

# Lab 3 – Adding Functionality

CPRG4202 – T. MacDonald

---

In Lab 2, you created a new class called WorkTicket. In this lab you will add the following functionality to the class.

## New Requirements for WorkTicket Class:

- A copy constructor that initializes a new WorkTicket object based on an existing WorkTicket object. For testing purposes, include the statement:  

~~cout << "\nA WorkTicket object was COPIED.\n";~~
- A conversion operator that converts a WorkTicket object to a string in the following format: Work Ticket # Number – Client ID (Date)–Description; e.g.:  

~~"Work Ticket # 2 – ABC123 (10/3/2012): User cannot locate 'any' key"~~
- Overload the equality ('==' operator to compare a WorkTicket object to another WorkTicket object using a member-wise comparison. Return true if all the attributes of both objects are the same; false if they are not all the same.
- Overload the assignment ('=' operator to assign all of the attributes of one WorkTicket object to another (member-wise assignment). For testing purposes, include the statement:  

~~cout << "\nA WorkTicket object was ASSIGNED.\n";~~
- Overload the '>>' operator relative to the class to allow the user to enter all of the attributes of a WorkTicket object from the console or any istream. **Include validation.**
- Overload the '<<' operator relative to the class to displays all the object's attributes neatly on the console or to any ostream. This will duplicate the functionality of the ShowWorkTicket() method however keep the original method intact for legacy reasons.

## Program Requirements:

The purpose of the main() function in this program is to demonstrate each of the added features of the WorkTicket class.

- Declare a new object based on an existing object.
- Typecast an object as a string and output it to the console.
- Have the user input an object's attributes from the console using cin.
- Test if two objects are equal.
- Assign an existing object to another existing object.
- Output an object to the console using cout.

## Things to Explore:

You are welcome to explore beyond the mandatory requirements if you wish. One suggestion you may be interested in is implementing the tmDate Class we discussed in-class to represent the work ticket date.

## General Requirements

- Include an opening comment with your full name, the full names on the student(s) you are working with, the name of the program, the date, and a short description.
- Follow the style guide! Use descriptive names and sensible data-types for variables, constants, arrays, functions, etc. that follow our naming conventions. Use good spacing and make sure braces ({} ) are located where they are supposed to be.
- Attach the unzipped source code file(s) (.cpp, .h) to the dropbox.