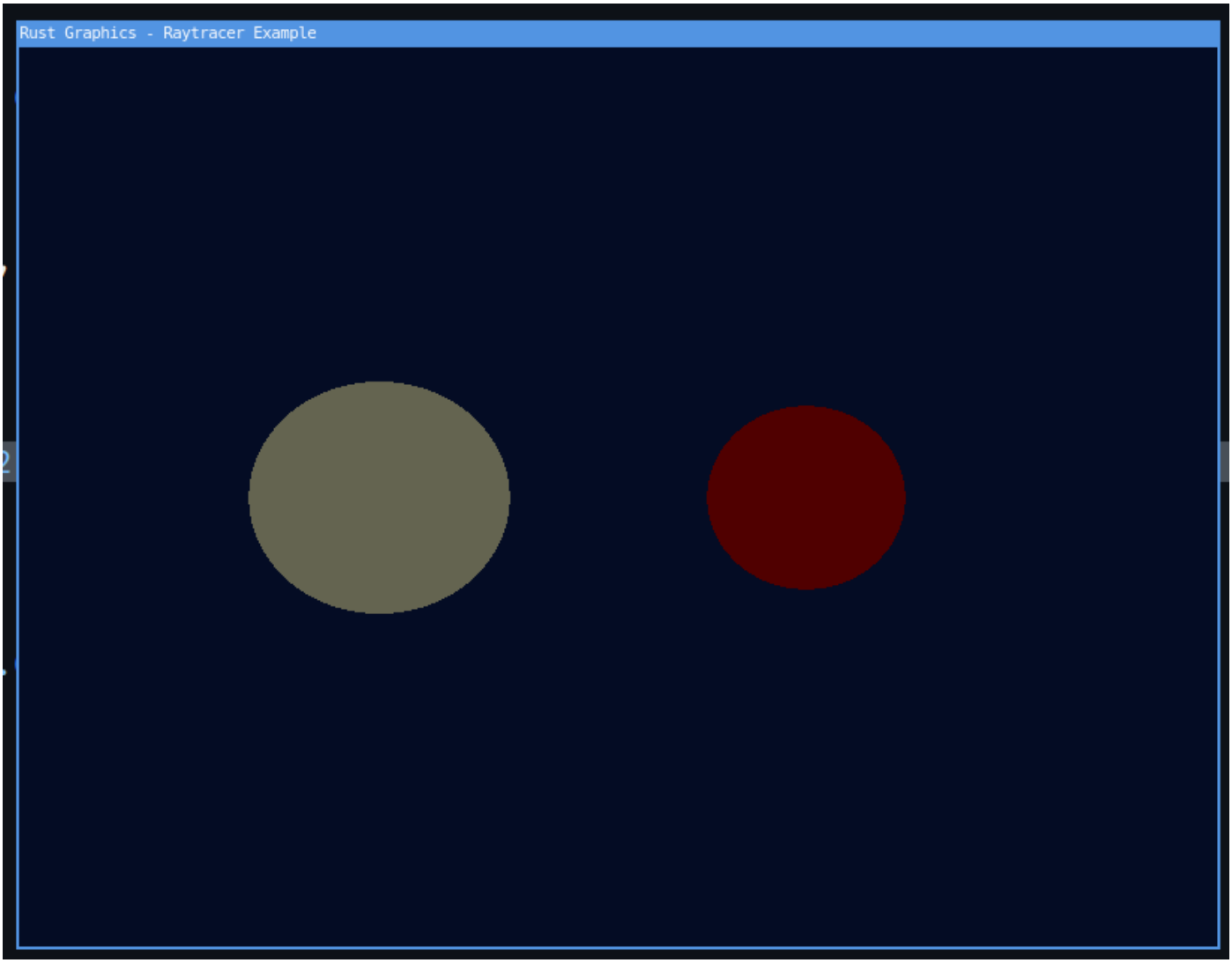
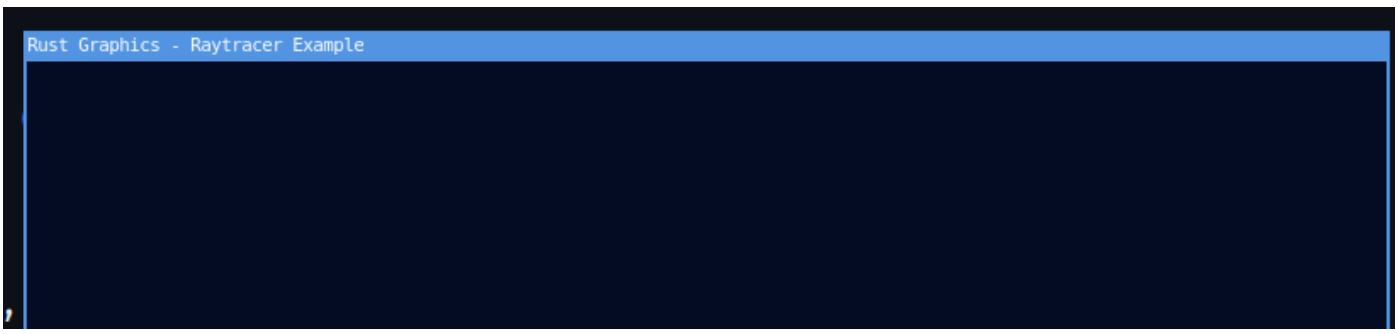
