COMP 1409 – Summary of Session 2 Concepts

- Parts of a class. An outer wrapping names the class, and a larger inner part contains the fields, constructors and methods of the class. Fields store data for the class. Constructors set up the object created from the class by initializing the data members (fields). Methods implement the behaviours of the object that is created.
- Access modifiers, private and public. Fields are defined as private to ensure that only the methods belonging to the class will have access to the data. The class itself and its methods are defined as public so that they may be referred to from outside. (A method could be defined as private and referred to only from another method of that class.)
- Constructors have the same name as the class. A constructor may or may not have one or more parameters passed to it. Java supplies a default constructor that initializes all the fields to default values. If a non-default constructor is provided there is no more default constructor. Therefore we provide a default constructor. The purpose of a constructor is to put the object into a valid initial state. It is the constructor that is called when you click on **`new`** in BlueJ to create an object.
- Comments are descriptions in English, meant to make code easier to understand. The class itself should be commented, as well as each field, each constructor and each method. Comments are also used within the methods to explain the logic. Three types of comments in Java:

```
/*
   this is a comment spanning several lines
*/

// this is a comment to the end of the line

/**
   this is a Javadoc comment - two ** at beginning
*/
```

- Parameter passing, sometimes called argument passing. The signature of a method or a constructor specifies the data type to be passed in. This is known as the formal parameter. The actual value that is passed when the method or constructor is called is known as the actual parameter, or the argument. The formal parameter is a variable (a named place in memory) that exists only while the method is being executed. The actual parameter might be a literal value, or a variable, or the return value from a method call.
- Variable declaration tells the compiler to reserve a chunk of memory of a specific size. The variable declaration must include the data type and the name of the variable, e.g. `private int age;`
- Assignment statement is the statement that changes the value in a variable. The assignment operator (=) takes the value in the expression on the right and puts that value into the variable on the left, e.g. `counter = 0;`

- Accessor methods, also called “getters”. An accessor is a method that allows us to read from a field in the object. It provides information about the object by returning the contents of a field (instance variable). Accessor methods start with the word “get”with the name of the method corresponding with the name of the relevant field.

```
/** Example class for comp 1409 */
public class Tree
{ // begin class definition

      private String kindOfTree; // declare fields
      private int ageOfTree;

      /** Tree constructor  // javadoc comment
        @param treeKind kind of tree
       @param treeAge age of tree
        */                   // end comment
      public Tree(String treeKind, int treeAge)
      {
            kindOfTree = treeKind; // initialize fields
            ageOfTree = treeAge;
      }


      /** Default constructor */
      public Tree()
      {}  // fields are set to default values

      /**
        Gets tree kind
        @return the tree kind as a String
      */
      public String getKindOfTree()
      {
          return kindOfTree; // return the field contents
      }

     /**
        Gets tree age
        @return the age of tree as an int
      */
      public int getAgeOfTree()
      {
          return ageOfTree; // return the field contents
      }
} // end class definition
```