ΠΡΟΓΡΑΜΜΑΤΙΣΤΙΚΗ ΑΣΚΗΣΗ 1-Α (2023)

Γραφικά Υπολογιστών και Συστήματα Αλληλεπίδρασης

Στοιχεία ομάδας:

Όνομα: Δημοσθένης Ζάγκας ΑΜ: 4359

Όνομα: Ανδρέου Άγγελος ΑΜ: 4628

Ημερομηνία : 03/11/2023

0. ΕΙΣΑΓΩΓΗ

Στα πλαίσια του εργαστηρίου του μαθήματος “Γραφικά Υπολογιστών και Συστήματα Αλληλεπίδρασης” μας ζητήθηκε να κάνουμε μία εισαγωγική άσκηση OpenGL (1-Α) ώστε να εξοικειωθούμε με τα βασικά στοιχεία και τις δυνατότητες που προσφέρει. Στο πρόγραμμα δημιουργήσαμε ένα παράθυρο που εμφανίζεται ένα τετράγωνο να μεταφέρεται σε 5 θέσεις. Το βασικό αρχείο που δουλέψαμε το πρόγραμμα κάνοντας αλλαγές πάνω σε αυτό είναι το “Source-1A.cpp” που μας δόθηκε με το υλικό για την άσκηση.

1. Περιγραφή της εργασίας

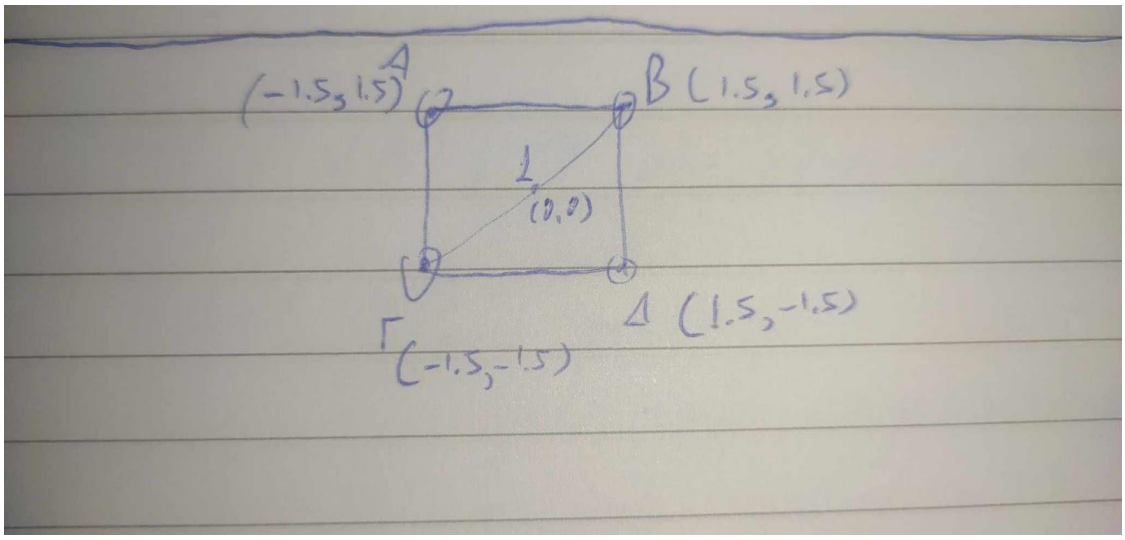
I. Αρχικά για να ανοίγει το παράθυρο σε 900x900 διαστάσεις και με το όνομα που ζητήθηκε αλλάξαμε τη συνάρτηση της glfw ως εξής:

```
window = glfwCreateWindow(900, 900, u8"Πρώτη άσκηση 2023", NULL, NULL);
```

II. Στο δεύτερο ζητούμενο καθορίζουμε τα vertices για τα τετράγωνα:

Αρχικά πρέπει να βρούμε τα 4 σημεία στα επίπεδα που εμφανίζεται το κάθε τετράγωνο με σειρά. Αφού έχουμε 2-Δ σχήμα η συνιστώσα στον άξονας z είναι πάντα 0 οπότε θα παραλείπεται στην εξήγηση. Αφού η πλευρά το τετραγώνου είναι 3 και το κέντρο του η αρχή των αξόνων έχουμε τα σημεία (εικόνα 1):

