C++ Class and Struct Problems

Inheritance and Override

Problem 1: Animal Base Class (Difficulty: 2)

Create a base class `Animal` with a method `makeSound()` that prints a generic sound.

Create classes `Dog` and `Cat` that inherit from `Animal` and override `makeSound()`.

Problem 2: Employee and Manager (Difficulty: 3)

Create a class `Employee` with name and ID and a `printInfo()` method.

Create `Manager` (inherits from `Employee`) with a `teamSize` member.

Override `printInfo()` to also show team size.

Pointers and Inheritance

Problem 3: Polymorphic Access (Difficulty: 3)

Using the `Animal` hierarchy, write a function `void makeItSpeak(Animal* a)` that calls `makeSound()`.

Show that `Dog*` and `Cat*` still call their own versions.

Problem 4: Base Pointer, Derived Object (Difficulty: 4)

Create base class `Shape` with virtual `area()`.

Create `Rectangle` and `Circle` that override it.

Use a `Shape*` pointer to call `area()` on derived objects.

Pure Virtual and Abstract Classes

Problem 5: Abstract Appliance (Difficulty: 3)

Create abstract base class `Appliance` with pure virtual `run()`.

Create `WashingMachine` and `Microwave` that override `run()` with their own messages.

Problem 6: Shape Interface (Difficulty: 4)

Create abstract class `Shape` with pure virtual `area()` and `perimeter()`.

Implement `Circle` and `Rectangle` with these methods.

Structs

Problem 7: Point Struct (Difficulty: 2)

Create a `struct Point` with `x` and `y`.

Write a function that takes a `Point` by reference and prints it.

Problem 8: AddressBookEntry (Difficulty: 3)

Create a `struct AddressBookEntry` with name, email, and phone.

Use a `std::vector<AddressBookEntry>`.

Add entries, search by name, and print them all.