Ray Tracing Project　　Deniz Cangı 25427

Task 1: In the whole project instead of Color, Point and Vec classes from the book, I have used glm's dvec3. For the first task, I have created Ray, Camera, hittable, hittable_list and Sphere class which I have implemented through the book. Then into the world I have added 5 spheres which are colorful and a big sphere at the bottom which acts as a stage.

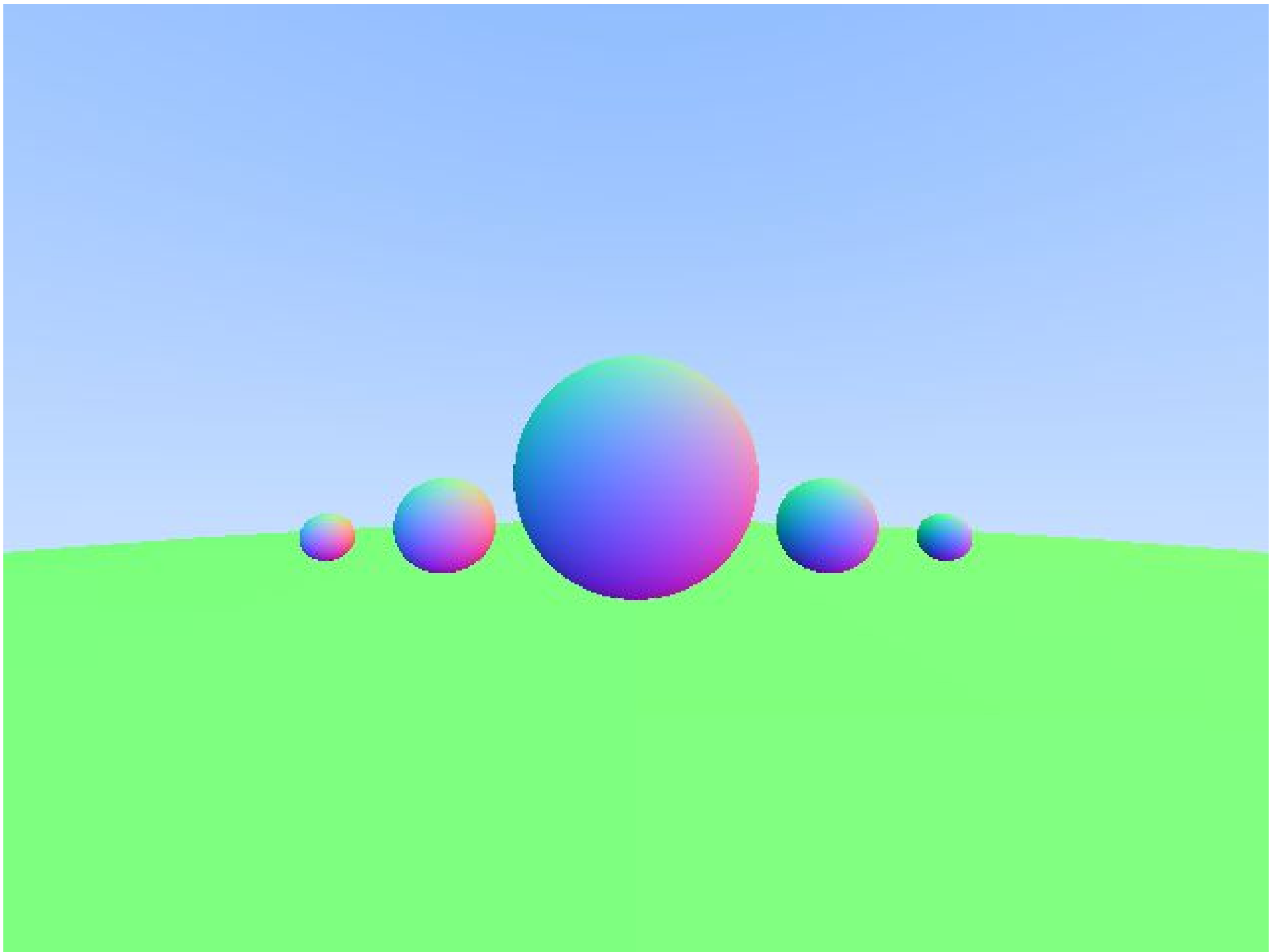
Position (p) and radius (r) of the spheres:

