WORKSHEET 01B – In class week 1 day 1

Do the following and answers the questions to hand in. Remember to label the answers correctly: 1, 2, etc.

1	***Have BlueJ window and File Explorer window side by side (or overlapping a little) so you can see files come and go.*** **Create a BlueJ project**: • Decide/name the folder in your disk that you are going to keep all your work – we will refer to it as **YourFolder** here • Start BlueJ • Do `Project > New Project` • Type the project name: `Proj01` and keep in **YourFolder** **Questions**: a) In BlueJ, what is inside the project window? Draw a picture of this file. What is its name? (*Hint: If you hover the mouse over it or right-click, you will see the name.*) b) What is the purpose of the file in question (a)? c) In your disk, what are in the `Proj01` folder? Write **full file names** (i.e. including the extensions). d) Quit BlueJ, then click on each file in the `Proj01` folder in your disk. What happens when you click on each file? ***Hint***: *To find a file extension, right-click on the icon in your disk then choose "Properties".* ***Note***: ***YourFolder*** *should contain* ***Proj01***.		a) In BlueJ, the image looks like a sheet of binder paper. Its name is "README". b) The purpose of the "README" file is for the programmer to write a summary about this Java project. c) In the disk, there is a folder called "Proj01" (i.e., with same name as your Java project), which contains 2 items: • The BlueJ icon, with the name "package.bluej", representing the project • The spiral notebook paper icon, with the name "README.TXT", which is the same as in (a) d) Clicking name "package.bluej" will launch BlueJ & open the project. Clicking "README.TXT" shows the text contents (in one of the applications that can open a text file, such as **WordPad**, MS Word, NotePad etc.)
2	**Prepare file:** • In your disk (possibly in **YourFolder**), create a folder called **StudentsFiles**, where you will keep the files downloaded for the course. • Download the `HelloPrinter.java` file from your teacher & keep in **StudentsFiles** ***Note***: ***YourFolder*** *should contain* ***StudentsFiles*** *(and* ***Proj01****).* **Question**: • Do you have the `HelloPrinter.java` file in your **StudentsFiles?**		• Yes

3	**Create a class in a BlueJ project**, in `Proj01`: • Do `Edit > Add a Class from File…` • Choose `HelloPrinter` from **StudentsFiles** folder. **Questions**: a) In BlueJ, what was added to the project window? b) Do you see the stripes over the item in (a)? c) In the `Proj01` folder in your disk, what was just added? Use full names (i.e. including the extensions) for files. What kind of file (i.e., of Java code) is it? d) In **StudentsFiles** folder, is the “HelloPrinter.java” file still there?		a) In BlueJ, a brown tile named “HelloPrinter” was added. b) Yes. c) In `Proj01` folder, there is a file named “HelloPrinter.java”. It is a **source file**. d) Yes, the “HelloPrinter.java” file is still in “StudentsFiles”. This means that “Add a Class from File…” makes a copy of the original file for the Java project.
4	**Compile a class in a BlueJ project** (in `Proj01`), by doing one of the following: • Right-click on the brown tile `HelloPrinter` & choose `Compile` OR • Click on the brown tile & do `Tools > Compile` OR • Click on the brown tile & click button `Compile` in the left panel OR • Double-click on the brown tile (& see source code), and click on `Compile` button in the top bar ***NOTE**: You have just learned 3 ways to compile in BlueJ, which invoke the “javac.exe” application/tool from the JDK.* **Questions**: a) In BlueJ, how did the brown tile for `HelloPrinter` change? What do you think caused the change? b) In `Proj01` folder on disk, what was just added? Use full names (i.e. including the extensions) for files. I.e., what is the output of the compiler?		a) In BlueJ, the stripes over brown tile disappeared. The successful compile took the stripes away. b) In the `Proj01` folder, there are 2 additional files “HelloPrinter.class” and “HelloPrinter.ctxt”. The “class” file is the bytecode, i.e., the output of the compiler. We are not going to pay attention to the “ctxt” file, which is used by BlueJ only.

