Problem Statement

Your favorite local restaurant has caught onto the health food trend, and has begun publishing basic nutritional information for all of their most popular menu selections. This is excellent timing, since your doctor has recently recommended various diet plans for you and several of your friends.

You are given int[] **protein**, int[] **carbs**, and int[] **fat**, representing the number of grams of protein, carbs, and fat in each of the available menu items. Elements from each int[] correspond to the elements of the other int[]s with the same index.

You are also given a String[] **dietPlans**, indicating the doctor's recommendation for how each person should make their meal selection. Each element of **dietPlans** describes the diet plan for an individual. Each character of each element of **dietPlans** specifies, in order of importance, a selection criteria, defined in the following way:

- 'C' = high carbs
- 'c' = low carbs
- 'P' = high protein
- 'p' = low protein
- 'F' = high fat
- 'f' = low fat
- 'T' = high calorie
- 't' = low calorie

As an example, the diet plan "tf" means the doctor recommends a meal with the lowest possible calories, and if more than one is tied, the one with less fat should be selected. Whenever more than one meal is tied according to the diet plan, then the one with a lower index should be selected.

The restaurant sloppily neglected to list the total calorie count on the menu. Fortunately, you happen to remember from days gone by that one gram of fat contains 9 calories, and one gram of carbs or protein contains 5 calories.

You are to return a int[] indicating the indexes of the menu selections that best suit each person's diet plan (indexed from 0). The return int[] should have the same number of elements as **dietPlans**, and each value of the return int[] should correspond to the element of **dietPlans** with the same index.

Definition

Class:	HealthFood
Method:	selectMeals
Parameters:	int[], int[], int[], String[]
Returns:	int[]
Method signature:	int[] selectMeals(int[] protein, int[] carbs, int[] fat, String[] dietPlans)
(be sure your method is public)	

Notes

- When no diet plan is specified, the default should be the menu selection with the lowest index (0).

Constraints

- **carbs** will contain between 1 and 50 elements, inclusive
- **protein** will contain between 1 and 50 elements, inclusive
- **fat** will contain between 1 and 50 elements, inclusive.
- **carbs**, **protein**, and **fat** will each contain the same number of elements.
- Each element of **carbs**, **protein**, and **fat** will be between 0 and 100, inclusive.
- **dietPlans** will contain between 1 and 50 elements, inclusive.
- Each element of **dietPlans** will be between 0 and 4 characters in length, inclusive, and will contain only the characters "CcPpFfTt".

Examples

0)

```
{3, 4}
{2, 8}
{5, 2}
{"P", "p", "C", "c", "F", "f", "T", "t"}
Returns: { 1,  0,  1,  0,  0,  1,  1,  0 }
```

This is a simple menu, with only two selections. We see each of the simplest diet plans here.

1)

```
{3, 4, 1, 5}
{2, 8, 5, 1}
{5, 2, 4, 4}
{"tFc", "tF", "Ftc"}
Returns: { 3,  2,  0 }
```

Note here that lowest total calories is tied between items 2 and 3. Note also that both of those items are tied for fat content. So, when we have lowest carbs as a tie-breaker, item 3 is selected. When there is no further tiebreaker, we select the one with lowest index. Note also that if highest fat is the first requirement, then the tiebreaker is irrelevant since item 0 has more fat than items 2 or 3.

2)

```
{18, 86, 76,  0, 34, 30, 95, 12, 21}
{26, 56,  3, 45, 88,  0, 10, 27, 53}
{93, 96, 13, 95, 98, 18, 59, 49, 86}
{"f", "Pt", "PT", "fT", "Cp", "C", "t", "",
 "cCp", "ttp", "PCFt", "P", "pCt", "cP", "Pc"}
Returns: { 2,  6,  6,  2,  4,  4,  5,  0,  5,  5,  6,  6,  3,  5,  6 }
```