۱۵.

Interface زیر را در نظر بگیرید. این واسط به عنوان تایپ توابع ریاضی با یک متغیر (x) تعریف شده است. متد evaluate تابع را به ازای مقدار پارامتر ورودی ارزیابی می‌کند و متد derivative یک شی باز می‌گرداند که معادل مشتق تابع اولیه است.

```
public interface MathFunc {
	public double evaluate();
	public MathFunc derivative();
}
```

الف) کلاسی به نام Polynomial تعریف کنید که واسط فوق را پیاده‌سازی کند و بتوان با استفاده از آن یک چندجمله‌ای به شکل $ax^n + bx^{n-1} + cx^{n-2} + \cdots + dx^1 + e$ تعریف کرد. این کلاس یک سازنده دارد که پارامتر ورودی آن یک آرایه حاوی ضرایب چندجمله‌ای هستند به نحوی که ضریب x^i در ایندکس i آرایه ذخیره شده است. نمونه‌ای از استفاده از این کلاس به شکل زیر است (معادل چند جمله‌ای $4x^3 + 5x^2 + 8$).

```
double[] c = {4, 5, 0, 8};
Polynomial p = new Polynomial(c);
System.out.println(p.evaluate(3)); // x را برابر با ۳ قرار می‌دهد و مقدار چند جمله‌ای را محاسبه می‌کند.
System.out.println(p.derivative().evaluate(3));
```

(برای محاسبه a^b می‌توانید از Math.pow(a, b) استفاده کنید.)

ب) کلاسی به نام AddFunc تعریف کنید که واسط MathFunc را پیاده‌سازی کند برای جمع دو تابع مورد استفاده قرار گیرد. این کلاس یک سازنده دارد که پارامترهای آن دو تابع دیگر هستند. به عنوان نمونه با فرض داشتن قطعه کد قسمت الف می‌توان قطعه کد زیر را مثال زد:

```
AddFunc a = new AddFunc(p, p.derivative());
System.out.println(a.evaluate(3));
System.out.println(p.derivative().evaluate(3));
```

(توجه داشته باشید که در تعریف AddFunc پارامترهای سازنده از نوع هر کلاسی از تایپ MathFunc می‌تواند باشد و نه فقط از نوع Polynomial).