0.0f, 0.0f));
713 model = glm::scale(model, glm::vec3(0.9f, 0.9f, 0.9f));
714 //color = glm::vec3(0.5f, 0.5f, 0.5f); //llanta con color gris
715 glUniform3fv(uniformColor, 1, glm::value_ptr(color));
716 glUniformMatrix4fv(uniformModel, 1, GL_FALSE, glm::value_ptr(model));
717 llantamoto_M.RenderModel();
718
719 //blending: transparencia o traslucidez
720 glEnable(GL_BLEND);

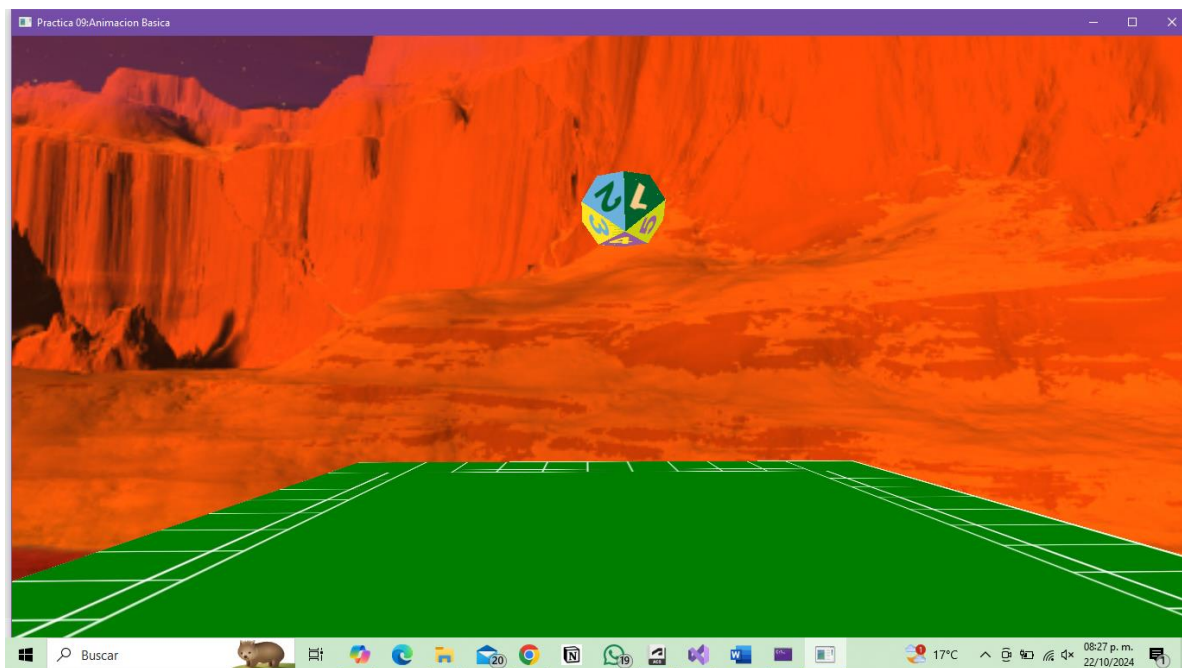
```

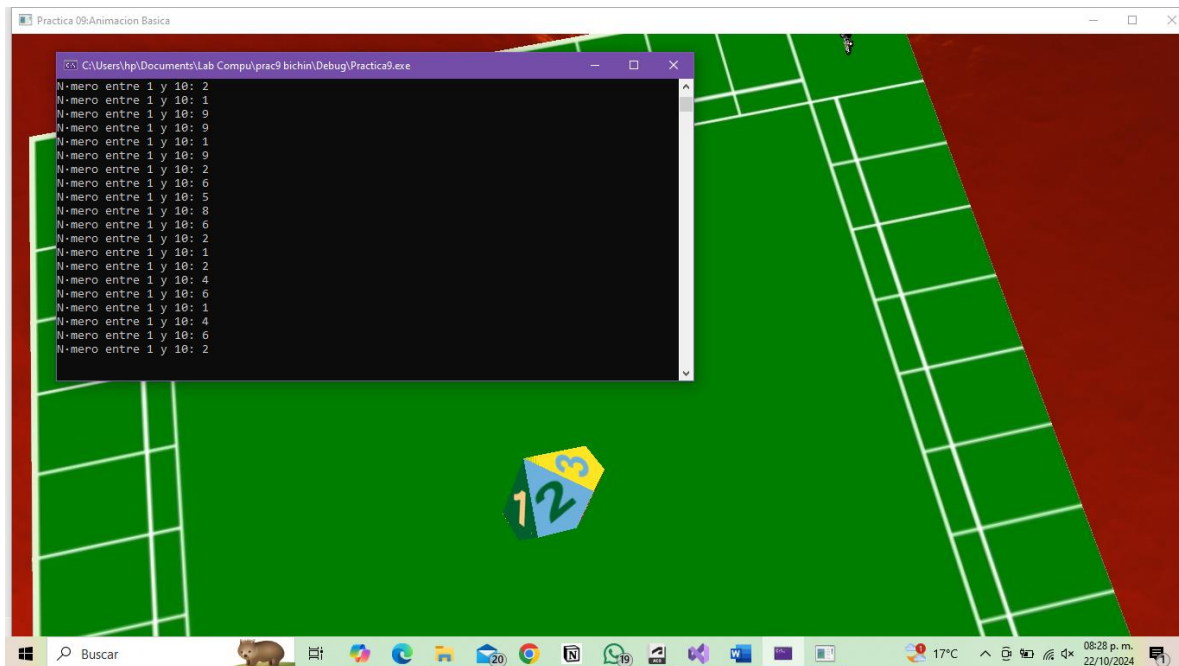
Ejecución del programa





Mediante la letra R se generan los números aleatorios y el dado gira





2.- Liste los problemas que tuvo a la hora de hacer estos ejercicios y si los resolvió explicar cómo fue, en caso de error adjuntar captura de pantalla.

