ITMD:469/569 - Week 3 Lab 2

Travis Smith

Lab Summary

Lab 2 builds upon the foundation established in Lab 1 and introduces the use of interfaces to create functions that are type agnostic.

Program Specifications

Imagine you're a software developer at a large corporation. Your assignment involves crafting a program that calculates the company's annual expenses. The expenses originate from two sources, namely Employees and Facilities, each contributing for different reasons. As these sources may vary in the future, you need to create a calculation function flexible enough to accept any type of input that can generate its annual cost.

1. You already have code from Lab 1 that allows the creation of 'employee' types and computes their annual salary. Your current task is to design a type that signifies 'Facilities'.
2. Make use of an interface type named 'Costs'. This should simplify the calculation of costs to the company, regardless of their origin (be it Employees, Facilities, or any other type that might be introduced in the future).

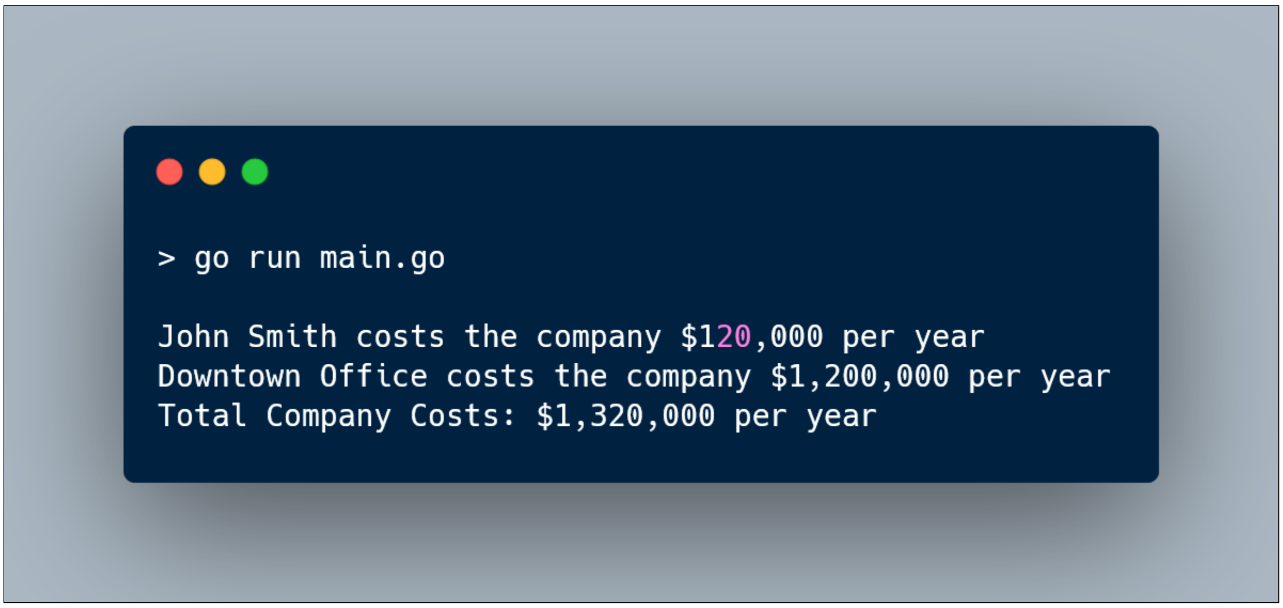
Helpful hint: The structure of this task closely resembles the in-class example about shapes. In that example, the area of different shapes could be computed using the same interface 'area' function, even though each shape had a distinct method for calculating area. It's crucial to understand the difference between a function and a method, as well as how interfaces are utilized.

Interfaces can be one of the more challenging concepts covered during the semester. If you're having difficulty understanding them, don't hesitate to reach out. Below are some useful resources on interfaces to help you out:

- GeeksForGeeks: Interfaces in Go
- Medium: Interfaces in Go
- CalliCoder: GoLang Interfaces
- YouTube Tutorial: Interfaces in Go

Steps

1. Starting with Lab 1, the first modification you need to make is to remove the printInfo() method.
2. Next, construct a new type named Facilities. This should include the following fields: Type of string, RentCost of float64, and MaintenanceCost of float64.
3. Define a method named calcYearlyCost on the Facilities type. This function should calculate the total yearly cost the facility incurs on the company and return this value. Note that this method has the same name as a method on the Employees type.
4. Now, create an interface type named Cost with a single field, a function named calcYearlyCost(). This function tells the interface, "Any type that wants to ’be’ a part of this interface, needs to have this method."
5. Then, implement the interface function. This function should take the Cost interface type, call its corresponding calcYearlyCost() method, and return the result. This approach lets us call the interface calcYearlyCost() function with ANY type that has a method named calcYearlyCost(). This concept is analogous to calling the measure() function with any type that has an area() method in the shapes example.
6. At this point, you should have two methods and an interface function, all named calcYearlyCost().
7. In your main() function, start by creating a single Employee type and fill in its fields.
8. Calculate the cost incurred by the company for this employee. Use the interface function calcYearlyCost(), not by directly calling the calcYearlyCost() method of the Employee type.
9. Print the name of the employee and the cost they bring to the company.
10. Then, create a single Facilities type and fill in its fields.
11. Calculate the cost this facility incurs on the company using the calcYearlyCost() interface function.
12. Print the type of the facility and the cost it incurs on the company.
13. Lastly, print the total cost to the company, combining all employee and facility costs.



```
> go run main.go

John Smith costs the company $120,000 per year
Downtown Office costs the company $1,200,000 per year
Total Company Costs: $1,320,000 per year
```

Submission

Please compress your source code into a ZIP file and upload it to Blackboard. Additionally, attach a separate screenshot demonstrating your code running successfully. week