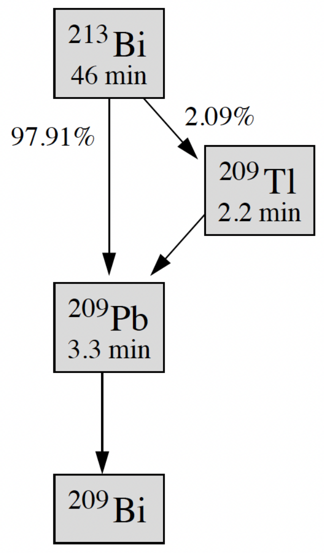
Assignment 8A

In this assignment, you will practice working with random number generators and their applications to physical simulations. You are required to use the *recommended random number generators* discussed in class, as outlined in the lecture notes. Always use a seed, to ensure reproducibility.

- **Problem 1 (20 points):** Rolling Dice
  - (a) Write a program that generates and prints out two random numbers between 1 and 6, to simulate the rolling of two dices.
  - (b) Modify your program to simulate rolling two dice one million times. Count how often a double six appears, then divide this count by one million to obtain the fraction of double sixes. You should get something close to, though probably not exactly equal to $\frac{1}{36}$.
- **Problem 2 (40 points):** Radioactive Decay Chain

  This problem extends the simple radioactive decay simulation covered in the class notes.

  The isotope ^{213}Bi decays to stable ^{209}Bi via one of two different routes, with probabilities and half-lives thus:



  (Technically, ^{209}Bi is not truly stable, but with a half-life exceeding 10^{19} years—over a billion times the age of the universe—it is effectively stable for practical purposes.)

  Starting with a sample consisting of 10,000 atoms of ^{213}Bi, simulate the decay of the atoms as shown in the example problem in notes by dividing time into slices of length $\delta t = 1s$ each and on each step doing the following:

(a) For each atom of ^{209}Pb in turn, decide at random, with the appropriate probability, whether it decays or not. The probability is given in the class notes:

$$P(t) = 1 - 2^{-t/\tau}\,.$$

Count the total number that decay, subtract it from the number of ^{209}Pb atoms, and add it to the number of ^{209}Bi atoms.

(b) Now do the same for ^{209}Tl, except that decaying atoms are subtracted from the total for ^{209}Tl and added to the total for ^{209}Pb.

(c) For ^{213}Bi, the situation is more complicated: when a ^{213}Bi atom decays, you have to decide at random with the appropriate probability the route by which it decays. Count the numbers that decay by each route and add and subtract accordingly.

Note that you have to work up the chain from the bottom like this, not down from the top, to avoid inadvertently making the same atom decay twice on a single step. Keep track of the number of atoms of each of the four isotopes at all times for 20,000 seconds and make a single graph showing the four numbers as a function of time on the same axes.

- **Problem 3 (40 points):** Brownian Motion

  Brownian motion is the motion of a particle, such as a smoke or dust particle, in a gas, as it is buffeted by random collisions with gas molecules. Make a simple computer simulation of such a particle in two dimensions as follows.

  The particle is confined to a square grid or lattice $L \times L$ squares on a side, so that its position can be represented by two integers $i, j = 0, \ldots, L-1$. It starts in the middle of the grid. On each step of the simulation, choose a random direction—up, down, left, or right—and move the particle one step in that direction. This process is called a random walk. The particle is not allowed to move outside the limits of the lattice—if it tries to do so, choose a new random direction to move in.

  Write a program to perform a 10000 steps of this process on a lattice with $L = 101$ and make an animation on the screen of the position of the particle. (We choose an odd length for the side of the square so that there is one lattice site exactly in the center.)

  If you are unsure how to animate, provide a few snapshots of the particle at different points in its motion. **There will not be any point deduction if the animation does not work**.

  Note: I recommend first writing a program that generates and saves each frame as a labeled JPEG figure. Once the frames are ready, if time allows, a separate script can be used to create an animation from them. For example, this can be done using the `imageio` and `os` packages.

  For the animation step ***only*** (once the frames have already been generated with a self-written program), students are permitted to use any language model of their choice (e.g., ChatGPT, Claude, DeepSeek) to obtain a script that assembles the frames into an animation.