ΠΡΟΓΡΑΜΜΑΤΙΣΤΙΚΗ ΑΣΚΗΣΗ 1-A

**Γραφικά Υπολογιστών και Συστήματα Αλληλεπίδρασης - ΜΥΥ702**

**Ακαδ. Έτος 2021-2022**

Κικιδης κωνσταντινοσ (4387)

Τσαμπίρασ Κωνσταντινοσ (4508)

Λειτουργία του προγράμματος

Το πρόγραμμα είναι σε μεγάλο ποσοστό βασισμένο στο αρχείο που μας δόθηκε στο 1ο εργαστήριο της OpenGL, όταν τρέξουμε το πρόγραμμα, ανοίγει ένα νέο παράθυρο διαστάσεων 800 x 800 με τίτλο «2Δ Σχήματα», όπου απεικονίζεται το σχήμα 1(α).

Background pattern

Description automatically generatedBackground pattern

Description automatically generated

Figure 2: Το σχήμα που φτιάξαμε εμείς

Figure 1: Το σχήμα της εκφώνησης

Όταν πατάμε το πλήκτρο “c” τότε, απεικονίζεται το σχήμα 1(β)

Background pattern

Description automatically generatedBackground pattern

Description automatically generated

Figure 3: Το σχήμα που θα πρέπει να έχουμε πατώντας C

Figure 4: Το σχήμα που έχουμε πατώντας C

Όταν αφήσουμε το “c” θα γυρίσουμε πάλι στο σχήμα 1(α).

Μόλις πατήσουμε το πλήκτρο “n” (ή το x του παραθύρου), η λειτουργία του προγράμματος, τερματίζεται.

Ιδιαιτερότητες/Ειδικές συνθήκες



Αρχικά όταν δημιουργήσαμε το παράθυρο, αλλάζοντας τον τίτλο του σε “2Δ Σχήματα”, μας εμφάνιζε τους ελληνικούς χαρακτήρες, έτσι.

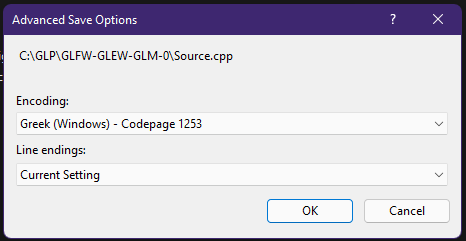
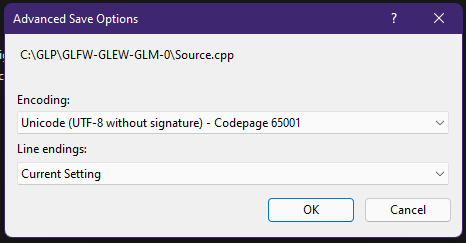
Η λύση σε αυτό το πρόβλημα ήταν η αποθήκευση του αρχείου με κωδικοποίηση Unicode (UTF-8)

Figure 6: Η λύση στο πρόβλημα

Figure 5: Αρχική ρύθμιση που προκαλούσε το πρόβλημα

Υπολογισμός Συντεταγμένων

Το κέντρο του παραθύρου, είναι η αρχή των αξόνων (0,0,0), τα υπόλοιπα σημεία είναι μαρκαρισμένα με x στις παρακάτω εικόνες. (οι άξονες που φαίνονται είναι ο x οριζόντια και ο y κατακόρυφα, ο z δεν φαίνεται εδώ)

Background pattern

Description automatically generated



Ανάλυση του κώδικα που χρησιμοποιήθηκε

Σαν βάση χρησιμοποιήσαμε τον κώδικα που μας δόθηκε στο 1ο εργαστήριο, εκεί έγιναν τροποποιήσεις στο test.cpp και στο SimpleFragmentShader.fragmentshader

Για το τελευταίο, η αλλαγή που έγινε αφορά το χρώμα όπου βάλαμε το ματζέντα αλλάζοντας την RGB τιμή σε 1,0,1

// Output color = Magenta

color = vec3(**1,0,1**);

Στο test.cpp, αλλάξαμε το όνομα σε Source.cpp και πρόσθέσαμε/αλλάξαμε τα παρακάτω

* Γραμμή 139, Διαστάσεις και τίτλος παραθύρου

window = glfwCreateWindow(**800, 800, "2Δ Σχήματα"**, NULL, NULL); ////

* Γραμμή 161, Μωβ χρώμα στο background, από το pdf της εκφώνησης πήραμε αυτές τις τιμές, R:142 G:13 B:211. Οι τιμές που βάλαμε είναι από την διαίρεση των παραπάνω τιμών με το 256 (142/256=0.55…, κ.ο.κ.)

glClearColor(**0.55f, 0.05f, 0.82f**, 0.0f); ////

* Γραμμές 181-205, 2 πίνακες που κρατάνε τις συντεταγμένες των κορυφών των τριγώνων για το 1ο και το 2ο σχήμα

static const GLfloat g\_vertex\_buffer\_data[] = {…};

static const GLfloat g\_vertex\_buffer\_data**\_2**[] = {…};

* Γραμμές 213-216, βάλαμε έναν vertexbuffer για το 2ο σχήμα, όπως υπάρχει για το 1ο στις γραμμές 208-211

GLuint **vertexbuffer2**;

glGenBuffers(1, &vertexbuffer**2**);

glBindBuffer(GL\_ARRAY\_BUFFER, vertexbuffer**2**);

glBufferData(GL\_ARRAY\_BUFFER, sizeof(g\_vertex\_buffer\_data\_**2**), g\_vertex\_buffer\_data\_**2**, GL\_STATIC\_DRAW);

* Γραμμή 239, Φτιάχνουμε 12 ακμές, για 4 τρίγωνα (3 ακμές για κάθε τρίγωνο)

glDrawArrays(GL\_TRIANGLES, 0, **4 \* 3**); // ..

* Γραμμή 247, αν και για όσο πατιέται το πλήκτρο “c”, θα εμφανίζεται το σχήμα 1(β), οι γραμμές 248-275 είναι ίδιες με τις γραμμές 220-245 εκτός των γραμμών 228 και 257 όπου στην πρώτη έχουμε χρήση του vertexbuffer (σχήμα 1(α)) και στη δεύτερη έχουμε τον vertexbuffer2 (σχήμα 1(β)).

while (glfwGetKey(window, **GLFW\_KEY\_C) == GLFW\_PRESS**) {…}

glBindBuffer(GL\_ARRAY\_BUFFER, **vertexbuffer**);

glBindBuffer(GL\_ARRAY\_BUFFER, **vertexbuffer2**);

* Γραμμή 278, για όσο το πλήκτρο “n” δεν πατιέται, το πρόγραμμα μας τρέχει, ενώ μόλις πατηθεί (ή αν πατηθεί το x του παραθύρου) θα τερματίσει

while (glfwGetKey(window, **GLFW\_KEY\_N) != GLFW\_PRESS** && ////