- Uno de los problemas que tuve al inicio fue con el dado de 10 caras ya que el que había realizado no estaba bien hecho para que en este ejercicio al momento de caer solo se viera una cara, por lo que tuve que descargar un modelo y agregarle una textura que elegí para que se viera en el dado.
- El segundo problema que tuve fue al momento de crear los casos para cada una de las caras del dado, ya que no sabía bien como delimitarlas por lo que estuve jugando con los valores hasta que logré encontrar cada uno correctamente.
- El tercer problema que tuve fue con el modelo del vehículo que elegí, ya que no se exportaba de manera correcta la textura y tenía que separar cada una de sus partes para poder tener por separado las llantas del modelo y así poderle dar movimiento.
- El cuarto problema que tuve fue al momento de hacer el movimiento de las llantas de la moto, ya que, si giraban, pero no de manera correcta, por lo que tuve que volver a abrirlas en 3DMAX y me di cuenta que no se había exportado bien con el pivote en donde iba, así que volví a modificar el pivote y volverlo a abrir en OpenGL y ahora si ya se movían de la manera correcta.
- El quinto problema que tuve fue al momento de hacer las líneas para el movimiento de la moto dentro del tablero de monopoly, ya que no supe como

hacer para que la moto siguiera todo el recorrido de cada una de las esquinas y tuve que agregarle un valor un poco alto en la variable del tiempo ya que si lo dejaba mas pequeño la moto se desplazaba de lugar.

- El sexto problema que presente fue con el dado, ya que al inicio no se genera el numero aleatorio y cae el dado de manera vertical, pero cuando se presiona la letra R ahora si ya genera el numero aleatorio correspondiente a la cara que mostrara y gira, aquí también le tuve que agregar un valor un poco alto en la variable del tiempo cuando ejecuta el código ya que si no no se visualizaba la primera vez que caía el dado.

3.- Conclusión:

- a. Los ejercicios del reporte: Complejidad, Explicación.

Después de haber realizado los ejercicios propuestos para este practica puedo concluir que no logre realizarlos todos con éxito, ya que aunque comprendí la manera de darle animación a los objetos mediante la respectiva implementación del código para el caso de estos ejercicios no supe muy bien como realizarlos, ya que me confunde un poco como manejar los valores, lo que hicimos durante estos ejercicios fue primero para el dado de 10 caras descargar un modelo y darle su correspondiente textura para después cargarlo en OpenGL y con esto realizar los casos para cada una de las caras del dado y la función de random que nos ayudaba con el número que debía salir al tirar el dado, así como también la implementación de una tecla para que esta acción se volviera a repetir. Para el caso del segundo ejercicio se buscó un modelo de una moto y se separaron el cuerpo y una de sus llantas para poder darles posteriormente el movimiento, se agrego el tablero de monopoly usado para nuestro proyecto y en una esquina diferente cada uno de los integrantes de nuestro equipo situó su modelo.

La complejidad de los ejercicios para el movimiento del dado y de la moto puedo decir que si fue mas complicado que cosas realizadas anteriormente ya que no me quedo bien claro y no supe bien cómo hacerlo mediante la forma correcta y como saber cada valor.

- b. Comentarios generales: Faltó explicar a detalle, ir más lento en alguna explicación, otros comentarios y sugerencias para mejorar desarrollo de la práctica.

Por mi parte puedo concluir que en el caso de esta practica me quedaron algunas dudas, quizá faltó comentar un poco mas acerca de como realizar cada ejercicio correctamente o como se podía implementar en el código.

c. Conclusión

Después de haber concluido con la practica puedo mencionar que se cumplió no al 100% los objetivos propuestos, debido a que, aunque comprendí el uso correcto de la animación básica en los modelos me faltó haber realizado de manera correcta los ejercicios, aun así mediante las actividades propuestas pude poner en practica los conocimientos que he ido aprendiendo durante las sesiones de laboratorio y así cumplir con la mayoría.