Function Name: `haveEnoughWings`

Inputs:

1. (*double*) The number of wings ordered
2. (*double*) The number of people at the party
3. (*double*) The number of wings per person

Outputs:

1. (*char*) An output string depending on if you have enough wings

Topics: (*logical operators*), (*if conditional*)

Background:

You are planning a party for the Superbowl and need to know if you've ordered enough wings for everyone. Since you will be inviting all of your CS1371 classmates and TAs to the party, you have to use MATLAB to answer this urgent question.

Function Description:

Determine if the ordered number of wings is more than, less than, or equal to the number of needed wings. Your output string should be in the following format depending on this condition:

- If you have more than enough wings:
  - `'We have enough wings! In fact, we have <number of extra wings> left over!'`
- If you don't have enough wings:
  - `'Call Wingzone immediately! We need <number of additional wings to buy> more wings!'`
- If you have exactly enough wings:
  - `'Wow! We have exactly enough wings for the party!'`

Example:

```
wingsOrdered = 600;
numPeople = 75;
wingsPerPerson = 10;
sentence = haveEnoughWings(wingsOrdered, numPeople, wingsPerPerson);
sentence → 'Call Wingzone immediately! We need 150 more wings!'
```

Hints:

- The num2str() function may be helpful

Function Name: `whoGonnaWin`

Inputs:

1. (*double*) A 1x2 vector containing the 49ers' score counts from the first two quarters
2. (*double*) A 1x2 vector containing the Chiefs' score counts from the first two quarters

Outputs:

1. (*char*) An output string predicting who is going to win the game

Topics: (*if conditional*), (*series of conditionals*)

Background:

After an exhausting first half of the Superbowl with the extra wings barely arriving in time, it's finally half time and there's chaos in the lounge. Your classmates are yelling about how the Chiefs are going to destroy the 49ers in the second half despite a rocky start, yet the TAs laugh it off and are assured that the 49ers will maintain their lead. In order to not have to choose between your classmates and your grade, you resort to MATLAB to solve the argument!

Function Description:

Given the scores of each team from the first two quarters, return a statement declaring what the outcome of the game will likely be based off of the net scores. The output statements should be structured as detailed below:

- If one team has more than 10 points over the other:
  - `'The <winning team> have a lead of <point difference> points! It looks like the <losing team> aren't catching up!'`
- If one team is up by 10 or fewer points over the other:
  - `'The <winning team> have a lead of <point difference> points! The <losing team> may still make a comeback though!'`
- If the two scores are tied:
  - `'It's a complete draw! Buckle in for an exciting second half!'`

Example:

```
score49ers = [0 7];
scoreChiefs = [14 3];
sentence = whoGonnaWin(score49ers, scoreChiefs);
sentence → 'The Chiefs have a lead of 10 points! The 49ers may still
           make a comeback though!'
```

Notes:

- The names to use for the two teams in the output are '49ers' and 'Chiefs' respectively.
- The net scores are calculated by summing the scores of each quarter.
- You can put an apostrophe (') into your char by placing two of them in consecutive order.
  - e.g. `phrase = 'How''s it going?'`

Function Name: `isReceiverOpen`

Inputs:

1. (*char*) A 1xN string containing only the characters `' '` and `'R'`
2. (*char*) A 1xN string containing only the characters `' '` and `'D'`

Outputs:

1. (*char*) A descriptive string detailing the outcome of the pass attempt

Topics: (*masking*), (*mask combination*)

Background:

Legendary Georgia Tech kicker Harrison Butker is working on a method to help the Kansas City Chiefs' quarterback coach track the results of Patrick Mahomes' pass attempts, so he turns to his Alma Mater, and its group of expert MATLAB students, to help him out.

Function Description:

You are given two strings of equal length. The first string details the field locations of receivers the quarterback can possibly pass to. A receiver is designated by the character `'R'`, and an empty field location is designated by a space. The second string details the field locations of defenders who are attempting to stop the pass. A defender is denoted by the character `'D'`, and an empty field location is designated by a space. Each index in the string represents a field location 5 yards from the quarterback.