Big Sphere at the bottom: p = (0, -100.5, -1), r = 100

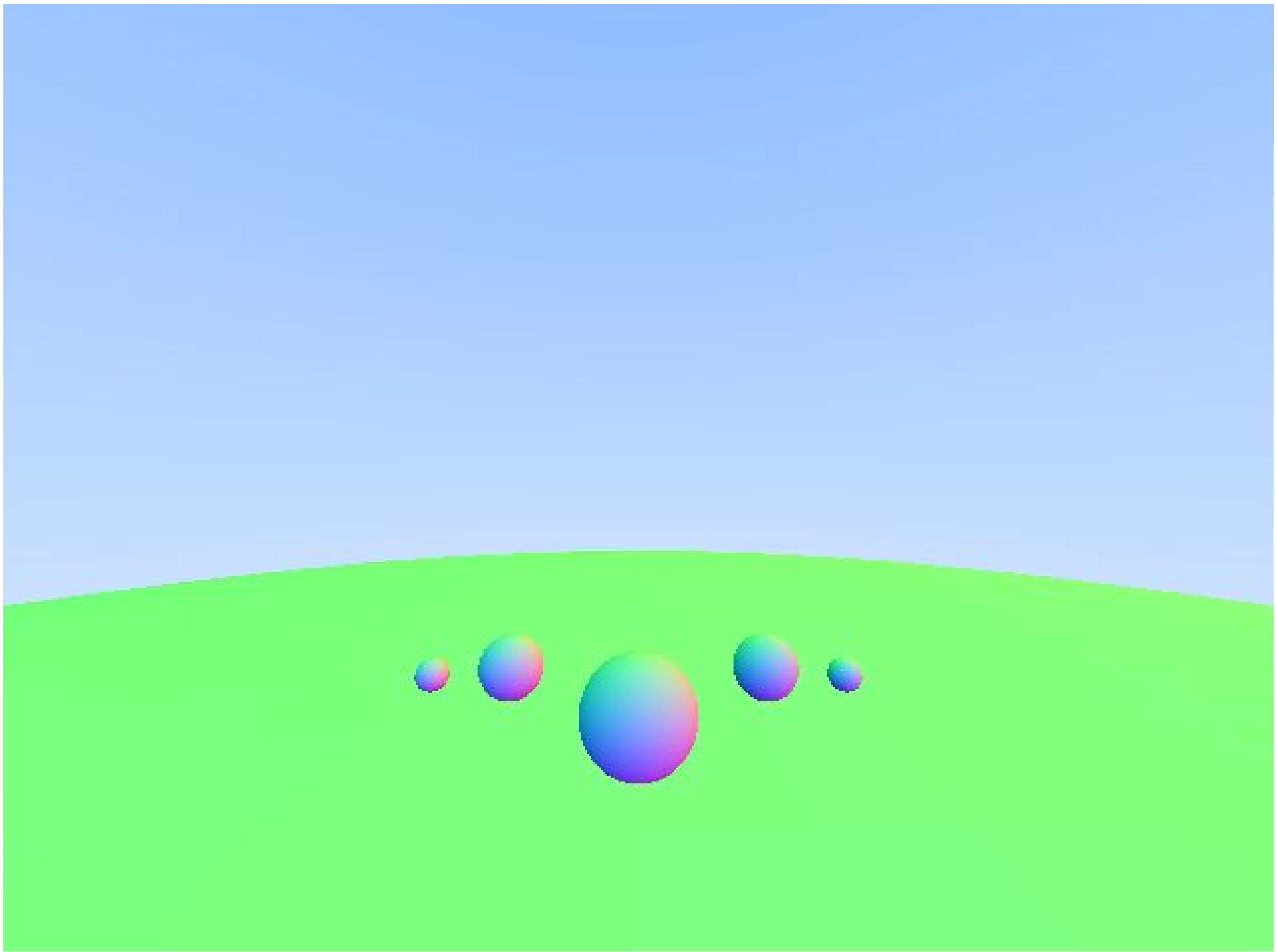
Other Spheres:

1. p = (0,0,-1), r = 0.25
2. p = (0.8,-0.2,-2), r = 0.2
3. p = (-0.8,-0.2,-2), r = 0.2
4. p = (1.3,-0.25,-2), r = 0.1
5. p = (-1.3,-0.25,-2), r = 0.1

For the first render camera position is origin = (0,0,0) and render is:



For the second render camera position is origin = (0, 1, 1) and render is:



Task 2: For anti aliasing I have used the random algorithm which I have benefited from the Ray Tracing in One Weekend. It simply sends multiple rays to each pixel and multiple number of sending rays is samples_per_pixel. For the task 2 part of the project I have used 500 samples per pixel but for the rest I usually used 100-200 samples per pixel. As it can be seen from the comparison, the image after anti aliasing has smoother edges.

