AP Computer Science A@Beijing National Day School

Problem Set 9: `Radiation Exposure`

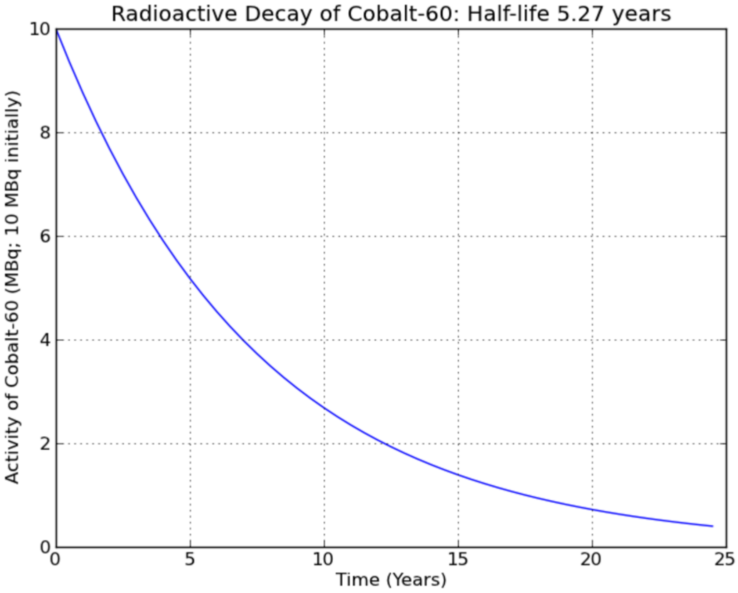
Due date: Monday, November 19, 2018
Instructor: Mr. Alwin Tareen
Total Points: 15

Task Overview

- Implement a program that determines the amount of radiation a person is exposed to, in a given period of time.

Background

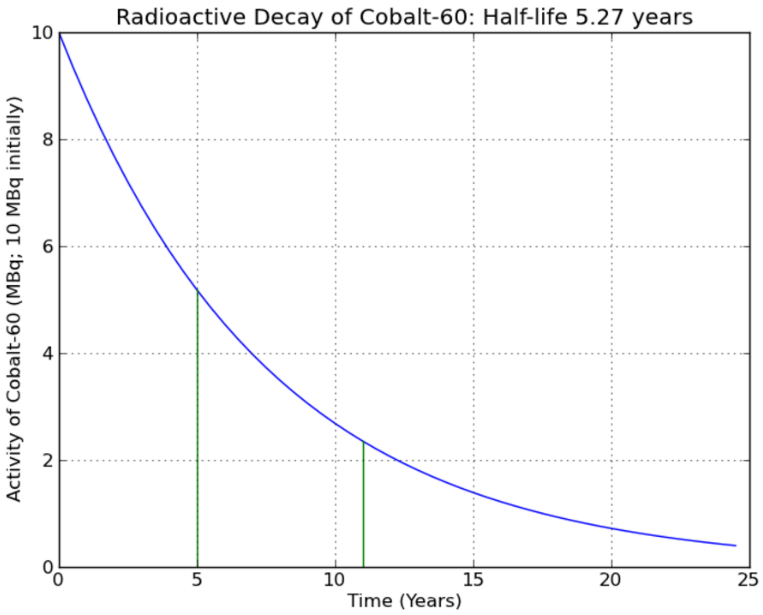
- *Radioactive decay* is the process by which an unstable atom loses energy and emits ionizing particles, in what is commonly referred to as radiation. Exposure to radiation can be dangerous, and is very important to measure, to ensure that a person is not exposed to a large amount of it.
- The radioactivity of a material decreases over time, as the material decays. A radioactive decay curve describes this decay. The x-axis measures time, and the y-axis measures the amount of *activity* produced by the radioactive sample.
- *Activity* is defined as the rate at which the nuclei within the sample undergo transitions. Put simply, this measures how much radiation is emitted at any one point in time.
- The measurement of activity is called the Becquerel(Bq). The following is a sample of a radioactive decay curve.



