Student Name:

Student ID :

Problem Statement A1

This problem is about a traditional way of naming the second child in a family in middle east. In this problem, we will assume that the people whose first name starts with a vowel are female and all other people are male. Note that in this problem the letter Y is considered a vowel (so the vowels are A, E, I, O, U, and Y). All people in this problem have exactly two names, e.g. Yasaman Sadat (female) or Mohammad Reza (male).

You are given the Strings Father, Mather and Child1: the names of a dad, a mom, and their first child. You are also given the String Gender: the gender of their second child (either "Boy" or "Girl"). Your task is to determine and return the name for the second child according to the rules given below.

Rule 1 - If the gender of the first child differs from the second child, the second child will use both names of the parent with the same gender, in reversed order.

Rule 2- If both children have the same gender, the second child will get its first name from the parent with the same gender as itself (i.e., girls from their mom, boys from their dad) and its second name from its older sibling.

Definition

Class: SecondChild
Method: chooseName
Parameters: string, string, string, string
Returns: string
Method signature: string chooseName(String Father, String Mather, String Child1, String Gender)
(be sure your method is public)

Notes

You should always follow the rules in the problem statement, even if the name they produce does not match the child's gender (see Example #1) or produces the same name as the first child had (see Example #4).

Constraints

- Each name (Father, Mather, Child1) will contain at most 20 characters.
- Each name will have the form "First Second", with exactly one space and with exactly the first letter of each name in uppercase.
- Father will start with a consonant and Mather will start with a vowel (AEIOUY).
- Gender will be either "Boy" or "Girl".

Examples

0) "Mohammad Reza"
"Yasaman Sadat"
"Baqer Ali"
"Boy"

Returns: "Mohammad Ali"
Both children are boys, so rule 2 applies: the child gets his first name ("Mohammad") from his father and his second name ("Ali") from his brother.

1) "Mohammad Reza"
"Yasaman Sadat"
"Baqer Ali"
"Girl"
Returns: "Sadat Yasaman"
The children have different genders. Thus, the child's name is obtained by swapping her mother's two names. **Note** that the returned name is not a proper name for a girl, but that's what the rules produced, so that's what you should return.

2) "Mohammad Reza"
"Yasaman Sadat"
"Umi Kulsum"
"Girl"
Returns: "Yasaman Kulsum"
Another rule 2 case, but this time the child gets her name by combining the names of her mother and her sister.

3) "Mohammad Reza"
"Yasaman Sadat"
"Umi Kulsum"
"Boy"
Returns: "Reza Mohammad"

4) "Mohammad Ali"
"Yasaman Sadat"
"Mohammad Reza"
"Boy"
Returns: "Mohammad Reza"

5) "Dhikrullah Ali"
"Umi Kulsum"
"Reza Hosseinzadeh"
"Boy"
Returns: "Dhikrullah Hosseinzadeh"

Note that the second child's name can sometimes have more than 20 characters.

Problem Statement A2

There are some chickens and some cows in Farmer Cameron's yard.

Cameron's daughter Susie counted that all the animals in the yard have a total of 3 heads. Cameron's son Billy counted their legs and got a total of 8. Using their answers, Farmer Cameron easily determined that there have to be exactly 2 chickens and 1 cow.

Write a method that will solve a general version of Farmer Cameron's problem. You are given two ints heads and legs. Compute the number of chickens and the number of cows. Return a vector with two elements: first the number of chickens, then the number of cows. If there is no solution, return an empty int[] instead.

Definition

Class: ChickensAndCows
Method: howMany
Parameters: int, int
Returns: vect<int>
Method signature: vect<int> howMany(int heads, int legs)
(be sure your method is public)

Notes

- If the solution exists, it is always unique.
- A chicken has 1 head and 2 legs. A cow has 1 head and 4 legs.

Constraints

- heads will be between 0 and 1,000,000, inclusive.
- legs will be between 0 and 1,000,000, inclusive.

Examples

0) 3
8
Returns: {2, 1 }
Two chickens and a cow have a total of three heads and eight legs.