Image part before anti-aliasing:

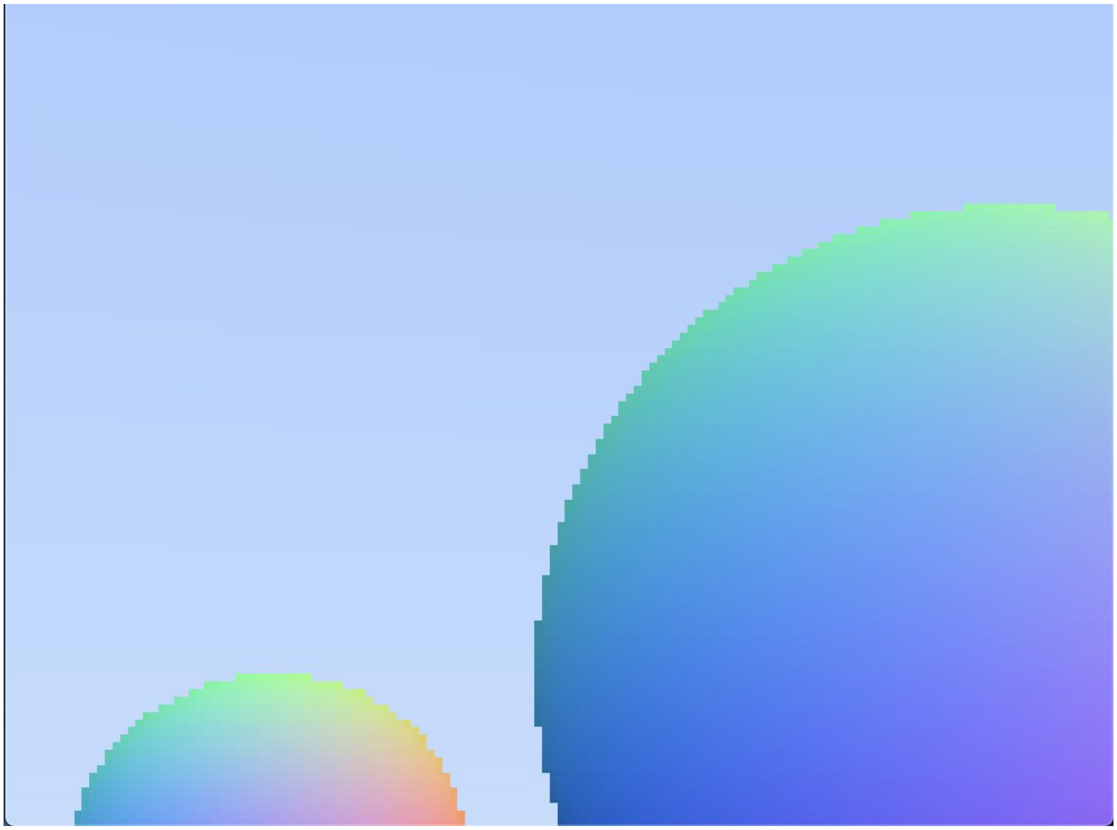
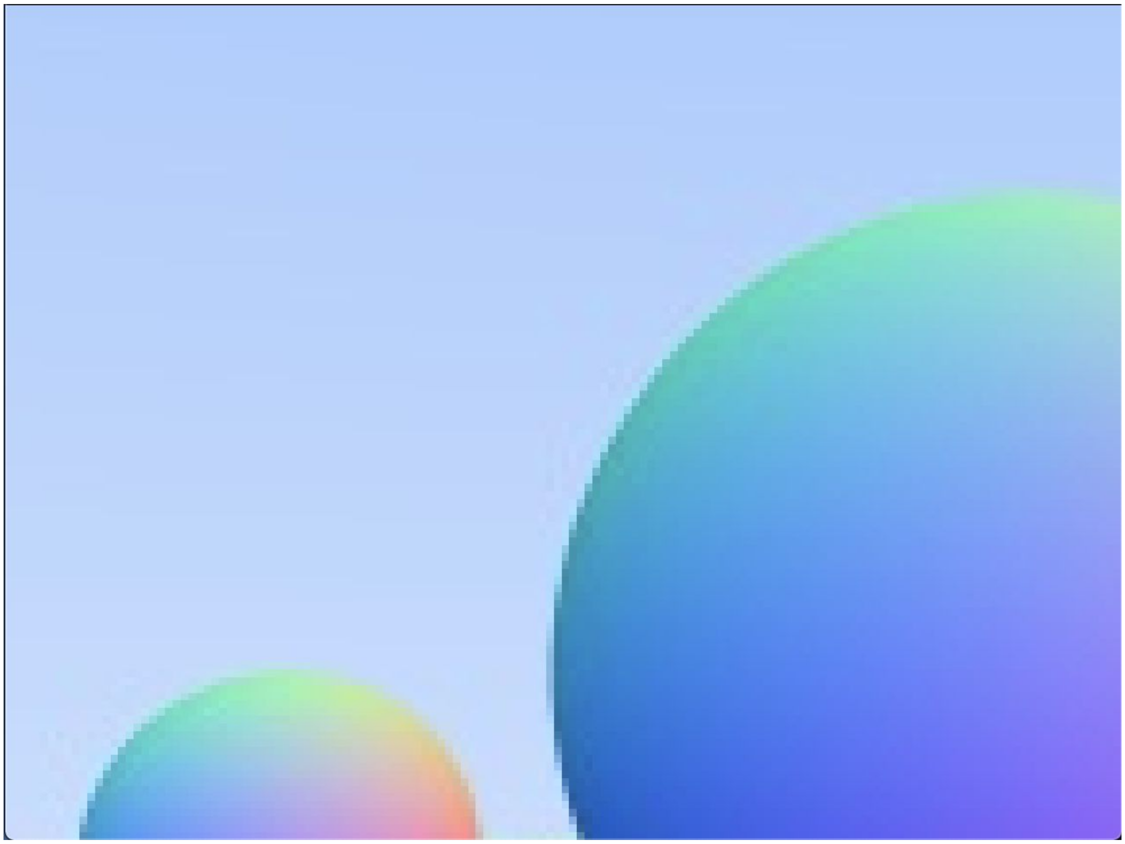