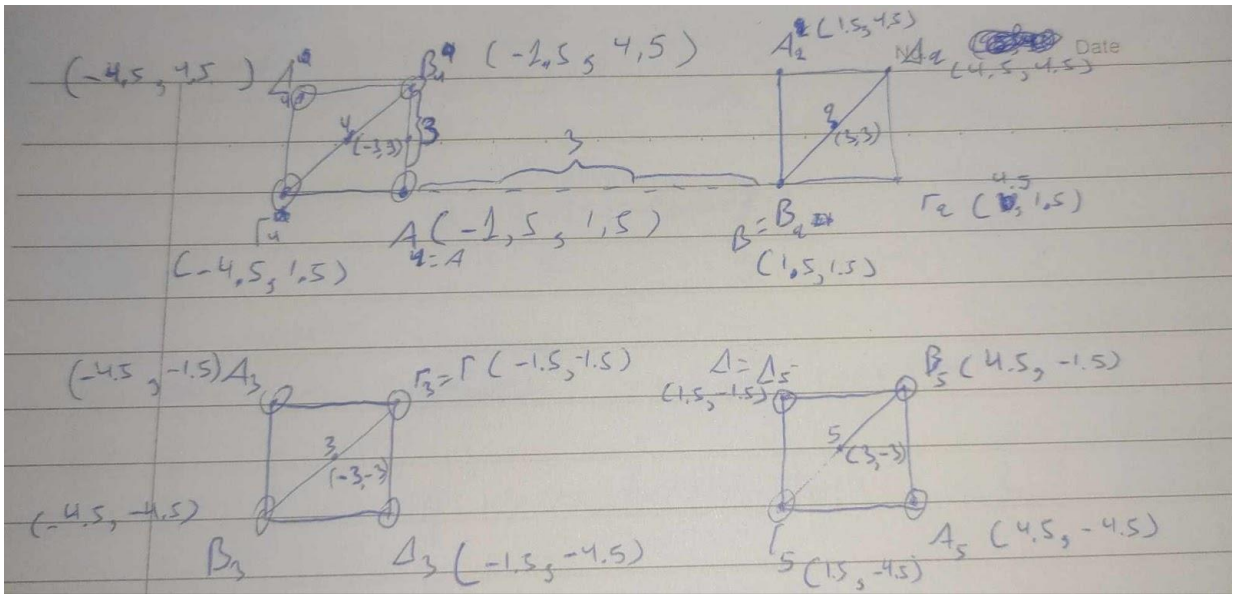
```
147
148     // Open a window and create its OpenGL context
149     window = glfwCreateWindow(900, 900, u8"Πρώτη άσκηση 2023", NULL, NULL);
150
151
```



εικόνα 1

Συμμετρικά θέλουμε 1.5 μονάδες απόσταση από το (0,0) ώστε η συνολική τους απόσταση να είναι 3.

Για τα υπόλοιπα τετράγωνα δουλεύουμε αντίστοιχα προσέχοντας το 1 κοινό σημείο που έχουν με το αρχικό τετράγωνο (εικόνα 2).



εικόνα 2

Στην OpenGL πρέπει να σχηματίσουμε τα τετράγωνα σαν 2 τρίγωνα με μία κοινή πλευρά και 2 κοινές κορυφές. Για παράδειγμα για το πρώτο τετράγωνο (εικόνα 1) σημειώνουμε τα 2*3=6 vertices ως εξής:

// First triangle

// Points: a=(1.5,1.5,0), b=(-1.5,-1.5,0), c=(1.5,-1.5,0)

// x-distance b to c is from -1.5 to 1.5 = 3, as per requirements

// y-distance a to b is from -1.5 to 1.5 = 3, as per requirements

1.5f,1.5f,0.0f,

1.5f,-1.5f,0.0f,

-1.5f,-1.5f,0.0f,

// Second triangle

1.5f,1.5f,0.0f,

-1.5f,1.5f,0.0f,

-1.5f,-1.5f,0.0f,

Για τα υπόλοιπα τετράγωνα:

// Second square

// First triangle

1.5f,1.5f,0.0f,

4.5f,1.5f,0.0f,

4.5f,4.5f,0.0f,

// Second triangle

1.5f,1.5f,0.0f,

1.5f,4.5f,0.0f,

4.5f,4.5f,0.0f,

// Third square

// First triangle

-1.5f,-1.5f,0.0f,

-1.5f,-4.5f,0.0f,

-4.5f,-4.5f,0.0f,

```
        // Second triangle
        -1.5f,-1.5f,0.0f,
        -4.5f,-1.5f,0.0f,
        -4.5f,-4.5f,0.0f,
    // Forth square
        // First triangle
        -1.5f,1.5f,0.0f,
        -1.5f,4.5f,0.0f,
        -4.5f,4.5f,0.0,
        // Second triangle
        -1.5f,1.5f,0.0f,
        -4.5f,1.5f,0.0f,
        -4.5f,4.5f,0.0f,
    // Fifth square
        // First triangle
        1.5f,-1.5f,0.0f,
        1.5f,-4.5f,0.0f,
        4.5f,-4.5f,0.0f,
        // Second triangle
        1.5f,-1.5f,0.0f,
        4.5f,-1.5f,0.0f,
        4.5f,-4.5f,0.0f
```