Your job is to find if any receiver is open. A receiver is open if there is no defender at the same field location. If there is an open receiver, output the string `'Patrick Mahomes completed the pass for <number of yards> yards!!'`, where `<number of yards>` is how many yards away the open receiver is. If no receiver was open, output the string `'Patrick Mahomes' pass was incomplete!'`

Example:

```
res = isReceiverOpen('R  R  R','D  DD  ')
res → 'Patrick Mahomes completed the pass for 35 yards!!'
```

Notes:

- You are guaranteed to have only either **one** or **no** receiver open
- There can be any number of defenders
- Each index represents 5 yards. A receiver at index 1 is 5 yards away from the quarterback; a receiver at index 2 is 10 yards away, so on and so forth.

Hints:

- Remember how to combine and manipulate masks (&, |, ~)

Function Name: `playoffCommittee`

Inputs:

1. (*char*) NFL Team 1
2. (*char*) NFL Team 2

Outputs:

1. (*char*) A sentence describing which team will make the playoffs

Topics: (*switch conditionals*), (*nested conditionals*)

Background:

The NFL Playoffs are about to begin, and you've been selected to determine which teams play past the regular season! There are a handful of teams that haven't secured a playoff berth yet, and it is your job to determine which of the lucky few will make it over the others. As a MATLAB whiz, you decide to write code that will determine the teams that make it into the playoffs for a chance to be in the Super Bowl!

Function Description:

Write a MATLAB function that will determine who makes the playoffs given the matchup of a certain teams by comparing the two strings given as inputs. These are the ONLY inputs we will use to test your function:

```
'Patriots'
'Falcons'
'Giants'
'49ers'
'Jaguars'
```

Based off regular season statistics, as well as past match-ups between the 5 teams, compare the matchups using these rules:

"The Jaguars beat the 49ers, the 49ers best the Patriots, the Patriots comeback and beat the Falcons, the Falcons crush the Giants, the Giants smash the Jaguars, the Jaguars beat the Falcons, the Falcons demolish the 49ers, the 49ers triumph over the Giants, the Giants unfortunately beat the Patriots, and as they always have, the Patriots dominate the Jaguars."

(Diagram on next page for simplicity)

(continued…)

After determining who wins and makes it into the playoffs, output one of the following strings:

- If NFL Team 1 wins:
  - `'The <NFL Team 1> make it into the playoffs over the <NFL Team 2>!'`
- If NFL Team 2 wins:
  - `'The <NFL Team 2> make it into the playoffs over the <NFL Team 2>!'`
- If NFL Team 1 and NFL Team 2 are the same team:
  - `'Oh this isn't a matchup, it's practice!'`

Example:

```
playoffTeam = playoffCommittee('Patriots', 'Jaguars')
winner → 'The Patriots make it into the playoffs over the Jaguars!'
```

Hints:

- To make an apostrophe inside a string, use 2 apostrophes next to each other.

Function Name: `myOtRules`

Inputs:

1. (*char*) Team 1 Overtime Scores
2. (char) Team 2 Overtime Scores

Outputs:

1. (*char*) A statement describing who won

Topics: (*nested conditionals -or- prioritization*), (*string tokenization*)

Background:

The current NFL overtime rules have received a lot of scrutiny for being unfair or generally confusing. Thus you take it on yourself to make football's overtime more like fútbol's overtime rules, but a bit more streamlined.

Function Description:

The new overtime rules are as follows. Each team has a maximum of three chances to score a touchdown, these chances are called an overtime period (OT). The teams alternate attempts to score. If both teams or neither team scores in an OT, they must continue to the next OT. If only one team scores, that team wins the entire game and further OTs are ignored.

You are given two character vectors which represent the team and how they scored in overtime. They are in the format `'<Team Name>, <First OT>, <Second OT>, <Third OT>'`. Where Team Name is the name of the team and the OTs are `'Touchdown'` if the team scored and `'Nothing'` if the team didn't score.

Output a string representing who won the game in the format `'<Team Name> Won!!!'`. If there was a tie, output `'I guess the NFL was right all along...'`

Example:

```
team1scores = 'Chiefs, Touchdown, Nothing, Touchdown';
team2scores = '49ers, Touchdown, Nothing, Nothing';
winner = myOtRules(team1scores, team2scores);
winner → 'Chiefs Won!!!'
```