COMP 1409 – Summary of Session 1 Concepts

- Class: an abstraction that defines a category of entity, e.g. Car. A class defines fields (data; attributes) and methods (behaviors; functions).
- Object: an instantiation of a particular class, e.g. myCar, yourCar, thisCar, thatCar. Each object holds its own data in its fields. All cars may have a color (the Car class may define such a field), but each actual car can have its own value for that color: some blue, or red, or even yellow! A yellow car! Ha ha ha!
- Field: also known as instance variable. This is where the data is actually stored for an object, in its instance variables. These are variables declared for the **class**, not inside a method. For example:

```
class Car
{
    private String colour;          // a field
}
```

- The methods defined for the class have access to the fields and can change their contents. Each object created from a class will have its own copy of each of these fields (just like you have your own copy of the "eye colour" field whose value may be different than my copy; my value is blue, yours may be brown or something different than blue). This is where the data **for the object** is stored.
- Variable: a named memory location, used to hold a value of a particular data type.
- Data type: the **kind** of data used, to specify how much memory to allocate. Some common data types are int (whole number, e.g. 5, 999, -34), char (single character inside single quotation marks, e.g. 'h', '9', '&'), String (group of characters inside double quotation marks, e.g. "abc", "1234", "hello there"), double (number with a decimal component, e.g. 8.98, -7654.9099), boolean (true, false…without quotation marks). Different data types use different sized memory locations.
- Default values: Java automatically gives initial values to the fields in a class. The default value depends on the type of data: int 0, double 0.0, boolean false, char null char, String null.
- State: the set of values of all attributes that define an object. The data values in the fields of a particular object.
- Method: a group of programming statements (java instructions) that is given a name. Methods are defined for a class. The objects of that class use the methods to access the data in their fields. (In non-OO languages methods are called subroutines, procedures, or functions.)
- Parameter: a value that is passed to a method inside parentheses when the method is invoked, e.g. myCar.drive(30). The value 30 is the parameter passed to the drive method belonging to the myCar object; another example: bankAccount.withdraw(20.0)
- Source code: the instructions that define the attributes and behaviors of a class. These instructions are written in plain text in a programming language, e.g. Java.

- Compile: an operation to change Java instructions into machine code the computer can understand.
- Compiler: a program that converts source code into machine code.
- Return value: a value output by a method. This might be the contents of one of the fields, or it might be the result of some calculation or other operation. The type of the return value is part of the method signature. Example:

  ```
  public int getCurrentYear( )
  {
      return 2011;
  }
  ```

- Method signature: also known as the method header. Information about the method, e.g. public void changeSize(int newHeight, int newWidth). This specifies a method that (1) is **public**ly accessible outside the class, (2) does not return a value, (3) is named "changeSize", (4) requires two parameters of type int to be passed to it when it is invoked as follows: changeSize(40, 50); // 40 is newHeight; 50 is newWidth
- void: part of a method header to specify that the method does not return a value. If the method does return a value, the data type of the return occupies the same position in the method signature, e.g. public boolean isVisible(). This method returns either true or false when it is invoked.

```
public class BankAccount                        // uppercase letters for class names
{
        private String      firstName;          // all class data is private
        private double      balanceDollars;
        private boolean     hasOverdraft;
        private int         overdraftAmount;

        public String getFirstName()            // methods are often public
        {
                return firstName;
        }

        public void setFirstName(String newName)
        {
                firstName = newName;
        }
}
```