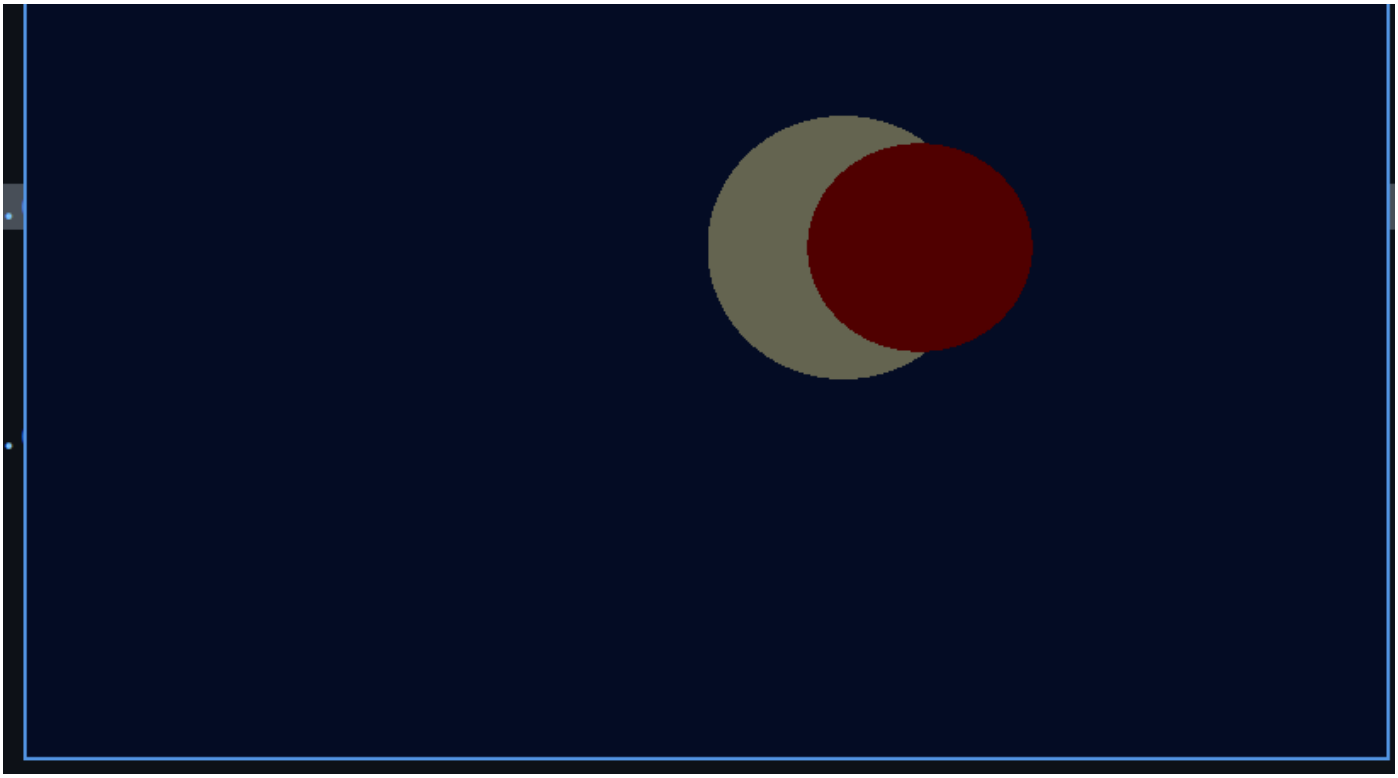
zbuffer