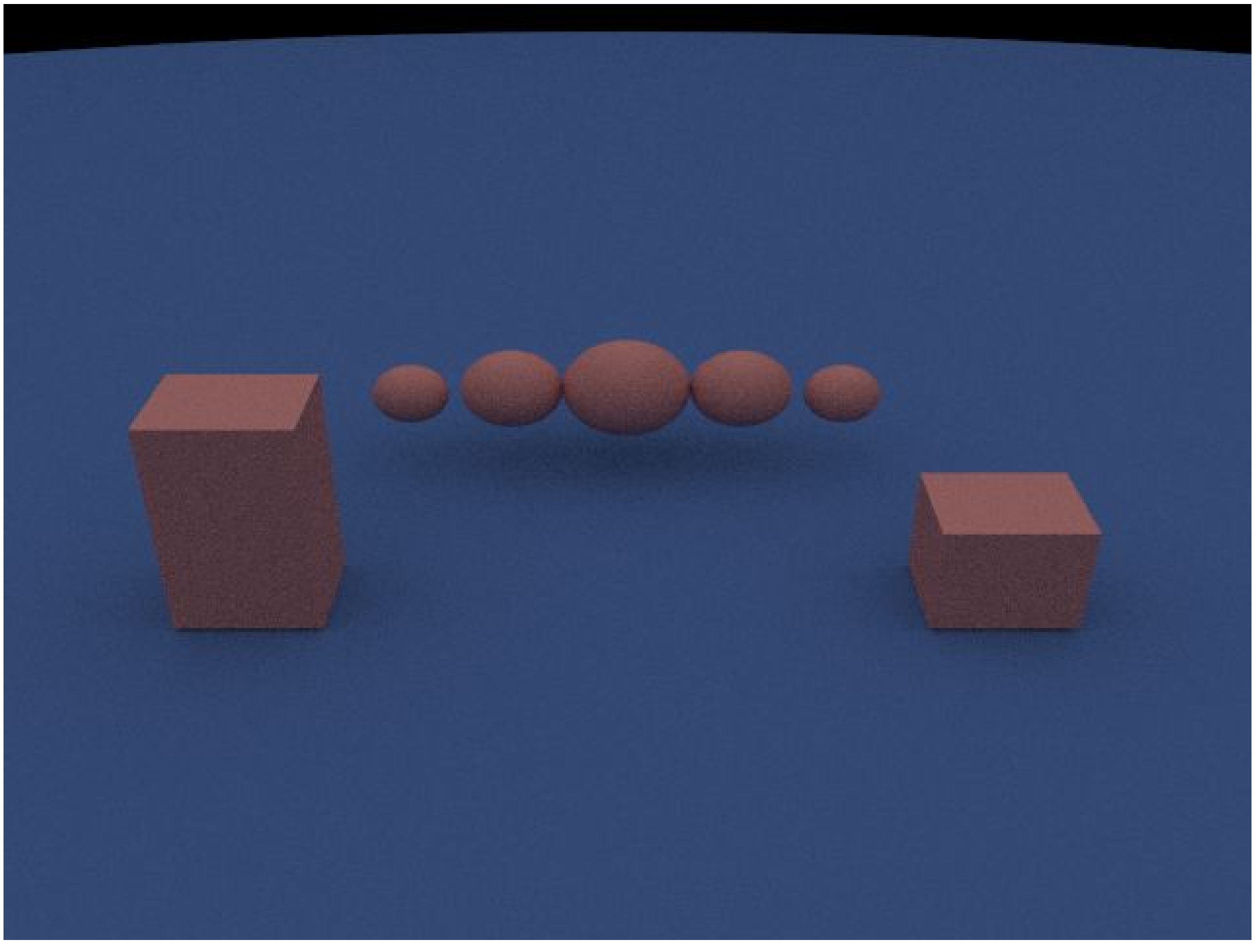
Image part after anti-aliasing:



Task 3: I have added a cube and rectangular prism to the spheres, changed the camera angle and position to see the cube and rectangular prism better.