1) 10
40
Returns: {0, 10 }
Ten cows.

2) 10
42
Returns: { }

This test case has no solution because the number of legs is too large (or the number of heads is too small).

3) 1
3
Returns: { }
No set of animals can have one head and three legs.

4) 0
0
Returns: {0, 0 }

Problem Statement B1

Austria is suffering from a heavily snowstorm. The heavy storm is causing the closure of an important highway connecting the north and south of the Alps. You've got several reports containing the start and end points of highway segments covered by heavy snow.

Given those reports as two vectors <int> startPoints and endPoints, you are to return the total length of highway segments covered by snow. Note that the reported segments may overlap.

Definition

Class: Snowstorm
Method: blockedLength
Parameters: vector <int>, vector<int>
Returns: int
Method signature: int blockedLength(vector<int> startPoints, vector<int> endPoints)
(be sure your method is public)

Constraints

- startPoints will contain between 1 and 50 elements, inclusive.
- endPoints will contain the same number of elements as startPoints.
- Each element of startPoints and endPoints will be between 0 and 10000, inclusive.
- The i-th element of startPoints will be smaller than the corresponding element in endPoints.

Examples

0) {17, 85, 57}
{33, 86, 84}
Returns: 44
These segments don't overlap, so the total length is the sum of each segment.

1) {45, 100, 125, 10, 15, 35, 30, 9}
{46, 200, 175, 20, 25, 45, 40, 10}
Returns: 132
In this case some segments overlap. There are 3 sections covered by snow: 9-25, 30-46 and 100-200.

2) {4387,711,2510,1001,4687,3400,5254,584,284,1423,3755,929,2154,5719, 1326,2368,554}
{7890,5075,2600,6867,7860,9789,6422,5002,4180,7086,8615,9832,4169, 7188,9975,8690,1423}
Returns: 9691
The snow covers a single section 284-9975.

3)
{4906,5601,5087,1020,4362,2657,6257,5509,5107,5315,277,6801,2136,2921,5233,5082,497,8250,3956,5720}
{4930,9130,9366,2322,4687,4848,8856,6302,5496,5438,829,9053,4233,4119,9781,8034,3956,9939,4908,5928}
Returns: 9510
There are 2 highway sections covered by snow: 277-4930 and 5082-9939.

4)
{51,807,943,4313,8319,3644,481,220,2161,448,465,1657,6290,22,6152,647,3185,4474,2168}

{1182,912,1832,7754,9557,7980,4144,3194,7129,5535,1172,2043,6437,7252,9508,4745,8313,8020,4017}
Returns: 9535

5)

{8786,7391,201,4414,5822,5872,157,1832,7487,7518,2267,1763,3984,3102,7627,4099,524,1543,1022,3060}
{9905,7957,3625,6475,9314,9332,4370,8068,8295,8177,7772,2668,7191,8480,9211,4802,2625,1924,9970,4180}
Returns: 9813

Problem Statement B2

A string composed of a's and b's is either simple or composite. A string is composite if and only if it can be expressed as a shorter string replicated multiple times.

For example, the string "abba" is a simple string and the string "ababab" is composite since it is 3 copies of "ab" concatenated together.

The string "ab" is called a root of "ababab". The string "aaaa" has 2 roots: "aa" and "a". Note every composite string has exactly 1 simple root, "a" in the example below but some string could have many composite roots.

In this problem, we are considering strings made up using only a's and b's. Return the number of composite strings that we can have for the given length.

Definition
Class: CompositeS
Method: howMany
Parameters: int
Returns: int
Method signature: int howMany(int length)
(be sure your method is public)

Constraints
- length will be between 1 and 60 inclusive.

Examples
0) 1
Returns: 0
There are no redundant strings of length 1.

1) 2
Returns: 2
Both "aa" and "bb" are redundant.

2) 4
Returns: 4
Here the redundant strings are "aaaa", "bbbb", "abab", and "baba".

3) 10
Returns: 34

4) 30
Returns: 33814