Task 0

Goto page 351-356 copy the code times1.cpp and times2.cpp. Run the codes in your IDE and understand how the cast operator works. You need to explain later.

Task 1

Implement a `BookCollection` class to manage a collection of books. It will have the following member variables:
1. `titles`: An array that can store up to 5 book titles.
2. `authors`: An array that can store up to 5 book authors.
3. `prices`: An array that can store up to 5 book prices.
4. `numBooks`: An integer variable that keeps track of the number of books currently in the collection.

The member functions:

1. `BookCollection()`: The constructor initializes an empty collection with zero books.

2. `operator[](std::string& title)`: To retrieve the price of a book by providing its title as an argument. If the book title is found in the collection, it returns the corresponding price. If the title is not found, it returns a default price (0.0).

3. `operator[](std::string& title)`: To update the price of a book by providing its title and the new price as arguments. If the book title is found in the collection, it updates the price. If the title is not found, it adds a new book with the provided title and price to the collection.

4. `addBook(std::string title, std.::string author, double price)`: To manually add a new book to the collection by providing its title, author, and price as arguments. If the collection is not already full (contains fewer than 5 books), it adds the new book to the collection. If the collection is already full, it displays an error message indicating that the collection is at its maximum capacity.

Implement the `BookCollection` class in a way so that it can manage a collection of books, retrieve book prices, update prices, and add new books to the collection while ensuring it doesn't exceed its capacity.

Task 2

Implement a `CurrencyConverter` class to help you convert between two different currencies, such as converting from US Dollars (USD) to Euros (EUR) or vice versa.

It will have the following member variables:
1. `exchangeRate`: A double that represents the conversion rate between the two currencies. For example, if 1 USD is equal to 0.85 EUR, `exchangeRate` would be 0.85.
2. `amountInUSD`: A double that stores the amount of money in US Dollars.
3. `amountInEUR`: A double that stores the amount of money in Euros.

The member functions:
1. `CurrencyConverter()`: The constructor initializes the object with default values, where the exchange rate is 1.0 (no conversion), and both `amountInUSD` and `amountInEUR` are set to 0.0.

2. `CurrencyConverter(double rate, double usdAmount)`: This constructor takes the exchange rate and an amount in US Dollars as arguments. It then calculates and stores the equivalent amount in Euros based on the provided exchange rate.

3. `CurrencyConverter(double rate, double eurAmount, bool isEUR)`: This constructor takes the exchange rate, an amount in Euros (or US Dollars if `isEUR` is false) as arguments. It calculates and stores the equivalent amount in the other currency based on the provided exchange rate.

4. `showCurrency()`: This function displays both the amount in US Dollars and the equivalent amount in Euros in a user-readable format.

5. `operator double()`: This conversion operator allows you to convert a `CurrencyConverter` object to a double, which represents the amount in US Dollars.

In the `main()` function, we demonstrate the `CurrencyConverter` class by creating two objects, converting between USD and EUR, and displaying the amounts in both currencies.

Task 3

You will implement a Currency Conversion calculator.
Three classes are defined:
CurrencyBDT represents an amount in BDT and poisha. It has two private member variables: taka and poisha.
CurrencyDollar represents an amount in US Dollars and cents. It has two private member variables: dollars and cents.
CurrencyEuro represents an amount in Euros and cents. It also has two private member variables: euros and cents.

The member functions:
The classes have constructors to initialize the currency amounts. The default constructor sets the amounts to zero, and the parameterized constructor is used to specify the amount in taka/dollars/euros and poisha/cents.
There are two conversion functions: The first one is used to convert an amount in USD to an equivalent amount in BDT. It takes into account the conversion rate (1 BDT = 0.0091 USD) and performs the necessary calculations to convert the amount (11 dollar 44 cents= 1254 taka 72 poisha). The second conversion is used for Euro to BDT(1 BDT = 0.0085 Euro)

In the **main()** function, show both conversions.