I have found the formula from the ‘Ray Tracing: The Next Week’, from the box class.

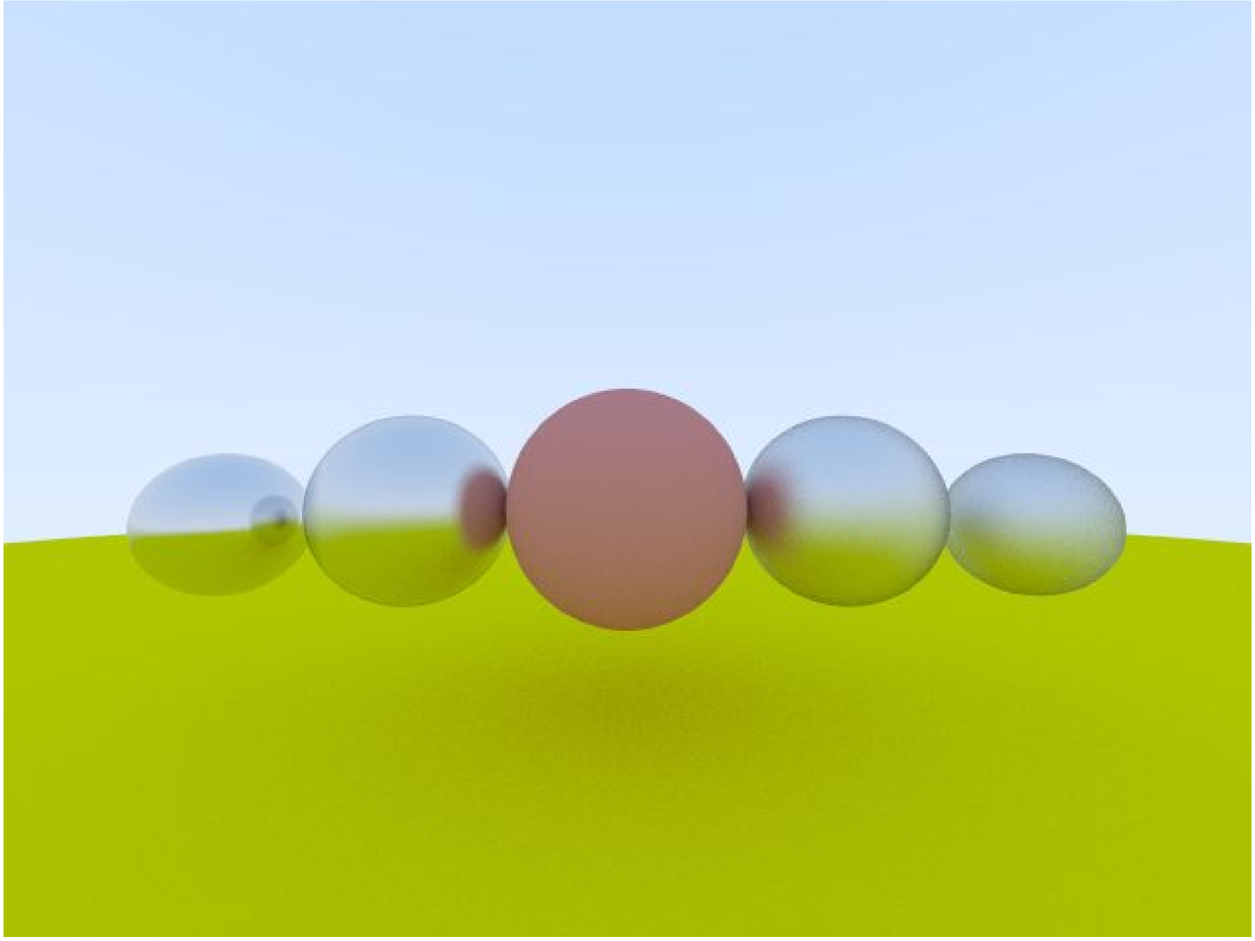
Render:



Task 4: I have implemented Lambertian Reflectance model and implemented the metal surfaces. In the render there are 6 spheres, 2 of them are implemented using the diffuse materials and 4 of them implemented using metal materials.

