Abstract Class Problems in C++

Problem 1: Appliance Power Consumption

Problem Statement:
Create an abstract class `Appliance` with a pure virtual method `powerConsumption()`. Derive `Fan` and `AC` classes that override `powerConsumption()` with different power consumption values.

Sample Input:

```
Appliance *a = new Fan();
a->powerConsumption();
```

Sample Output:

```
Fan consumes 50W.
```

Problem 2: Employee Salary Calculation

Problem Statement:
Create an abstract class `Employee` with a pure virtual method `calculateSalary()`. Derive `Manager` and `Developer` classes to compute salaries based on different formulas.

Sample Input:

```
Employee *e = new Manager(5000);
e->calculateSalary();
```

Sample Output:

```
Manager's salary: 7000
```

Problem 3: Geometric Shapes

Problem Statement:
Create an abstract class `GeometricShape` with a pure virtual method `perimeter()`. Derive `Triangle` and `Rectangle` classes that override `perimeter()`.

Sample Input:

```
GeometricShape *g = new Triangle(3, 4, 5);
g->perimeter();
```

Sample Output:

```
Perimeter of triangle: 12
```

Problem 4: Sorting Algorithm

Problem Statement:
Create an abstract class `Sorter` with a pure virtual method `sort()`. Derive `BubbleSort` and `QuickSort` classes that override `sort()`.

Sample Input:

```
Sorter *s = new QuickSort({3, 1, 4, 1, 5});
s->sort();
```

Sample Output:

```
Sorted array: 1, 1, 3, 4, 5
```

Problem 5: Bank Interest Calculation

Problem Statement:
Create an abstract class `BankAccount` with a pure virtual method `calculateInterest()`. Derive `SavingsAccount` and `FixedDeposit` classes that override `calculateInterest()`.

Sample Input:

```
BankAccount *b = new SavingsAccount(1000, 5);
b->calculateInterest();
```

Sample Output:

```
Interest earned: 50
---
```