Report

Question-1:

C.2.a:

In this methodology, once the primitive is placed at its correct position, we can't pick it again, so, once it's placed we need to render it at the same angle, size and position even if others are changed, so we need to maintain a set which has transformation matrices of all primitives. Once we put it at the correct position, we'll remove it from that set and we write a condition where if the transformation matrix of a primitive is there in the set, then only we can pick it else we can't

C.2.b:

In this methodology, you can always pick a primitive, so until the system changes from mode-1 to mode-2, we can always pick the primitives. You need to constantly update the transformation matrices of the primitives in this as they may change.

Question-2:

API critical for implementing "picking" using mouse click is "addEventListner" API

Question-3:

In mode-0 not many key clicks happen so no need to reduce it. In mode-1 we use key clicks to navigate among the primitives, but we can use 'w', 'a', 's', 'd' to navigate among them and use arrow keys to move the selected primitive.

Question-4:

Centroid is important for primitives because we draw all the vertices of primitives based on our centroid. So when we translate a primitive then we translate the centroid and compute a new centroid. Then we draw our primitive. For rotation, our primitive rotates about the centroid of the primitive. That's why Centroid is important.

Implementation-Strategy:

For Implementation, I have created 2 canvas and 2 java script files to render it. The first file consists the vertices needed to render the configuration shown on the left side (C0). It is not changed once its rendered.

For the configuration on the right side (C1) I created a new java script file. Different classes are created for different primitives. There is one shader file which handles the vertex and fragment shaders. The code for shaders is in vertex.js and fragment.js files. The primitives are rendered on the canvas with random centroids and are rotated at random angles.

In mode – 0:

The left side canvas shows the final configuration to be achieved. The right side has the primitives at random positions

In mode-1:

Now we can select an element using mouse click and move that object using arrow keys, rotate clock wise using ‘(‘and anti-clock wise using ‘)’. We can decrease the size of the primitive with “- “and increase the size with “+”.

In mode-2:

Now in mode 2 we cannot select the individual primitives, we can only move the configuration achieved using arrow keys, rotate clock wise using ‘(‘and anti-clock wise using ‘)’. We can decrease the size of the primitive with “- “and increase the size with “+”.

In mode-3:

The primitives on the right side are deleted and a blank screen is displayed.

For showing two configurations on the left side I store the vertices of the primitives in a separate array and render them whenever necessary.

References:

1. https://webglfundamentals.org/
2. https://glmatrix.net/docs/index.html
3. https://developer.mozilla.org/en-US/docs/Web/API/EventListener