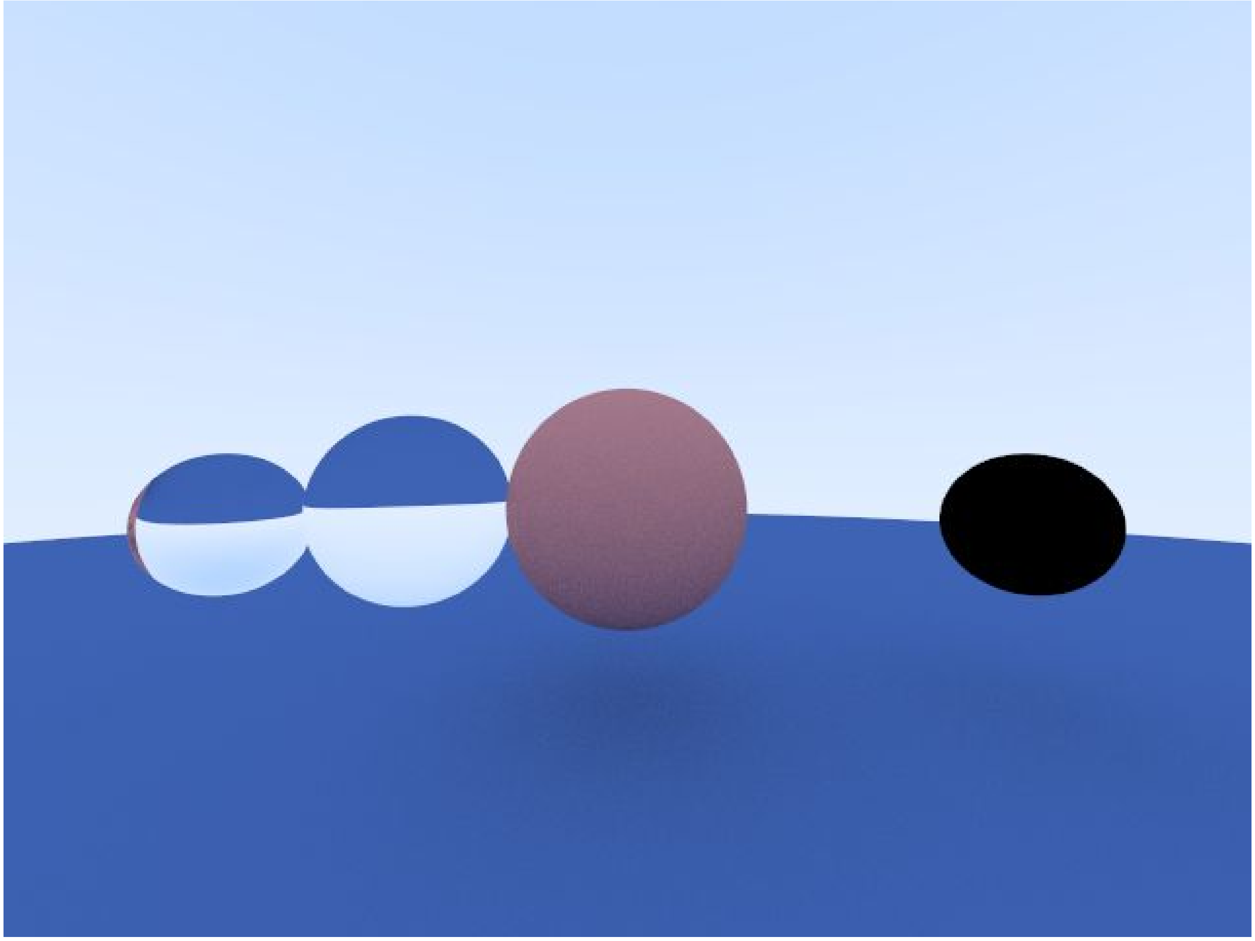
The big sphere at the bottom and the center sphere are implemented using diffuse materials, the leftmost sphere implemented using metal materials and fuzz is 0.1, the second leftmost sphere implemented using metal materials and fuzz is 0.3, the rightmost sphere implemented using metal materials and fuzz is 1 and the second rightmost sphere implemented using metal materials and fuzz is 0.7. All of the spheres in the right and left have the same color to see the effect of the fuzz parameter.

Render:



Task 5: For this task I have created 6 spheres, one is at the bottom with diffuse surface. There are 5 different spheres on top of it. The one in center has diffuse surface. The others are glass. The leftmost one has index of refraction 15, so it refracts the ray more. The second leftmost one has refraction index 1.5. The rightmost sphere has refraction index 0, so it won’t refract any ray and it’s black. The second sphere has refraction index 1, which means it allows rays to pass away it, so it looks like a space here.