5	**Run a program in a BlueJ project**, in `Proj01`: In BlueJ, right-click on the brown tile & choose “`void main(String[] args)`”, then click OK **Question**: a) What happened? b) There is a name for this kind of programs, where the input & output between the computer & user is ***text-based*** (i.e., no fancy pictures or images or GUI). What is this kind of programs called?		a) The program launched, and displayed “Hello, World!” in a separate window called “terminal window”. b) It is called **console programs**.
6	**Explore BlueJ:** a) What are the menus in the **project window**? b) What indicates that a “class” tile is selected? c) What are the menu items in the **terminal window** or **console window**? ***Note***: it is a good idea, in the Option menu, to ***always*** check “Clear screen at method call” and “Unlimited buffering”. d) What happens when a class tile in the project window is double-clicked? ***<u>NOTE</u>**: We have just identified **3 kinds of windows** in BlueJ.*		a) 5 menu items: Project, Edit, Tools, View, Help b) 2 slant lines at the lower right corner of a tile indicates that it is selected. c) 1 menu item: Options d) When the tile is double-clicked, the source file is opened, ready to be edited. We will refer to this window as the **class window**.
7	**Study class `HelloPrinter`:** study the source code in the class window for `HelloPrinter`: a) What is the **header** of the `HelloPrinter` class? b) What is the **body** of the `HelloPrinter` class contained in? c) How many items are there in the body of the `HelloPrinter` class? d) What is the header of the **main method**? Why is the main method important? e) What is the body of the main method contained in? f) What is the **statement** that displays text in the terminal window? g) What are the pieces called in the (only) statement? `System.out.println(“ ……”);` ***<u>NOTE</u>**: All of these questions relate to Java syntax; and you must memorize the correct syntax.*		a) `public class HelloPrinter` b) The body of the class is contained in a pair of braces: {} c) There is only one: the main method d) The header is `public static void main (String[] args)`. The main method is important because (1) each Java application must have a main method, and (2) when a Java application is launched, the starting point is the first line of the main method. e) The body of the main method is contained in a pair of braces: {} f) It is the “print line” statement: `System.out.println(“ ……”);` g) `System.out.println` is the method call; the content of the parentheses is the parameter (or input) to the call. The parameter in this case is a `String`, which must appear in double quotes. The statement must end with a semi-colon.

8	**Delete a class from a BlueJ project**, in `Proj01`: • In the project window, click on the `HelloPrinter` tile and choose “Edit > Remove” **Questions**: a) In BlueJ, what is or is not in the project window? b) In the `Proj01` folder on disk, what is there? c) In the **StudentsFiles** folder, is the `HelloPrinter.java` file still there? Now “add class from file ...” the `HelloPrinter` class from the **StudentsFiles** folder, so as to continue with the next steps		a) The tile was deleted from the project window. b) All the files related to this `HelloPrinter` class were deleted from the `Proj01` folder on disk. c) Yes, it is still there; the **StudentsFiles** folder is not affected (same as when a class was added to the Java project).
9	**Print special characters (escape sequence and characters from Unicode Table), and various characters in one statement**: • Lessons/demos on special characters & **Unicode Table**, and operator **+** • In the BlueJ, bring up `HelloPrinter` to edit **Questions**: a) Edit the `print` statement to show: Hello <*your classmate’s name*> from <*your name*> ☺ b) Change `println` to `print`, and add a `print` statement passing to it 3+4. What is the difference between `println` & `print`? `System.out.print(3+4);` Add a `println` statement passing to it the word “times” c) Replace all the `print` and `println` statements with ***one*** `println` statement, using the **concatenation operator**, to show the same message d) Replace ☺ with something else in the Unicode Table, and put a tab in the beginning ***<u>NOTE</u>***: • If both operands are numeric, the plus sign adds the 2 numbers • If at least 1 operand is a string, the plus sign concatenates the strings (after converting the numeric, if any, to string). ***<u>Note</u>***: The string operand does not have to be the 1st; it may be the 2nd operand also.		a) Compile, run & show to your classmate, who should verify the result. What is displayed? b) Compile, run & show to your classmate, who should verify the result. What is displayed? c) Compile, run & show to your classmate, who should verify the result. What is displayed? d) Which Unicode character/symbol did you choose?

