Overview

The goal with this assignment is to get you comfortable running Geant4 and making minor modifications to existing Geant4 codes before we dive into writing our own from scratch. Refer first to the Application Developer's Guide and other online documentation if you get stuck on something.

Remember, the examples are in your Geant4 install directory, under share/<version>/examples/basic (B1, B3b) and /share/<version>/examples/extended/optical (OpNovice). On the nec-geant server, this is /usr/local/geant4/share/Geant4-10.7.2/examples

Code Submission

Create a folder in your Github repository called **assignment5/**. Here, you should copy and modify the 3 examples we will be using for this assignment: **B1**, **B3b**, and **OpNovice**. Please make sure to use the versions from 10.07, since as we discussed, there were some changes within the last few updates.

For each part of the questions below, create a separate .mac file to accomplish the task. So if part 1 of an example (say, B1) is to run with the default particle type, a particle energy of 5 MeV, and 100 particles, you would create a new .mac called **part1.mac** that contains:

```
/run/initialize
/gun/energy 5 MeV
/run/beamOn 100
```

In cases where I ask for the output, this will make it a lot easier to find the number you're looking for. Put your answers to these questions in **answers.txt** in the folder for that example in your repository. When you get the answer, make sure you're looking at the final, total result, and not the sub-result from a single thread.

Place your .mac files **in the base directory of the example folder** next to the others, such as run1.mac in example B1/. Add your .mac files to, for example, the **EXAMPLEB1_SCRIPTS** variable in CMakeLists.txt so that your files get copied when CMake is run. That way, I can grab your code, run CMake, and be able to access your .mac files easily. **If you just put them in the build directory with files that typically get ignored in the repository, you might not commit them!**

This assignment will be due by **Monday, November 1st at midnight Pacific Time.**

Example B1: Simple Dose Calculation

This example calculates the total dose deposited in a trapezoidal phantom object placed in a box of water.

1. Run 100,000 particles with 511 keV gamma rays (part1.mac)
    a. Question: how much dose was deposited?
2. Run 100,000 particles with 2 MeV neutrons (part2.mac)

    a. Question: how much dose was deposited?
3. Run 100,000 particles with 1 MeV electrons (called e- in your .mac file) (part3.mac)
    a. Question: how much dose was deposited?
4. Change the size of the scoring volume (the trapezoid, G4Trd in the code) by cutting every input parameter to G4Trd in half, then run with **part1.mac**
    a. Question: how much dose was deposited? Why did it go down and not up?
5. Change the G4Cons (the conic section) to a G4Sphere with a diameter of 7 centimeters and run with **part1.mac**
    a. Question: did the dose change? Why or why not?
6. Modify the program to print out the mass of the scoring volume at the end of the program when it prints out the dose. Use the **G4BestUnit** function we discussed in class to use the best units for it.

Example B2b

This example does a (very simple) simulation of a medical PET scanner. It counts "coincident" 511 keV detections (2 energy depositions above a threshold) and outputs the number of usable e+/e- annihilations, along with the dose deposited in the phantom.

1. Run with 100,000 particles (part1.mac)
    a. Question: how many usable annihilations were there?
2. Change the detector material from Lu2SiO5 to NaI (G4_SODIUM_IODIDE) and run **part1.mac**
    a. Question: are there more or less usable annihilations, and why?
3. Question: what energy threshold is used for the detections? Where in the code is it applied?
4. Run with N-13 instead of the default F-18 for the positron-emitting radioisotope (part2.mac)
    a. Question: how do the number of annihilations and dose compare?

Example OpNovice

This example adds optical physics to model an imaginary, scintillating water. All relevant optical and scintillation parameters are defined in the DetectorConstruction, along with the rules for how optical photons should behave at certain interfaces.

1. Run with 1,000 particles using positrons (e+) and with the source placed at position (0, 0, 0) (part1.mac)
    a. Question: how many scintillation photons were created per event?
2. Run with 1,000 particles using 511 keV gamma rays with the source placed at position (0, 0, 0) (part2.mac)
    a. Question: how many scintillation photons were created per event?
    b. How many absorption interactions were there per event?
3. Question: what is the scintillation yield of the scintillator defined in the code?
4. Question: what are the fast and slow time constants of the scintillator defined in the code?
5. Change the absorption length of the scintillator to 1 mm at all energies. Instead of changing every value in the table, use a loop to set the correct number of elements to

1*mm (using the units system we learned about in class). The number of elements should match the number of the **refractiveIndex1** vector.

a. Question: how many absorption interactions were there per event? Did it change as you expected?