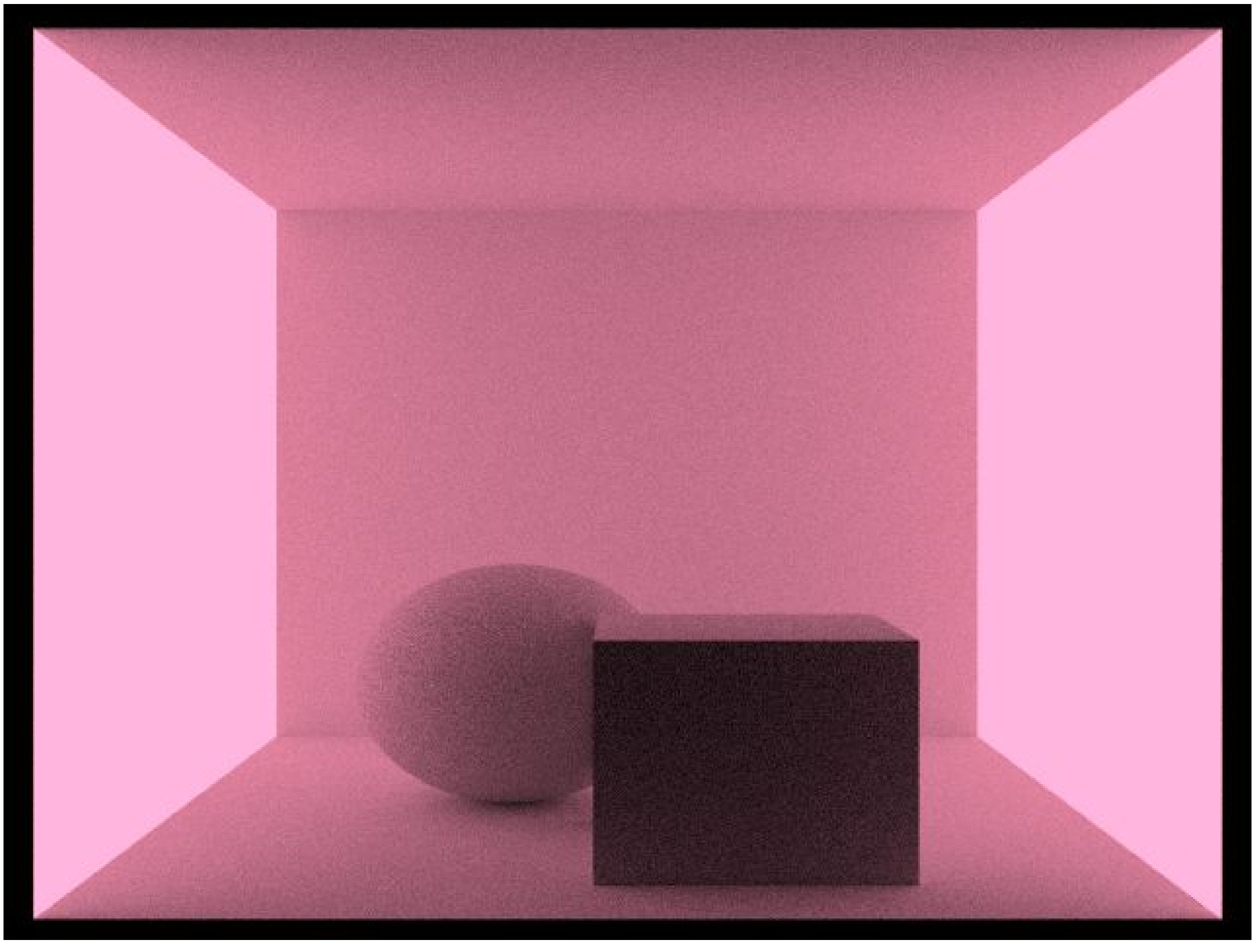
Render:



Task 6: For both of the renders I have used Cornell Box because it has diffuse surfaces.
The first one has no background light and the light has a color different from white which is pink. In the Cornell box I have a cube and a sphere which have surfaces diffusive, left and right sides of the Cornell box are all light. All the other surfaces of the box is diffusive and have color white so we can see the pink color clearly.

Left and right sides of the Cornell box are light with value (1.5, 0.5, 0.75). Other sides of the box, cube and sphere have surface implemented with Lambertian Reflectance which is implemented with noise implemented with value 4.

