Chapter III

Mandatory part - miniRT

Program name	`miniRT`
Turn in files	`All your files`
Makefile	`all, clean, fclean, re, bonus`
Arguments	`a scene in format *.rt`
External functs.	• `open, close, read, write, printf, malloc, free, perror, strerror, exit` • `All functions of the math library. (Man page: man math.h or man 3 math. Don't forget to compile with the -lm flag.)` • `All functions of the MinilibX`
Libft authorized	`Yes`
Description	`The goal of your program is to generate images using the Raytracing protocol. Those computer-generated images will each represent a scene, as seen from a specific angle and position, defined by simple geometric objects, and each with its own lighting system.`

The constraints are as follows:

- You `must` use the `miniLibX`. Either the version that is available on the operating system, or from its sources. If you choose to work with the sources, you will need to apply the same rules for your `libft` as those written above in `Common Instructions` part.

- The management of your window must remain fluid: switching to another window, minimization, etc..

- You need at least these 3 simple geometric objects: plane, sphere, cylinder.

- If applicable, all possible intersections and the inside of the object must be handled correctly.
- Your program must be able to resize the object's unique properties: diameter for a sphere and the width and height for a cylinder.
- Your program must be able to apply translation and rotation transformation to objects, lights and cameras (except for spheres and lights that cannot be rotated).
- Light management: spot brightness, hard shadows, ambiance lighting (objects are never completely in the dark). You must implement Ambient and diffuse lighting.
- the program displays the image in a window and respect the following rules:
  - Pressing `ESC` must close the window and quit the program cleanly.
  - Clicking on the red cross on the window's frame must close the window and quit the program cleanly.
  - The use of `images` of the `minilibX` is strongly recommended.
- Your program must take as a first argument a scene description file with the `.rt` extension.
  - Each type of element can be separated by one or more line break(s).
  - Each type of information from an element can be separated by one or more space(s).
  - Each type of element can be set in any order in the file.
  - Elements which are defined by a capital letter can only be declared once in the scene.

- Each element first's information is the type identifier (composed by one or two character(s)), followed by all specific information for each object in a strict order such as:
- **Ambient lightning:**

  ```
  A 0.2 255,255,255
  ```

  - identifier: **A**
  - ambient lighting ratio in range [0.0,1.0]: **0.2**
  - R,G,B colors in range [0-255]: **255, 255, 255**
- **Camera:**

  ```
  C  -50.0,0,20     0,0,1   70
  ```

  - identifier: **C**
  - x,y,z coordinates of the view point: **-50.0,0,20**
  - 3d normalized orientation vector. In range [-1,1] for each x,y,z axis: **0.0,0.0,1.0**
  - FOV : Horizontal field of view in degrees in range [0,180]: **70**
- **Light:**

  ```
  L   -40.0,50.0,0.0  0.6     10,0,255
  ```

  - identifier: **L**
  - x,y,z coordinates of the light point: **-40.0,50.0,0.0**
  - the light brightness ratio in range [0.0,1.0]: **0.6**
  - (unused in mandatory part)R,G,B colors in range [0-255]: **10, 0, 255**
- **Sphere**:

  ```
  sp   0.0,0.0,20.6  12.6   10,0,255
  ```

  - identifier: **sp**
  - x,y,z coordinates of the sphere center: **0.0,0.0,20.6**
  - the sphere diameter: **12.6**
  - R,G,B colors in range [0-255]: **10, 0, 255**

- **Plane:**

  ```
  pl  0.0,0.0,-10.0  0.0,1.0,0.0  0,0,225
  ```

  - identifier: **pl**
  - x,y,z coordinates of a point in the plane: **0.0,0.0,-10.0**
  - 3d normalized normal vector. In range [-1,1] for each x,y,z axis: **0.0,1.0,0.0**
  - R,G,B colors in range [0-255]: **0,0,225**

- **Cylinder:**

  ```
  cy  50.0,0.0,20.6   0.0,0.0,1.0  14.2  21.42   10,0,255
  ```

  - identifier: **cy**
  - x,y,z coordinates of the center of the cylinder: **50.0,0.0,20.6**
  - 3d normalized vector of axis of cylinder. In range [-1,1] for each x,y,z axis: **0.0,0.0,1.0**
  - the cylinder diameter: **14.2**
  - the cylinder height: **21.42**
  - R,G,B colors in range [0,255]: **10, 0, 255**

- Example of the mandatory part with a minimalist **.rt** scene:

```
A   0.2                                     255,255,255

C   -50,0,20      0,0,1       70
L   -40,0,30                  0.7           255,255,255

pl  0,0,0         0,1.0,0                   255,0,225
sp  0,0,20                    20            255,0,0
cy  50.0,0.0,20.6 0,0,1.0     14.2  21.42   10,0,255
```

- If any misconfiguration of any kind is encountered in the file the program must exit properly and return "Error\n" followed by an explicit error message of your choice.
- For the defense, it would be ideal for you to have a whole set of scenes with the focus on what is functional, to facilitate the control of the elements to create.