Τα οποία αποθηκεύονται στον πίνακα:

```
static const GLfloat shape_1_buffer[] {...}
```

```
207
208     static const GLfloat shape_1_buffer[] = {
209
210         // First square
211             // First triangle
212             // Points: a=(1.5,1.5,0), b=(-1.5,-1.5,0), c=(1.5,-1.5,0)
213             // x-distance b to c is from -1.5 to 1.5 = 3, as per requirements
214             // y-distance a to b is from -1.5 to 1.5 = 3, as per requirements
215             1.5f,1.5f,0.0f,
216             1.5f,-1.5f,0.0f,
217             -1.5f,-1.5f,0.0f,
218             // Second triangle
219             // Same as first triangle
220             1.5f,1.5f,0.0f,
221             -1.5f,1.5f,0.0f,
222             -1.5f,-1.5f,0.0f,
223             // Second square
224                 // First triangle
225                 1.5f,1.5f,0.0f,
226                 4.5f,1.5f,0.0f,
227                 4.5f,4.5f,0.0f,
228                 // Second triangle
229                 1.5f,1.5f,0.0f,
230                 1.5f,4.5f,0.0f,
231                 4.5f,4.5f,0.0f,
232                 // Third square
```

```
231                 4.5f,4.5f,0.0f,
232                 // Third square
233                     // First triangle
234                     -1.5f,-1.5f,0.0f,
235                     -1.5f,-4.5f,0.0f,
236                     -4.5f,-4.5f,0.0f,
237                     // Second triangle
238                     -1.5f,-1.5f,0.0f,
239                     -4.5f,-1.5f,0.0f,
240                     -4.5f,-4.5f,0.0f,
241                     // Forth square
242                         // First triangle
243                         -1.5f,1.5f,0.0f,
244                         -1.5f,4.5f,0.0f,
245                         -4.5f,4.5f,0.0,
246                         // Second triangle
247                         -1.5f,1.5f,0.0f,
248                         -4.5f,1.5f,0.0f,
249                         -4.5f,4.5f,0.0f,
250                         // Fifth square
251                             // First triangle
252                             1.5f,-1.5f,0.0f,
253                             1.5f,-4.5f,0.0f,
254                             4.5f,-4.5f,0.0f,
255                             // Second triangle
256                             1.5f,-1.5f,0.0f,
257                             4.5f,-1.5f,0.0f,
258                             4.5f,-4.5f,0.0f
259
260     };
```

III. Για το iii) ερώτημα θα γίνεται εμφάνιση μέσα σε επανάληψη do while όπως στο πρότυπο και το test που είδαμε στο εργαστήριο, με τις διαφοροποιήσεις:

Χρήση των παραμέτρων:

```
int square_number = 0;
```

//για να ξέρουμε το τετράγωνο που πρέπει να εμφανιστεί σε κάθε φάση, αυξάνεται μετά από την εμφάνιση του μεσαίου, αρχικού τετραγώνου

```
bool flag = true;
```

//flag για έλεγχο αν εμφανίζουμε το μεσαίο τετράγωνο sq1->sq2->!sq1!->sq3 έτσι δεν χρειάζεται να επαναλάβουμε vertices στον πίνακα buffer

```
double wait_time = 1;//secs
```

//χρόνος αναμονής που θα γίνεται εμφάνιση κάθε τετραγώνου πριν πάμε στο επόμενο

```
double time = 0;//secs
```

//πόσο χρόνο έχουμε στην εμφάνιση αυτού του τετραγώνου

Για να υπολογίσουμε τον χρόνο που εμφανίζεται το ίδιο τετράγωνο κάναμε χρήση των time συναρτήσεων της glfw:

```
//get the time for transition
```

https://www.glfw.org/docs/3.3/group__input.html#gaa6cf4e7a77158a3b8fd00328b1720a4a

```
glfwSetTime(0);
time = glfwGetTime();//secs
//wait time for transition
```

