Assignment 5 Intro to C++
Value: 7pts
Due at the beginning of the next session.

This assignment is to write a class that exercises operator overloading in C++. Hand in the files you write, and console output to demonstrate correct operation of your program.

Write a class named "Pennies", internally it has a protected integer that keeps track of the number of pennies representing its current value.

Define four constructors: a default constructor, a copy constructor, a constructor that takes an integer to initialize the value of pennies, and a constructor that takes a double representing the value in <u>dollars</u>.

Overload the type conversion operators for int and double to return the value in pennies and dollars respectively. So in main.cpp if an instance of Pennies is created as: "Pennies P(1000);" then "int n = P;" will result in setting n to 1000.

Overload addition and subtraction for both integers and doubles. Adjust the number of pennies appropriately for the values provided by the "user".

Overload "ostream" to handle output to the screen. Pennies should print in correct monetary notation: $0.01, $1.00, or $1.01 for instance for integer penny values of 1, 100, and 101.

Test your class in main. A partial sample of your testing should be:

```
Pennies P = 1000; // Pennies will construct at $10.00

Pennies Bucks = 2.048;  // bucks will construct at $2.05

P = P + 2;  // P should now be 1002 or $10.02

Bucks = P - 1.99; // bucks = $8.03 pennies is unchanged

int Cents = Bucks; // Cents should be 803

double Value = Bucks; // Value should be 8.03

cout << Cents << endl;  // should print 803

cout << Value << endl;  // should print 8.03

cout << P << endl;      // should print $10.02

cout << Bucks << endl;  // should print $8.03
```

Background:

While its generally not a good idea to have a class return a very different value when used as an integer from its value as a double. In this case it was chosen as a way to distinguish between whether a value is represented dollars or cents, or the user needed to get the value as dollars or cents.

A better idea is to define Pennies which only handles integer conversions and arithmetic, then derive a class Dollars from Pennies which only handles conversions and arithmetic with doubles. This takes a little more effort but the program design is much cleaner.

Either solution will be acceptable for the assignment. Alternate test:

```
Pennies P = 1000; // Pennies will construct at $10.00

Dollars Bucks = 2.048;  // bucks will construct at $2.05

P = P + 2;  // P should now be 1002 or $10.02

Bucks = P - 1.99; // bucks = $8.03 pennies is unchanged

int Cents = Bucks; // Cents should be 803

double Value = Bucks; // Value should be 8.03

cout << Cents << endl;  // should print 8.03

cout << Value << endl;  // should print 2.05

cout << P << endl;      // should print $10.02

cout << Bucks << endl;  // should print $8.03
```