**Function Name:** `corruptedMessage`

**Inputs:**

1. (*char*) The corrupted message
2. (*char*) The string used to fix the message

**Outputs:**

1. (*char)* The fixed message

**Background:**

Every friend group has the one friend who owns an Android phone and sends green texts in the group chat. One day, that friend sent a text that got corrupted so that some letters in the message were replaced by question marks, and in an attempt to redeem themselves, they sent a second text with seemingly random letters. Replacing the question marks in the first text with the corresponding letters from the second text is the only way to fix the text and understand what the friend wanted to say.

**Function Description:**

Given two character vectors, replace all the question marks in the first string with the corresponding letters from the second string.

**Example:**

```
str1 = 'And??id is ?ct?a?ly ?uper?or!'
str2 = 'hturomngth aojuelioxsokdeioro'
fixedStr = corruptedMessage(str1, str2)
fixedStr →  'Android is actually superior'
```

**Notes:**

- The two strings are guaranteed to be the same length.

**Function Name:** `olympicTryouts`

**Inputs:**

1. (*double*) 1xN vector of Athlete IDs
2. (*double*) 1xN vector of 100m times (seconds)
3. (*logical*) 1xN vector representing if they made the high jump
4. (*double*) 1xN vector of discus distances (meters)
5. (*double*) 1xN vector of long jump distances (meters)

**Outputs:**

1. (*double*) 1xM vector of Athlete IDs who passed
2. (*double*) The number of Athletes who passed

**Background:**

There are too many people attempting to tryout for the Olympics this year, even the stadium can barely hold everyone. As such you have been elected to reduce the numbers of athletes allowed to tryout. You have to find a way to select only athletes who pass certain metrics.

**Function Description:**

Given a set of athlete IDs and various information about the athlete, you must determine if an athlete passes all the following set of requirements.

1. The Athlete's 100m time does not exceed 13 seconds.
2. The Athlete either made the high jump OR has a long jump distance greater than 6 meters.
3. The Athlete threw the discus 60 meters or farther.

If an athlete passes, return their ID in the first output. Return the total number of athletes who passed in the second output.

**Example:**

```
IDs         = [98,    2,     43,    10,    5];
runTimes    = [12.98, 10.76, 11.21, 13.09, 12.21];
highJump    = [true,  false, false, true,  true];
discusDists = [62.1,  63.2,  60.0,  51.1,  21.9];
ljDists     = [5.21,  6.00,  8.11,  7.35,  6.56];
[passed, numPass] = olympicTryouts(IDs, runTimes, highJump,...
                                   discusDists, ljDists);
passed → [98, 43]
numPass → 2
```

**Hints:**

- Try writing the problem description first as a single statement then turn that statement into code. (e.g. The athlete must pass condition 1 and either condition 2 or condition 3.)

**Function Name:** `yeaOrNay`

**Inputs:**

1. (*char*) 1x4N character vector of 'Yea' and 'Nay'
2. (*double*) 1xN vector of volunteer IDs
3. (double) The number of volunteers needed

**Outputs:**

1. (*double*) 1xM vector of IDs who were selected

**Background:**

You are running the volunteer pool for the Olympic Marathon Tryouts. Unfortunately, your bosses have declined to give you any resources to organize the multitudes of volunteers required for this event. Fortunately, you don't need any resources other than MATLAB!

**Function Description:**

You are given a character vector consisting of the words `'Yea'` and `'Nay'` separated by spaces where the first word represents the first volunteer's answer, the second represents the second volunteers answer, and so on. First find all those who answered `'Yea'`. Then select the number of volunteers given by the third input from the front of those who answered `'Yea'`.

**Example:**

```
answers = 'Yea Nay Yea Yea Nay ';
volunteers = [21, 5, 31, 10, 2];
needed = 2;
selected = yeaOrNay(answers, volunteers, needed);
selected → [21 31]
```

**Notes:**

- There will always be enough `'Yea'` answers to fill all selected spots.
- The string will always follow this capitalization pattern.

**Hints:**

- Think of how you can reduce the number of characters in the first input without reducing the amount of information contained.

**Function Name:** `heatPlanner`

**Inputs:**

1. (*double*) The number of athletes competing in an event

**Outputs:**

1. (*char*) A description of each heat of the event

**Background:**

As part of the Olympic Planning Committee, you must organize the schedule of events. For many events in swimming and track, there are so many athletes that races must be done in multiple heats. To make sure everything is as fair as possible, you decide that each heat of an event should have the same number of athletes. You turn to MATLAB to help do all the calculations quickly and efficiently.

**Function Description:**

Your function will take in a single number. First, find the largest divisor of the number (see the Hint below on how you can achieve this). This will be the number of heats of the event. Then, find the number of athletes that will be competing in each of these heats. Finally, output your results as a string of the format

```
'There will be <number of heats> heat(s), each with <number of athletes per
                              heat> athletes!'
```

**Example:**

```
str = heatPlanner(15);
str → 'There will be 5 heat(s), each with 3 athletes!'
```

**Hints:**

- The `mod()` function can take vectors of numbers as inputs. Try experimenting with a vector as the second input to `mod()`, to test divisibility of multiple numbers at once!

**Function Name:** `eventList`

**Inputs:**

1. (*char*) A 1x(6*N) vector of Olympic events
2. (*logical*) A 1xN vector of whether or not the athlete competes in the event

**Outputs:**

1. *(char*) A 1xP vector of the Olympic events that the athlete doesn't compete in
2. (*double*) A 1xM vector of the original positions of the events the athlete competes in

**Background:**

You just started working for the Olympic Committee and on your first day the system mixed up what events different athletes compete in. As MATLAB is your specialty, you've been tasked with determining the correct events for a certain athlete! Fix the Summer Olympics!

**Function Description:**

Given a list of the first 5 letters of different Olympic events and a logical vector of whether the athlete competes in that event, output the events that should not be included in the athlete's event list. You should also output a vector of the positions of the events that belong in ascending order.

**Example:**

```
events = 'TRACK CYCLI SWIMM BOXIN DIVIN FENCI '
belong = [true false true false true true]
[wrong, positions] = eventList(events, belong)
wrong → 'CYCLI BOXIN '
positions → [1 3 5 6]
```

**Notes:**

- Event names will always be five capital letters followed by a space, both in the input and output.
- You will not be given a test case with no falses in the logical input vector.