```
# >>>>>>>>>>>    This is a SELF EXPLAINED INPUT FILE.    <<<<<<<<<<<<<

# Every line beginning with '#' will not be read: it's a user comment!

# In this file the user sets the scene content: materials (i.e. colors),
# shapes (i.e. objects) and camera (i.e. observer point of view).

# In this scene we want a checkered blue and green floor, a bright
# shining sky, a red mirroring sphere intersecting the floor on the
# left, a grey sphere quite far in the center and a 3/4 green cylinder on the right.
# The point of view is quite high, looking a bit downward.

#-----------------------------------------------------------------------|
#                               MATERIALS                               |
#-----------------------------------------------------------------------|
# material "mat_name"("BRDF"("pigment"), "emitted_radiance")            |
#                                                                       |
# where:                                                                |
#   "mat_name" =    a name to identify this material                    |
#   "BRDF" =        diffuse                                             |
#                   specular                                            |
#       "pigment" =  uniform(<"r", "g", "b">)                           |
#                    checkered(<"r", "g", "b">, <"r", "g", "b">)        |
#                    image(imageFileName.pfm)                           |
#             where:  "r", "g", "b" are color components intensities,   |
#                     expressed by a floating point number in [0, +infty)|
#                     and imageFileName.pfm is a string                 |
#       "emitted_radiance" = uniform(<"r", "g", "b">)                   |
#                            checkered(<"r", "g", "b">, <"r", "g", "b">)|
#-----------------------------------------------------------------------|
material sky_material(       diffuse(uniform(<0, 0, 0>)),          uniform(<1, 1, 1>)    )
material ground_material(    diffuse(checkered(<0.3, 0.5, 0.1>, <0.1, 0.2, 0.5>, 4)),    uniform(<0, 0,
0>)    )
material sphere_material(    diffuse(uniform(<0.5, 0.5, 0.5>)),    uniform(<0, 0, 0>)    )
material mirror_material(    specular(uniform(<1.5, 0.2, 0.2>)),   uniform(<0, 0, 0>)    )
material cyl_material(       diffuse(uniform(<0.5, 0.7, 0>)),      uniform(<0, 0, 0>)    )
material image_material(     diffuse(image("../Media/Readme_imgs/memorial.pfm")), uniform(<0, 0, 0>))


#-----------------------------------------------------------------------|
#                                SHAPES                                 |
#-----------------------------------------------------------------------|
# "typeofshape"("material", "transformation",                           |
#               **cylinder only: "z_min", "z_max", "radius, "phi_max"**)|
# where:                                                                |
#   "typeofshape" = sphere                                              |
#                   plane                                               |
#                   cylinder                                            |
#       where: parameters are respectively inf and sup base coordinates,|
#              and the cylinder slice angle, in degrees                 |
#   "transformation" =  rotation_x("angle in degrees")                  |
#                       rotation_y("a.i.d.")                            |
#                       rotation_z("a.i.d.")                            |
#                       translation(["dx", "dy", "dz"])                 |
#                       scaling(["stretch_x", "stretch_y", "stretch_z"])|
#       where: every "..." is a floating point number                   |
#              (note that transformations can be combined via a "*",    |
#               but they are NOT COMMUTATIVE!!)                         |
#-----------------------------------------------------------------------|
sphere (sky_material, translation([0, 0, 0.4]) * scaling([200, 200, 200]))
plane (ground_material, identity)
sphere (image_material, translation([10, 0, 1]))
sphere (mirror_material, translation([0, 2, 0]))
cylinder (cyl_material, translation([0, -2.5, 1]), 0, 3, 1, 6.29)


#-----------------------------------------------------------------------|
#                                CAMERA                                 |
#-----------------------------------------------------------------------|
# camera("projection", "transformation", "aspectRatio", "screenDistance")|
#                                                                       |
# where:                                                                |
#   "projection" =  orthogonal,                                         |
#                   perspective                                         |
#   "aspectRatio" = floating point number, defines how larger than the  |
#                   height is the image                                 |
#   "screenDistance" = floating point number, representing how far      |
#                      from the eye of the observer is the screen       |
#                                                                       |
#-----------------------------------------------------------------------|
```

```
camera(perspective, rotation_y(17)*translation([-5, 0, 4]), 1.0, 1.0)


#-------------------------------------------------------------------------|
#                              POINTLIGHT                                 |
#-------------------------------------------------------------------------|
# pointlight("position", "color", "linearRadius")                         |
#                                                                         |
# where:                                                                  |
#   "position" =  3D point identifying the position of the light          |
#                 ["xfloat", "yfloat", "zfloat"]                          |
#                                                                         |
#   "color" = the color of the Point Light                                |
#             <"rfloat", "gfloat", "bfloat">                              |
#                                                                         |
#   "linearRadius" = floating point number used to compute the solid      |
#                    angle subtended by the light at a given distance     |
#                                                                         |
#-------------------------------------------------------------------------|
pointlight([-30, 30, 30], <1, 1, 1>, 0)
```