Basic Exercise Using Objects and Object Pointers in C++

In this exercise start by copying the three files from your last assignment, then change main to use object pointer variables in place of the object you were given for the previous assignment. Again, there is no design challenge, just a little practice with object pointers.

Using the same procedure as always start a new C++ project called <you>BookPointers, and create the cpp file with the same name as your project (or main.cpp if you insist on using Code Blocks). Also create Book.h and Book.cpp, as in the last assignment. Copy the contents of all 3 files from your previous assignment. This time However in your youBookPointers.cpp file, create two functions: `void main1()`, and `void main2()`. Move the code from your original assignment 4 main to the `main1()` function. The code in `main2()` should be the same as main1() with one major difference. The `textbook`, `reference`, and `alternateRef` variables in `main2()` must all be object pointer variables instead of object variables. Then adjust the code in `main2()` with all changes needed for the same behavior to work with the three pointer variables.

Java does not have "explicit" pointer types, or pointer operations. But Java uses pointers all the time. All object variables in Java are pointers. That's why you need the "new" operation to initialize objects in Java. It's just that other than "new", there is no special syntax. Object variables in C++ are not pointers. To do what Java does, C++ uses explicit pointers with their own syntax for many operations. To declare a pointer variable in C++, you append a * to the end of the type name. While `int iVal` is an integer, `int* addrVal` is a pointer to an integer. Similarly, `City dest`; is an object of the City class, while `City* ptrDest`; is a pointer to an object of the City class. (In Hungarian notation, p, ptr, addr, and hndl were all prefixes used to indicate a pointer variable.)

Note that in C++, it is also possible to indicate a pointer variable by prepending a * to the variable name (`int *addrVal;`). I discourage using that notation as it represents a context-dependent syntax that can lead to ambiguity. Unfortunately it is not so uncommon, so you will see examples using that form in textbooks and online discussions.

For this assignment, you need to know that, like Java, pointer variables when first declared, o not point to any object of that type. To get an actual object you need to use the "new" keyword to invoke the constructor, which returns an object of that type. In the above example declaration, to initialize the pointer so it points at an actual object, you would write:

```
City* ptrDest = new City();.
```

As in Java, if there is a constructor that takes the name of a city as an argument, the declaration would look like:

```
City* ptrDest = new City("Detroit");
```

Unlike Java, to access the members of an object using a pointer variable, you have to use something other that a dot. Instead you have to use the two letter "dereference" operator "->".

```
City dest;
dest.setName("Detroit");

City* pDest = new City();
pDest->setName("Cleveland");
```

For the next assignment, there are two more pointer operators you will need to know. These two operators are used to switch back and forth between variables and pointers to variables. If you have a variable called dest, the &dest is its address. If you have a pointer variable called ptrDest, then *ptrDest is the value at that address.

```
ptrDest = &dest;
```

For this assignment you can just turn in the .cpp file with main1() and main2().