Let’s collaborate on a program that recommends a restaurant from a list of options. The program should use a few variables, like price range, delivery time, distance, and whether the restaurant is open in order to recommend customers the perfect spot.

The problem is that this recommendation program does not recommend much. While the program runs without any fatal JavaScript errors, it does not recommend any satiating suggestions. All we get is a disappointing 'We found 3 restaurants, the first is undefined.'.

Use your knowledge of TypeScript to fix type errors and add the missing features, so we can get customers on their way to good eats.

**1.**

Let’s take a look at the existing codebase! Currently, it consists of two files:

1. A **restaurants.ts** file, which has a list of all the possible restaurants.
2. An **index.ts** file, which contains the logic for recommending a restaurant to the customer.

Take a moment to look at the two files.

**2.**

Run the **index.ts** file with command tsc in the command line, and notice the errors. Scroll to the top of the list to see the first error:

Type 'number' is not assignable to type 'string'.

This error occurs because there’s a discrepancy between the value priceBracket is assigned and its type annotation. Update the type annotation so that it matches the value’s type.

**Note**: you can also use tsc index.ts and that will give you the same output.

**3.**

Save and run the code again with tsc. That error should be gone. But notice there’s a new error flagging a problem at the top of your output.

The > operator cannot act on two different types. The intent is to compare two numbers, but if you look at the **restaurants.ts** file, you’ll see that the priceBracket property has string values.

Since you defined priceBracket in **index.ts** as a number in the last step, use the [Number](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Number) constructor to transform the restaurant’s price bracket property to a number.

**4.**

Run tsc. The next error is similar to the one from the previous task. This time, assign the deliveryTimeMax variable to a value that will match a restaurant’s deliveryTimeMinutes property’s value in type.

**5.**

Run tsc. Yikes! Another similar error. Fix the type and save your code before moving on.

**6.**

Run tsc. There’s only one error left! It looks like we’re trying to access the restaurantName property on a restaurant but that property does not exist. Read the error to try to find a property to print the restaurant’s name here.

**7.**

Now when you run tsc, you’ll notice there are no remaining errors, however, this program is still inferring that the result variable is of type any. You can see this if you hover over the result variable with your mouse. Figure out what type this variable should be and annotate it. Check your work with tsc.

**8.**

When adding types with tsc, you’ve also compiled the code into JavaScript inside **index.js**.

The program should now run and provide a result. Run the program with node index.js and verify the program recommends something other than undefined.

**9.**

Now that the code has proper typings, let’s add a feature to make sure the restaurant is currently open based on the current time’s hour.

At the top of **index.ts**, declare a variable named hour with a value of new Date().getHours(). The getHours method will return the current hour in 24 hour UTC time. Then annotate this variable with the correct type.

**10.**

Within the filter function, we want to add another condition alongside the existing checks. This condition will check if the restaurant is currently open. If the restaurant is not currently open, it should return false (this will prevent the restaurant from appearing in the result).

In the restaurants list in **restaurants.ts**, each restaurant has an openHour and a closeHour property. Using an if statement, compare the hour from the last step against openHour and closeHour to make sure the restaurant is currently open.

**11.**

Run tsc again. You might notice there’s another type mismatch in the condition you just wrote. Fix the error and verify with tsc.

**12.**

When tsc no longer exposes any type errors, run node index.js to view the result of the restaurant recommendation. If you see no results, try changing the hour variable to another time of day.

Nice work on making this program more maintainable and free from a whole class of bugs caused by type mismatches.