10	**Convention, followed by programmers – Do the following & answer the questions:** a) In the `HelloPrinter` class window, change the name `HelloPrinter` to `helloPrinter`. Does the program still compile without errors? Does it still run as before? What is the name of the tile? What are the names of the files in the `Proj01` folder? ***Note*: it is *good convention* (i.e., not syntax) to name a *class* similar to `HelloPrinter`**: use an upper-case letter at the start of each word in the name, including the very first letter of the name). b) Observe the color blocks. Which color block contains which color block? c) In the `HelloPrinter` class window again, delete the lines bounded by /* … */. Does the program still compile & run correctly? Why or why not? d) Do the following: • Copy the entire `HelloPrinter` class • Click the “New class” button in the left panel, or choose “Edit > New class” • Delete the whole contents of this new class • Paste `HelloPrinter` • Change the class name from `HelloPrinter` to `Temp` as the in the window • Delete all “new lines” and tabs/indentations, to make the whole class as a block of very-hard-to-read text. • Compile & run Does the program run successfully with good result? Why or why not? e) Observe the color “blocks” again. Are they similar to (b)? f) Remove `Temp` from this project. Did the `Temp` tile disappear?		a) Yes, the program still compiles and runs correctly, but not respected by other fellow programmers. The name of the tile in BlueJ and the names of the files in the disk change to “`helloPrinter`”, i.e., lower case. ***NOTE***: By convention, Java programmers use camel-case to name classes. Java compiler does not insist on certain case; and it keeps the name consistent all throughout. b) Green block is for the whole file/class, which contains the yellow block for the class body, which contains the white block for the body of the main method. c) Yes, the program still compiles and runs correctly. The text bounded by /* … */ is documentation, for programmers; the compiler ignores all such text. d) Yes, the program runs successfully because all the new lines and indentations are for humans only; the compiler looks for things like braces & semi-colons. e) Yes f) Yes

11	**Syntax, expected by the compiler – Do the following with `HelloPrinter`, and answer the questions:** a) What happens if you remove one or both double quotes around `Hello, World`? b) Change the lower/upper cases of any text other than the class name or the text inside the double quotes; for example `public` or `class` or `System.` Does the program still compile & run correctly? Why or why not? c) Delete any of the semi-colon at the end of any statement. Does the program still compile & run correctly? Why or why not? ***NOTE***: The **reserved words** are those that ***must*** appear as-is. They are part of the **Java syntax**.		a) This causes a compile-time error; text must be inside double quotes b) No, the program no longer compiles. That is because they are reserved words, as expected by the compiler, and are case-sensitive. For example, you must use `public`, not `Public; System`, not `system`, etc. c) No, the program no longer compiles. The Java compiler expects a semi-colon at the end of each statement; thus, when it is not there, a compile error occurs.
12	**Logic, expected by you/programmer – Do the following with `HelloPrinter`, and answer the questions:** a) Delete the letter "l" from the word "World". Does the program still run? What is the result? b) What is the result if you put **`3+4`** in double quotes? Why is the result not the same as previously? c) What is the result if you put only one number in double quotes: **`3+"4"`**? Why is the result like that?		a) Yes. Similar to previous result, but with the misspelled "World". b) The result is **`3+4`**. When in quotes, it is treated as a `String` of text. c) The result is **`34`**. When in double quotes, a number is treated as a `String`. When at least one operand of operator + is a `String`, the operator means concatenation (i.e., not addition). The non-`String` operand is converted to `String` first, then the concatenation is applied.
	Correct your answers, noting mistakes to learn from (if any) Hand in your work to be graded.		

THE END