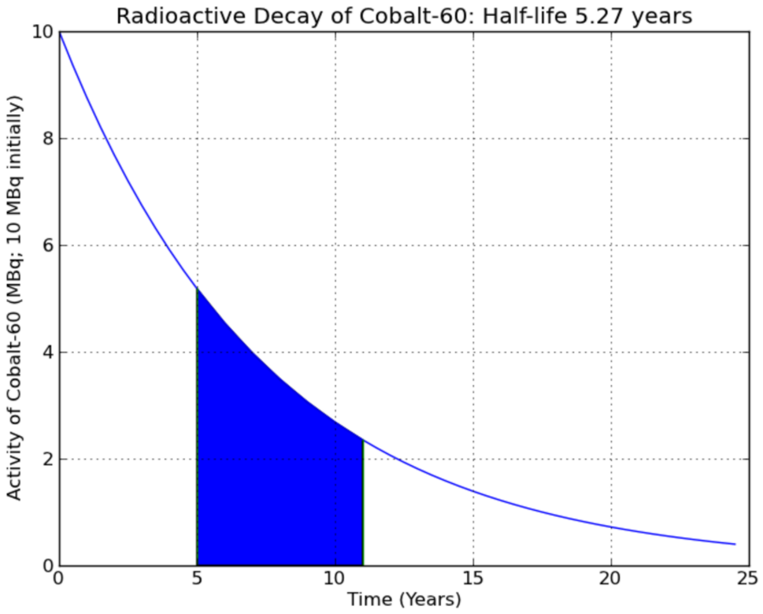
The Problem Scope

- Here's the problem we'd like to solve. Let's say that Sarina has moved into a new apartment. Unknown to her, there is a sample of Cobalt-60 inside one of the walls of the apartment. Initially, that sample had 10MBq of activity, but she moves in after the sample had been there for 5 years. She lives in the apartment for 6 years, then leaves. How much radiation was she exposed to?

- We can actually figure this out using the radioactive decay curve from above. What we want to know is her *total radiation exposure* from year 5 to year 11.

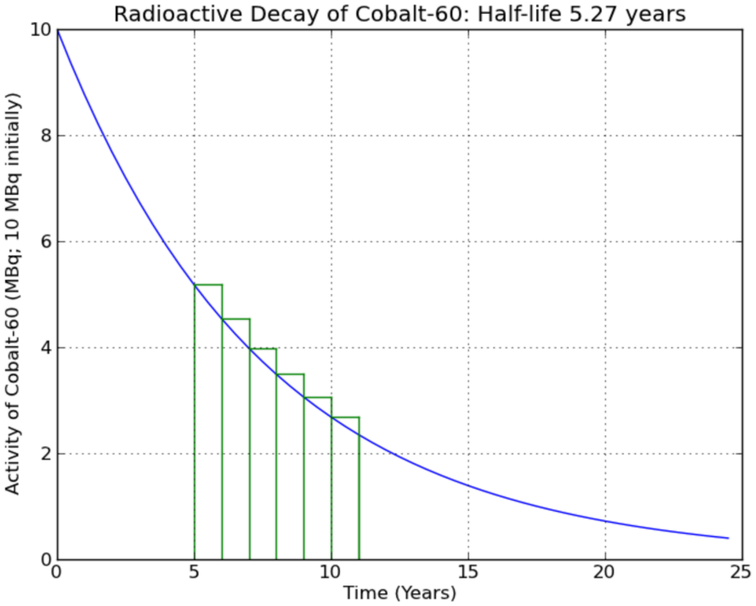


- Total radiaton exposure corresponds to the area between the two green lines at *time = 5* and *time = 11*, and under the blue radioactive decay curve. This should make intuitive sense, if the x axis measures time, and the y axis measures activity, then the area under the curve measures (time * activity), or (years * MBq), which is approximately the total number of MBq that Sarnia was exposed to during her time in the apartment.



