```
// Representa un ingrediente
public class Ingredient
{
	public string Name { get; init; }
	public double Cost { get; init; }

	public Ingredient(string name, double cost)
	{
		this.Name = name;
		this.Cost = cost;
	}
}

// Almacena los ingredientes existentes
public class IngredientHolder
{
	public static IngredientHolder Instance = new IngredientHolder();
	private IngredientHolder() {}

	private IList<Ingredient> _ingredients = new List<Ingredient>()
	{
		new Ingredient("coffee", 1.0),
		new Ingredient("milk",   0.5),
		new Ingredient("sugar",  0.2),
	};

	public IList<Ingredient> GetIngredients()
	{
		return new List<Ingredient>(this._ingredients);
	}
}

// Pregunta por todos los ingredientes
public interface IIngredientGetter
{
	public bool AskForOne(Ingredient ingredient);

	public IList<Ingredient> AskForAll(IEnumerable<Ingredient> ingredients)
	{
		IList<Ingredient> result = new List<Ingredient>();

		for (Ingredient ing : ingredients)
		{
```

```
                if (this.AskForOne(ing)) result.Add(ing);
            }

            return result;
        }
}

public class IngredientGetter : IIngredientGetter
{
        public static IngredientGetter Instance = new IngredientGetter();
        private IngredientGetter() {}

        public bool AskForOne(Ingredient ingredient)
        {
            Console.WriteLine($"Do you want {ingredient.Name}? (yes/no):");
            string input = Console.ReadLine();
            return input == "yes";
        }
}

// Prepara el café
public interface ICoffeeBuilder
{
        public void AddIngredient(Ingredient ingredient);
        public double GetCost();
}

public class CoffeeBuilder : ICoffeeBuilder
{
        public CoffeeBuilder() {}

        private double _cost { get; set; } = 0;

        public void AddIngredient(Ingredient ingredient)
        {
            this._cost += ingredient.Cost;
        }

        public double GetCost()
        {
            return this._cost;
        }
}
```

```
class Program
{
	static void Main()
	{
		IList<Ingredient> potentialIngredients =
IngredientHolder.Instance.GetIngredients();
		IList<Ingredient> ingredients =
IngredientGetter.Instance.AskForAll(potentialIngredients);

		ICoffeeBuilder coffeeBuilder = new CoffeeBuilder();
		for (Ingredient ing : ingredients) coffeeBuilder.AddIngredient(ing);

		double cost = coffeeBuilder.GetCost();
		Console.WriteLine("Total cost: $" + cost);
	}
}
```