Tutorial Week 5: Assorted Chocolate Truffles

Chocolate boxes can be of different width and height. We will create chocolate boxes with an assortment of chocolates. By default, all chocolates in the box are hazelnut crunch. We will add assorted chocolate truffles in chocolate boxes: butterscotch walnut, salted caramel, creamsicles, milk white, crusty toasted peanut, and so on.

Figure 1: A chocolate box with an assortment of chocolates.

You need to complete the following classes.

1. Chocolate
2. ChocolateBox

Chocolate Class:

```
class Chocolate{
	std::string type;

public:
	Chocolate (std::string choco);
	Chocolate() :Chocolate("Hazelnut Crunch") {}
	Chocolate(const Chocolate&);
	Chocolate& operator=(const Chocolate&);
	Chocolate& setType(std::string cType);
	std::string getType() const;

	friend bool operator==(const Chocolate& chocoOne, const Chocolate& chocoTwo);
	friend Chocolate operator+(const Chocolate& chocoOne, const Chocolate& chocoTwo);
	friend bool operator!(const Chocolate& choc);
	friend std::istream& operator>>(std::istream&, Chocolate&);   //non-member friend
function
	friend std::ostream& operator<<(std::ostream&, const Chocolate&);   //non-member
friend function
};
```

Chocolate Box Class:

```
class ChocolateBox{

	friend bool operator==(const ChocolateBox&, const ChocolateBox&);
	friend bool operator!(const ChocolateBox&);
	friend ChocolateBox operator+(const ChocolateBox&, const ChocolateBox&);
	friend std::ostream& operator<<(std::ostream&, const ChocolateBox&);
	friend std::istream& operator>>(std::istream&, ChocolateBox&);
	friend bool isEmpty(const ChocolateBox&);

public:
	ChocolateBox(int w, int h, Chocolate c);		// delegating constructor
	ChocolateBox():ChocolateBox(2, 2, Chocolate()) {};	// default constructor calls
delegating constructor
	ChocolateBox(const ChocolateBox&);			// copy constructor
	ChocolateBox(ChocolateBox&&);				// move constructor
	ChocolateBox& operator=(const ChocolateBox&);		// copy assignment operator
	ChocolateBox& operator=(ChocolateBox&&);		// move assignment operator

	~ChocolateBox();					//destructor

private:
	int width;
	int height;
	Chocolate** choco;



};
```

Compile and execute your code with g++ or the Microsoft compiler.

Complete **Chocolate** and **ChocolateBox** classes.

- **Chocolate.h, Chocolate.cpp, ChocolateBox.h**, and **ChocolateBox.cpp**.

Once you complete both **Chocolate** and **ChocolateBox** classes, you can test with the attached test file "**test_chocolate_box.cpp**". You will get the following output:

```
syasmin@CSCD110497WP:~/CSCD305/ChocolateBox$ ./ChocolateBox
Chocolates in cBox=
[0][0]:Hazelnut Crunch
[0][1]:Hazelnut Crunch
[1][0]:Hazelnut Crunch
[1][1]:Hazelnut Crunch

Now, you can fill Box One with your favorite truffles:
[0][0]:Butterscoch Walnut
[0][1]:Salted Caramel
[1][0]:Cruncy Roasted Nut
[1][1]:Creamsicle

Chocolates in cBoxOne=
[0][0]:Butterscoch Walnut
[0][1]:Salted Caramel
[1][0]:Cruncy Roasted Nut
[1][1]:Creamsicle

Chocolates in cBoxTwo=
[0][0]:Milk White
[0][1]:Milk White
[1][0]:Milk White
[1][1]:Milk White

Chocolates in cBoxThree =
[0][0]:Butterscoch Walnut Milk White
[0][1]:Salted Caramel Milk White
[1][0]:Cruncy Roasted Nut Milk White
[1][1]:Creamsicle Milk White

 "1"  will be printed if chocolates in cBoxOne and cBoxTwo are equal
0
It is not "Butterscotch Walnut"!
Chocolates in cBoxFour=
[0][0]:Milk White
[0][1]:Milk White
[1][0]:Milk White
[1][1]:Milk White

Chocolates in cBoxFive=
[0][0]:Butterscoch Walnut
[0][1]:Salted Caramel
[1][0]:Cruncy Roasted Nut
[1][1]:Creamsicle

Chocolates in cBoxOne=
The chocolate box is empty!

Chocolates in cBoxTwo=
The chocolate box is empty!

syasmin@CSCD110497WP:~/CSCD305/ChocolateBox$
```