Έτσι γίνεται συνεχόμμενα εμφάνιση του ενός τετραγώνου μέχρι να φτάσει το wait_time ή τερματίσουμε το πρόγραμμα

```
//check time for wait
```

```
            time = glfwGetTime() - time;

        } while (wait_time > time && glfwGetKey(window, GLFW_KEY_C) !=
GLFW_PRESS && glfwWindowShouldClose(window) == 0);
```

Για να εμφανίσουμε τα τρίγωνα μέσα στη do-while έχουμε

```
// Draw the triangle !
            if (flag) //we are at 1st squad
                glDrawArrays(GL_TRIANGLES, 0, 6); // draw square 1
            else
                glDrawArrays(GL_TRIANGLES, square_number, 6);
// 6 indices starting at square_number -> 2 triangles
```

Μετά αφού εμφανιστεί το τετράγωνο για το χρόνο που ορίσαμε έξω από το πρώτο do-while αλλάζουμε το square_number ώστε να πάμε στο επόμενο (το αρχικό αν ήμασταν σε κάποιο από τα εξωτερικά τετράγωνα ή το επόμενο μετά το κεντρικό σύμφωνα με τη σειρά που μας ζητήθηκε)

```
//if done 1st square go next
            flag = !flag;
        // Increase the square number after every draw to render the next set of
vertices that create the next square
        if (square_number <= 24 && !flag) {
            square_number += 6;
        }
        // When the last square is drawn, start over
        if ( square_number > 24) {
            square_number = 0;
            flag = true;
        }
```

Έτσι θα έχουμε διαδοχικά τις εκτελέσεις:

```
        glDrawArrays(GL_TRIANGLES, 0, 6);
```

glDrawArrays(GL_TRIANGLES, 6, 6);

glDrawArrays(GL_TRIANGLES, 0, 6);

glDrawArrays(GL_TRIANGLES, 12, 6);

glDrawArrays(GL_TRIANGLES, 0, 6);

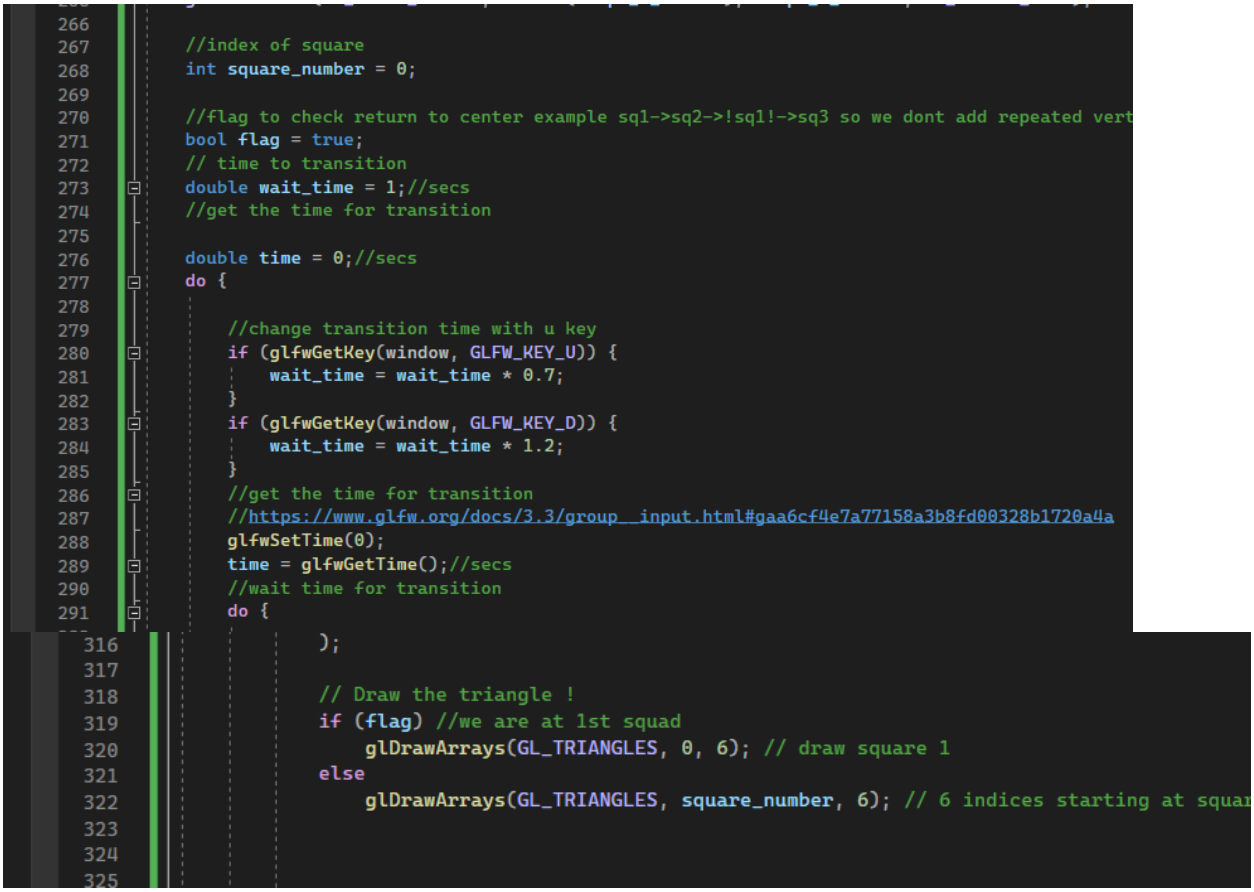
glDrawArrays(GL_TRIANGLES, 18, 6);