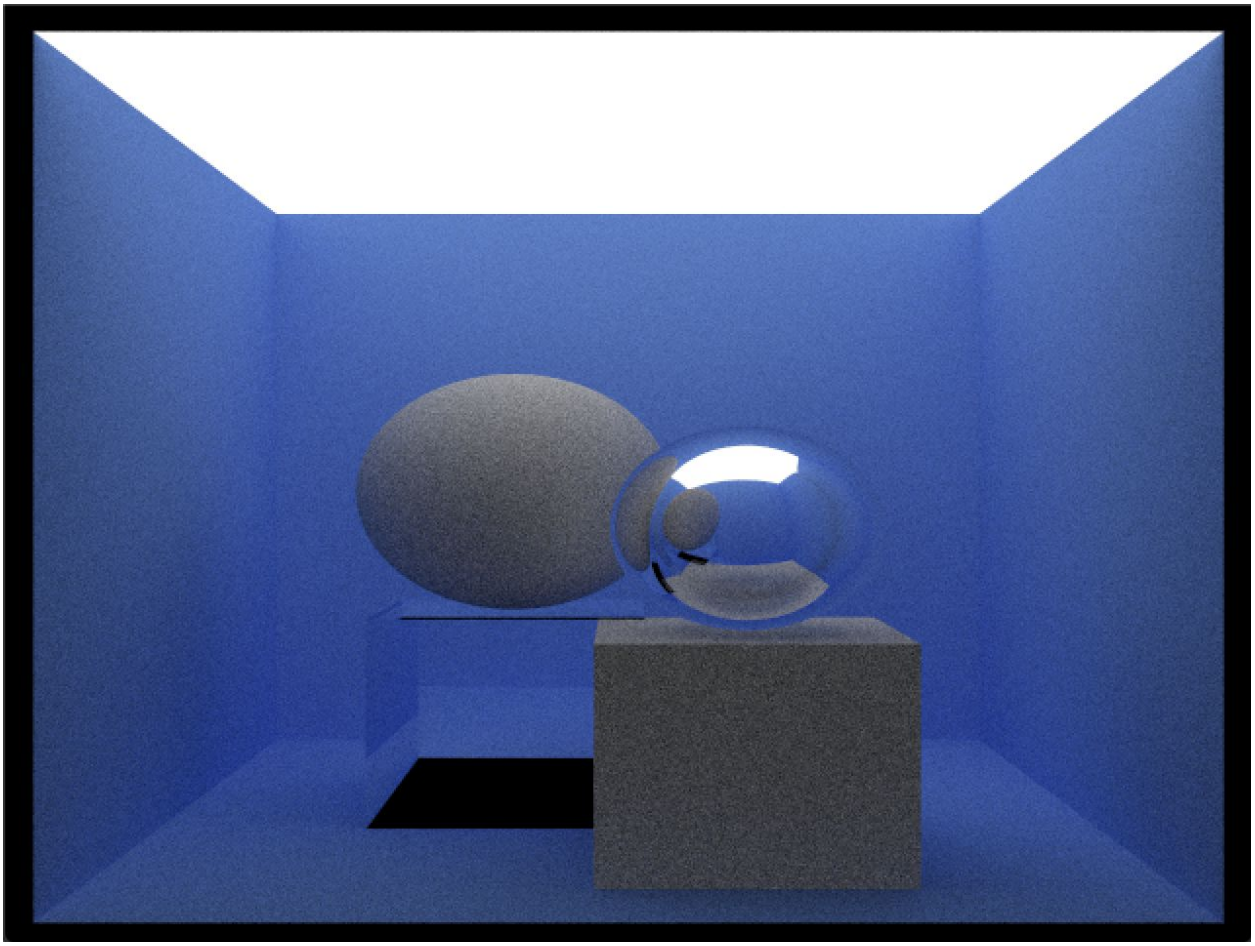
Render:



The second one has no background color too, this time light is the upper surface of the Cornell box, the second light is behind the camera and both lights have white color this time. All the other sides of the box have color blue that I have generated, I have 2 boxes and 2 spheres. One of the boxes and one of the spheres has an diffusive surface and they have white color. The other box and sphere have glass surface, the sphere has a hollow glass surface which I made by converting the radius into minus version.

Upper side of the box is light with value (1,1,1), one light is coming right after the camera which made the render more brighter than one light. Other sides of the box and one cube and one sphere have diffusive surface which is implemented with Lambertian Reflectance which is implemented with noise texture value 4. The other box and sphere have glass texture which is implemented with dielectric class with value 1.5, then I have made the radius of the sphere ‘-’ so that it looks nicer and it has hollow glass.

Render:



Task 7: For the creativity task, I have implemented a Cornell box like the one above but this time the opposite wall of the box has 14 spheres which creates my initials D and C and Create DC. I use DC like my signature and the closest people in my life call me DC.

To create DC, I had to make some calculations to make them fit and then implemented spheres with glass. Again the light is white and coming from the upside of the box and behind the camera.

Render:

Task 8:

- The ray tracing algorithms depend on the number of pixels that we are using to render the image and the number of objects that we have created in the scene. So the asymptotic time complexity depends on the number of pixels on screen and number of objects in the scene. If the number of pixels on screen is ‘p’ and number of objects in scene is ‘b’ the asymptotic time complexity for ray tracing is O(bp).
- Preprocessing is like first step before the other algorithms, it prepares the images for further algorithms. It ensures that the image does not have any disorder after the preprocessing. Computing an image is like creating an image with the given data from

the beginning. So one works with an already existing image the other is creating an image.

- Critical parameters are image width and image height which we call the pixels. The performance also depends on the number of objects in the scene since we find the intersection with ray.
- We have 6000x6000 pixels which make in total: 36,000,000, average scene has 500 objects so we have 1.8 * 10^10. In the slide set of ray tracing, we have resolution 512x512 and have 2 objects in the image which make 524,288 that takes 6 seconds to render. Then it takes 205,993 seconds which is equal to 57.2 days