Major Program 2: Ray Tracer version 2

Due: November 15 11:59 p.m.

Overview

In this assignment you will extend the first version of the raytracer using the components you have been building in the short programs and in the lab. New elements that must be supported include: *a procedural plane, finite planes,* and diffuse lighting.

Requirements

Your program *must* read the model description from the standard input, write the ppm image to the standard output and write model dump and debugging data to the standard error. You must provide a makefile that will build an executable named *a.out*

You must include at least as much conditionally compiled debugging code as is described in the notes. The previous debugging code and the debugging code for diffuse lighting must be present. You should disable the debugging code in the makefile you submit by commenting out the macro in which it is defined.

Your program must support *infinite plane, sphere, finite plane, and omnidirectional light objects* but may support others.

You must *submit two scene models* of your own design. One must demonstrate *one or more procedural planes* and may include any other object types you desire (including quadrics or other stuff you may have invented). This one must be named *pplane.txt.* Using diffuse illumination in your procedural plane(s) is encouraged but not required. The other one must be named *model.txt,* and should contain one or more examples of each of: *infinite plane, sphere, finite plane, and omnidirectional light objects* but *must not contain any procedural planes or other proprietary extensions*!.   The better model.txt files will be used in testing everyones submissions so it is imperative that they contain only required elements.

Your program must compile without warnings, contain a reasonable level of commenting, consist of independently compiled modules as described in the notes and not contain lines of source code > 72 characters in length.

Sample model files and output data are in *assns/mp2data,* but the ability to process these inputs correctly does *not* guarantee that you have a correct program.