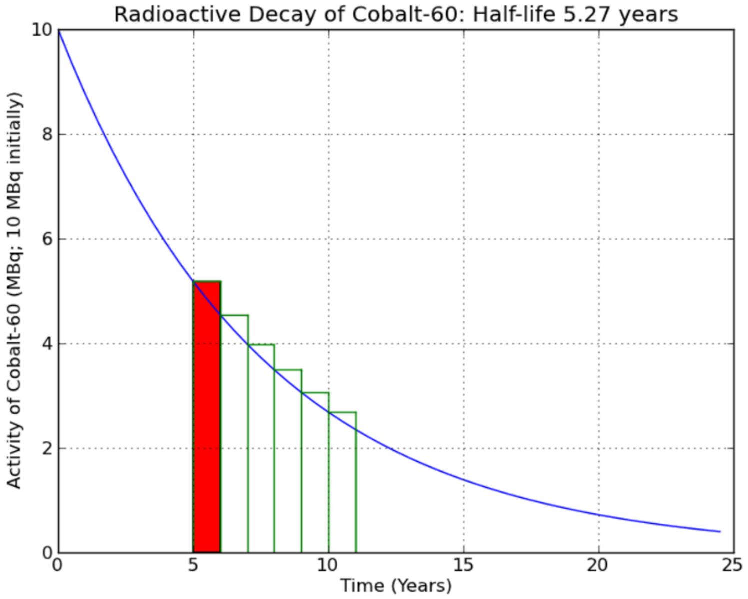
- Now, how do we calculate this area? Unlike a simple shape, like a square, or a circle, we have no easy way to determine what the area under this curve is.

- However, there is a technique that can provide us with some assistance, namely, *approximation*. Let's use an approximation algorithm to estimate the area under the curve. We can do so by first splitting up the area into equally-sized rectangles. In this case, there are six of them, corresponding to one rectangle per year:



- Once we've done that, we can figure out the area of each rectangle pretty easily. Recall that the area of a rectangle is determined by multiplying the height of a rectangle by its width. The height of this rectangle is the value of the curve at `5.0`. If the curve is described by a function, `f(x)`, then we can obtain the value of the curve by calculating `f(5.0)`.

  ```
  f(5.0) = 5.181
  ```



- The width of this rectangle is `1.0`. So, the area of this single rectangle is: `1.0 * 5.181 = 5.181`. To approximate how much radiation Sarina was exposed to, we then calculate the area of each successive rectangle, and then sum up the areas of each rectangle to get the total. When we do this, we find that Sarina was exposed to `22.94241041057671 MBq` of radiation.

Specification

The Information Box Which Includes Your Name[5 points]

- Type your English and Pinyin name into the `Author` field, where it says: `YOUR NAME HERE`

Compute the Amount of Radiation Exposure in a Given Time Period [10 points]

- Write a `Java` program in the file `RadiationExposure.java` which calculates the amount of radiation exposure in a given time period.
- You will write your solution in a function called `public static double decayCurveArea(int start, int stop, double step)` right below the place where it says: `YOUR CODE HERE`.
- Make sure that you run your `Java` program, and ensure that it is free of errors. If the following statements are executed:

  ```
  double result = decayCurveArea(5, 11, 1);
  System.out.println(result);
  ```

  Then the output of your program should be:

  ```
  22.94241041057671
  ```

Hints

- In this problem set, you are asked to find the amount of radiation that a person is exposed to, during some period of time. In order to complete this assignment, you'll need to know what the value of the radioactive decay curve is, at various points. There is a function `f(x)` that has been provided for you, which mathematically describes the radioactive decay curve for this problem.
- This problem set is relatively straightforward if you only consider the cases where the widths of the rectangles are integer values. Those cases can be handled with a simple `for` loop.
- However, if you examine the provided test bench, you should notice that some of the rectangle widths are decimal values, like `0.5`.
- In other words, you will need to seek an approach that can handle decimal-valued rectangle widths.
- You should find out the number of rectangles that are within the defined region. How would you calculate this number? Perhaps the following equation would work:

  ```
  int gaps = (int)((stop-start)/step);
  ```

- Then, you should create an array which has the same size as the number of rectangles, with all of its elements initialized to `0.0`. Perhaps you should call this array `timePosts`.
- Next, you should loop across the `timePosts` array, and fill it with `time` values that correspond to the position of each rectangle. For the example described in the Background section, the `timePosts` array would be as follows:

  ```
  [5.0, 6.0, 7.0, 8.0, 9.0, 10.0]
  ```

- Then, it is simply a matter of looping across the `timePosts` array, calculating the product of `f(x)*step`, and summing each of these products to result in the area under the curve.

Testing

- The file RadiationExposureJUnitTest.java contains the JUnit test cases which verify the correct functionality of the program.

Submission

- Submit your Java program by uploading it to the Web-CAT automated grading platform. Click on the following link:
  http://ec2-54-65-207-33.ap-northeast-1.compute.amazonaws.com:8080/Web-CAT/WebObjects/Web-CAT.woa