Assignment 3: Pink Torus World

We will use Three.js to create a landscape of pink toruses.

In class, we learned how to create a `THREE.Mesh` by combining the `THREE.BoxGeometry` and the `THREE.MeshStandardMaterial`. We also learned how to *unproject* a mouse click from 2D (viewport / screen space) to a 3D position. This way, we were able use the `window.onclick` callback to move a cube to a new position in the 3D scene. Now, we will extend our code.

The goal of this assignment is to create many different pink torus geometries in the scene. This means, rather than moving an existing mesh, we will create new ones in the `renderer.domelement.onmousedown` callback. On each click, our code places a new torus geometry at the current mouse position. And, the torus should be **hotpink** - at least for now!

Now, we want to use mouse drag to scale the torus after placement. For this, we will need to deactivate the OrbitControls and listen to mouse movements to scale the torus until the mouse button is released and OrbitControls are activated again. But let's do this step-by-step:

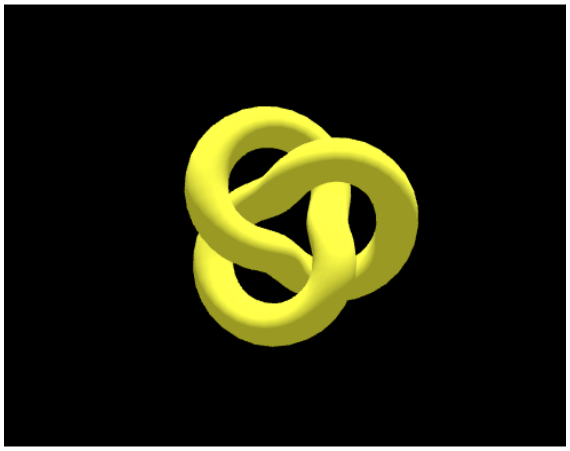
Part 1 (10 points): The torus geometry.

Before we start coding, we want to understand the parameters of a `THREE.TorusKnotGeometry`. You can find this information in the Three.js documentation at `https://threejs.org/docs/` (scroll down to Geometries). Keep notes - you will need this information for the form submission.

Part 2 (10 points): Replace the cube with a torus.

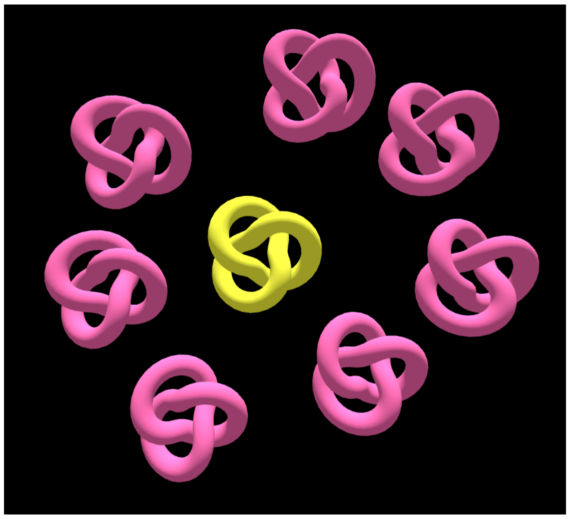
Please write code to place a new torus on each click at the current mouse position. We will use the `SHIFT`-key to distinguish between geometry placement and regular camera movement. Copy the starter code from `https://cs460.org/shortcuts/08/` and overwrite **03/index.html** in your github fork. This code includes the `renderer.domElement.onclick` callback and the `unproject` functionality.

Now replace the cube from class with a solid torus geometry in a color of your choice.



Part 3 (20 points): Switch the interaction and place some hotpink toruses.

We will need to change the `renderer.domElement.onclick` callback to `renderer.domElement.onmousedown` and check whether the `SHIFT`-key is pressed. Only then, we should turn off the `OrbitControls` using the `enabled` property and place a new hotpink torus for every mouse click (look up hotpink in hex).



Part 4 (10 points): Re-enable the controls.

Since we turned of `OrbitControls` in the `onmousedown` callback, we want to re-activate them in the `onmouseup` callback - again using the `enabled` property.

Part 5 (35 points): Now the tricky part.

With disabled `OrbitControls` we can catch the `renderer.domelement.onmousemove` event and interact with the last placed torus. In fact, we want to scale it. The torus should get larger or smaller when the mouse is dragged up or down (along the Y-axis).

Hint: Here we need to have access to the last placed torus, a global `LASTOBJECT` variable pointing to it makes sense :).

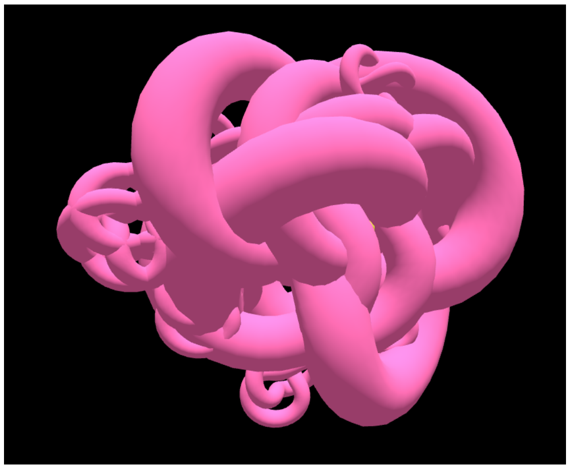
And note, that a torus can be scaled using the following code:

```
torus.scale.set(torus.scale.x + DELTA,
                torus.scale.y + DELTA,
                torus.scale.z + DELTA);
```

Luckily, in latest JavaScript, we can grab the `movementY` property from the `mousemove`-event. Check with some console.log how this property reacts when dragging the mouse.

Hint: Oh, and we don't want to always scale the last placed torus. We only want to do so when it has just been placed and the mousebutton was not released yet. **Sooo, we need a flag for that.**

With all these changes, we can now place a torus when holding SHIFT and the left mouse button, then drag up and down to make it smaller/larger until mouse button release. We can create scenes like this:



Part 6 (10 points): Be creative :)

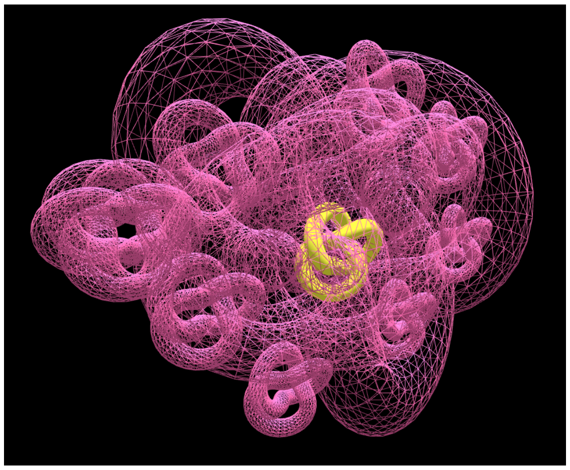
Now please create a beautiful hotpink torus blob and take a screenshot! Alright, if you want to use other colors - it's fine too :)

Part 7 (5 points): Submit your assignment.

Push everything to your fork and make sure that everything looks good on Github Pages!

As always, please send a pull request and then fill in the form at `https://cs460.org/assignments/03/` including your screenshot and all the urls and other stuff!

Bonus (33 points): Wireframes!



For the bonus, please visualize all placed toruses as a wireframe when `w` is pressed. Based on this view and your calculations, please estimate the number of vertices, edges, and faces for your scene! And, take a screenshot!