glDrawArrays(GL_TRIANGLES, 0, 6);

glDrawArrays(GL_TRIANGLES, 24, 6);

Μέχρι να πατηθεί το κουμπί c ή κλείσουμε το παράθυρο:

while (glfwGetKey(window, GLFW_KEY_C) != GLFW_PRESS &&
glfwWindowShouldClose(window) == 0);



```
333         } while (wait_time > time && glfwGetKey(window, GLFW_KEY_C) != GLFW_PRESS && g
334
335         //if done 1st square go next
336         flag = !flag;
337         // Increase the square number after every draw to render the next set of verti
338         if (square_number <= 24 && !flag) {
339             square_number += 6;
340         }
341         // When the last square is drawn, start over
342         if (square_number > 24) {
343             square_number = 0;
344             flag = true;
345         }
346         /*
347         glDrawArrays(GL_TRIANGLES, 6, 6);
```

IV. Στο 4ο ερώτημα για αλλαγή του χρόνου μετάβασης με τα κουμπιά u , d:

Ελέγχουμε μέσα στο πρώτο do-while αν πατηθεί κάποιο από τα δύο κουμπιά. Αν πατηθεί το “u” θέλουμε να αυξάνεται ο ρυθμός έτσι μειώνουμε το χρόνο που εμφανίζεται το τετράγωνο (wait_time) κατά 0.7.

Αν πατηθεί το “d” θέλουμε να μειώνεται ο ρυθμός έτσι αυξάνουμε το χρόνο που εμφανίζεται το τετράγωνο (wait_time) κατά 1.2.

```
//change transition time with u key
        if (glfwGetKey(window, GLFW_KEY_U)) {
            wait_time = wait_time * 0.7;
        }
        if (glfwGetKey(window, GLFW_KEY_D)) {
            wait_time = wait_time * 1.2;
        }
```

- Όλα τα ερωτήματα υλοποιήθηκαν.

2. Πληροφορίες σχετικά με την υλοποίηση:

- Λειτουργικό σύστημα: Windows 64-bit
- Περιβάλλον: Visual Studio versions 17.7.6
- Παρότι το λειτουργικό σύστημα είναι 64-bit μπορεί χωρίς πρόβλημα να τρέξει εφαρμογές 32-bit
- Η συγγραφή αυτού του report έγινε σε google docs.

3. Σύντομη αξιολόγηση της λειτουργίας της ομάδας:

Η ομάδα λειτούργησε χωρίς προβλήματα κυρίως εξ αποστάσεως με χρήση του gitHub για συντονισμό αρχείων. Μαζί κάναμε τον σχεδιασμό που απαιτήθηκε, με τον Δημοσθένη να κάνει την εγγραφη του κώδικα και το πέρασμα των vertices των τετραγώνων ενώ ο Άγγελος ασχολήθηκε με τη σωστή λειτουργία και τον ρυθμό εναλλαγής του μοτίβου και με τη συγγραφή αυτής της αναφοράς.

4. Αναφορές:

- https://www.glfw.org/docs/3.3/group__input.html#gaa6cf4e7a77158a3b8fd00328b1720a4a
- https://www.glfw.org/docs/3.3/quick_guide.html
- https://www.glfw.org/docs/3.3/window_guide.html
- https://www.opengl-tutorial.org/
- https://www.wikihow.com/Set-Up-OpenGL-GLFW-GLEW-GLM-on-a-Project-with-Visual-Studio