⚠️ This document is publicly accessible ⚠️

Sureify does not need to handle a massive amount of data very quickly. Sureify does not need to handle massive amounts of traffic very quickly. Instead we focus on reducing defects and development costs. We strive to find errors early in the development process – ideally errors are discovered during compilation or testing.

In this exercise we're asking you to write TypeScript code that is easy to maintain. We are *not* looking for code that is efficient with time or space. Bonus points if you solve the problem without using classes. Bonus points for use of immutability.

The exercise

We're building an automated coffee vending machine. This machine might be found at a train station or a large office building. The user interface for this coffee vending machine presents the user with options. The user can change their options many times before they choose the purchase. Your job is to write a function that provides an updated price each time an option is selected. This function is deep inside the coffee vending machine, this function is not exposed as an HTTP API. All other parts of the coffee vending machine have already been built. The interface for this function has already been defined. You cannot change the signature of the function even though there may be a better way to design the function.

The following variables affect the price of the coffee: size and creamer.

The prices are:

Size
Small: $1.00
Medium: $1.50
Large: $2.00

Creamer
None: $0.00
Dairy: $0.25
Non-Dairy: $0.50

The interface is:

```
export type Category = 'size' | 'creamer';
export type Option = 'small' | 'medium' | 'large' | 'none' | 'dairy' |
'non-dairy';
export type Price = number;

export interface Pricer {
   /**
    * Invoked each time the user makes a selection.
    * No need to validate arguments,the caller validates the arguments
before this function is invoked.
    * @returns the _total_ price of the coffee so far given all the
selections made
    */
   (category: Category, option: Option): Price
}

/**
* A new pricer is created for each coffee being purchased.
*/
export const createPricer = (): Pricer => {
   // your code goes here
}
```

Here is a test that your code must pass:

```
it('provides the latest price given the options selected so far', ()
=> {
   // starting a coffee order
   const pricer = createPricer();

   // set the default options
   pricer('size', 'small');
   const defaultPrice = pricer('creamer', 'none');
   expect(defaultPrice).toBe(1.00);

   // user selects dairy creamer
   const priceAfterFirstSelection = pricer('creamer', 'dairy');
   expect(priceAfterFirstSelection).toBe(1.25);

   // user selects non-dairy creamer
   const priceAfterSecondSelection = pricer('creamer', 'non-dairy');
   expect(priceAfterSecondSelection).toBe(1.50);

   // user selects large
   const priceAfterThirdSelection = pricer('size', 'large');
```

```
    expect(priceAfterThirdSelection).toBe(2.50);
});
```