Nuestro raytracer tiene un problema en este momento, no estamos considerando la posición en z de nuestros objetos. Si tratamos de pintar dos objetos por separado, se ve bien:



Estas esferas son del mismo tamaño, pero la esfera blanca está más cerca que la roja, por lo que se ve más grande. Sin embargo, qué pasa si las acercamos más?

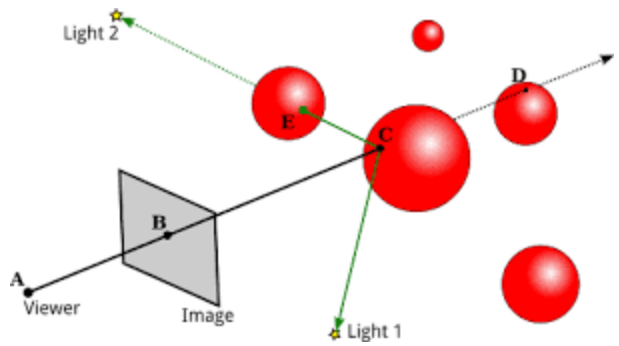




Como pueden ver, se está dibujando la esfera que está más cerca detrás de la que está más lejos, esto no es correcto. Esto ocurre porque mi raytracer está pintando las esferas en el orden en que las agregué y no según su posición en el espacio.

```
let objects = [
    Sphere {
        center: Vec3::new(1.0, 0.0, -4.0),
        radius: 1.0,
        material: ivory,
    },
    Sphere {
        center: Vec3::new(2.0, 0.0, -5.0),
        radius: 1.0,
        material: rubber
    },
];
```

El problema es que el rayo está tocando muchos objetos en el camino. Debemos elegir el objeto que el rayo tocó que está más cerca a la cámara. La estrategia más común para resolver esto es la de agregar una variable que pueda guardar la distancia a la que el rayo chocó con los objetos. Es por esto que guardamos la distancia en el Intersect:



En este ejemplo, el rayo impacta en C primero y luego en D, por lo que la esfera que debe dibujarse es C. Agreguen a su raytracer la funcionalidad de un zbuffer que guarde estas distancias:

```rust
pub fn cast_ray(ray_origin: &Vec3, ray_direction: &Vec3, objects: &[Sphere]) -> Color {
    let mut intersect = Intersect::empty();
    let mut zbuffer = f32::INFINITY;  // what is the closest element this ray has hit?

    for object in objects {
        let tmp = object.ray_intersect(ray_origin, ray_direction);
        if tmp.is_intersecting &&
            tmp.distance < zbuffer { // is this distance less than the previous?
            zbuffer = intersect.distance;  // this is the closest
            intersect = tmp;
        }
    }

    if !intersect.is_intersecting {
        // return default sky box color
        return Color::new(4, 12, 36);
    }

    let diffuse = intersect.material.diffuse;

    diffuse
}
```

Prueben que sus modificaciones funcionaron bien antes de continuar: