**Concepts and Keywords**

**CH1**

* **Compiler** - Software that translates the Java source code into the Java class file which can be run on the computer.
* **Compiler or syntax error** - An error that is found during the compilation.
* **Main method** - Where execution starts in a Java program.
* **Variable** - A name associated with a memory location in the computer.
* **Declare a Variable** - Specifying the type and name for a variable. This sets aside memory for a variable of that type and associates the name with that memory location.
* **Initializing a Variable** - The first time you set the value of a variable.
* **data type** - determines the size of memory reserved for a variable, for example int, double, boolean, String.
* **integer** - a whole number like 2 or -3
* **boolean** - An expression that is either true or false.
* **Camel case** - One way to create a variable name by appending several words together and uppercasing the first letter of each word after the first word (myScore).
* **Casting a Variable** - Changing the type of a variable using (type) name.
* **Operator** - Common mathematical symbols such as + for addition and \* for multiplication.
* **Compound assignment or shortcut operators** - Operators like x++ which means x = x + 1 or x \*=y which means x = x \* y.
* **modulo** - The % operator which returns the remainder from one number divide by another.
* **arithmetic expression** - a sequence of operands and operators that describe a calculation to be performed, for example 3\*(2 + x)
* **operator precedence** - some operators are done before others, for example \*, /, % have precedence over + and -, unless parentheses are used.
* **boolean** - used to declare a variable that can only have the value true or false.
* **double** - used to declare a variable of type double (a decimal number like 3.25).
* **int** - used to declare a variable of type integer (a whole number like -3 or 235).
* **String** - used to declare a variable of type String which is a sequence of characters or text.
* **System.out.print()** - used to print output to the user
* **System.out.println()** - used to print output followed by a newline to the user
* **=** - used for assignment to a variable
* **+, -, \*, /, %** - aritmetic operators

**CH2**

* **class** - defines a new data type. It is the formal implementation, or blueprint, of the attributes and behaviors of the objects of that class.
* **object** - a specific instance of a class with defined attributes. Objects are declared as variables of a class type.
* **constructors** - code that is used to create new objects and initialize the object’s attributes.
* **new** - keyword used to create objects with a call to one of the class’s constructors.
* **instance variables** - define the attributes for objects.
* **methods** - define the behaviors or functions for objects.
* **dot (.) operator** - used to access an object’s methods.
* **parameters (arguments)** - the values or data passed to an object’s method inside the parentheses in the method call to help the method do its job.
* **return values** - values returned by methods to the calling method.
* **immutable** - String methods do not change the String object. Any method that seems to change a string actually creates a new string.
* **wrapper classes** - classes that create objects from primitive types, for example the Integer class and Double class.
* **new** is used to create a new object.
* **null** is used to indicate that an object reference doesn’t refer to any object yet.
* The following String methods and constructors, including what they do and when they are used, are part of the Java Quick Reference in the AP exam:
  + **String(String str)** : Constructs a new String object that represents the same sequence of characters as str.
  + **int length()** : returns the number of characters in a String object.
  + **String substring(int from, int to)** : returns the substring beginning at index from and ending at index (to -1). The single element substring at position index can be created by calling substring(index, index + 1).
  + **String substring(int from)** : returns substring(from, length()).
  + **int indexOf(String str)** : returns the index of the first occurrence of str; returns -1 if not found.
  + **boolean equals(String other)** : returns true if this (the calling object) is equal to other; returns false otherwise.
  + **int compareTo(String other)** : returns a value < 0 if this is less than other; returns zero if this is equal to other; returns a value > 0 if this is greater than other.
* The following Integer methods and constructors, including what they do and when they are used, are part of the Java Quick Reference.
  + Integer(value): Constructs a new Integer object that represents the specified int value.
  + Integer.MIN\_VALUE : The minimum value represented by an int or Integer.
  + Integer.MAX\_VALUE : The maximum value represented by an int or Integer.
  + int intValue() : Returns the value of this Integer as an int.
* The following Double methods and constructors, including what they do and when they are used, are part of the Java Quick Reference Guide given during the exam:
  + Double(double value) : Constructs a new Double object that represents the specified double value.
  + double doubleValue() : Returns the value of this Double as a double.
* The following static Math methods are part of the Java Quick Reference:
  + **int abs(int)** : Returns the absolute value of an int value (which means no negatives).
  + **double abs(double)** : Returns the absolute value of a double value.
  + **double pow(double, double)** : Returns the value of the first parameter raised to the power of the second parameter.
  + **double sqrt(double)** : Returns the positive square root of a double value.
  + **double random()** : Returns a double value greater than or equal to 0.0 and less than 1.0 (not including 1.0!).
* **(int)(Math.random()\*range) + min** moves the random number into a range starting from a minimum number. The range is the **(max number - min number + 1)**. For example, to get a number in the range of 5 to 10, use the range 10-5+1 = 6 and the min number 5: (int)(Math.random()\*6) + 5).