**Java - Competitive Programming Challenge**

**Question 1**

Given an array of long integers (arr) and a number (num). Iterate through the elements in arr and double the value of num whenever an element equals num. arr can be reordered before the iteration to maximize the value of num. Find the maximum possible value of num.

**Example**

arr = [1, 2, 4, 11, 12, 8]

num = 2

Iterating through arr:

|  |  |
| --- | --- |
| **arr** | **num = 2** |
| 1 | 2 |
| 2 | 4 |
| 4 | 8 |
| 11 | 8 |
| 12 | 8 |
| 8 | 16 |

**Question 2**

A music player allows users to choose songs to play, but only in pairs and only pairs of songs with durations that add up to a multiple of 60 seconds (e.g., 60, 120, 180). Given a list of song durations, calculate the total number of different song pairs that can be chosen.

**Example**

n = 3

songs = [40, 20, 60]

One pair of songs can be chosen whose combined duration is a multiple of a whole minute (40 + 20 = 60) and the return value would be 1. While the third song is a single minute long, songs must be chosen in pairs.

**Function Description**

Develop the function playlist.

playlist has the following parameter(s):

* int songs[n]: array of integers representing song durations in seconds

**Returns:**

* int: the number of songs pairs that add up to a multiple of a minute

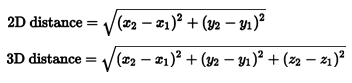
**Constraints**

* 1 ≤ n ≤ 105

1 ≤ songs[i] ≤ 1000, where 0 ≤ i < n

**Question 3**

This challenge involves points in two and three dimensional space. The classes and methods to implement will store values for coordinates as well as calculate distances between points. The 2D and 3D distances between two points are calculated using the following formulae:



Implement the classes and methods defined below:

A superclass named Point2D:

|  |  |
| --- | --- |
| **Instant Variable** | **Functionality** |
| X | Stores the value of the x-coordinate. |
| Y | Stores the value of the y-coordinate. |
| **Constructor** | **Functionality** |
| Point2D(…) | A parameterized constructor that initializes the instance variables. |
| **Methods** | **Functionality** |
| double dist2D(Point2D p) | Calculates and returns the 2D distance between two points (the current Point2D object and Point2D parameter p). |
| void printDistance(double d) | Prints the 2D distance between two points as 2D distance = k, where k is distance d as a ceiling-rounded integer, on a new line. |

A derived class named Point3D that extends Point2D:

|  |  |
| --- | --- |
| **Instant Variable** | **Functionality** |
| Z | Stores the value of the z-coordinate. |
| **Constructor** | **Functionality** |
| Point3D(…) | A parameterized constructor that initializes the instance variables. |
| **Methods** | **Functionality** |
| double dist3D(Point3D p) | Calculates and returns the 3D distance between two points (the current Point3D object and Point3D parameter p). |
| void printDistance(double d) | Prints the 3D distance between two points as 3D distance = k, where k is distance d as a ceiling-rounded integer, on a new line. |

A main method is provided in the locked portion of the editor. It parses six values representing point coordinates and calls the implemented constructors and methods. Here, x[1], y[1], and z[1] represent the coordinates of the first point, and x[2], y[2], and z[2] represent the coordinates of the second point. Note that printed output must exactly match the above for the test cases to pass.

**Constraints**

-128 ≤ x,y,z ≤ 127