ICT167 ANS1

NOTE: JAVA always PASS BY VALUE (primitive) unless object

Passing by value means you're giving a subprogram nothing but the contents of a variable. AKA it means that the recipient receives a **copy of the value.**

Passing by reference means you're giving it the whole variable. AKA changes will affect the original

---

https://www.javatpoint.com/post/java-scanner-next-method

***Steps for establishing program requirements:**

- Input data to be supplied
- Tasks to be performed
- Output results to be produced

Process of executing Java Program:

- Java program is compiled by the compiler (javac): source code translates to byte code (not machine code)
    - Byte code: intermediate layer in-between human and machine*
    - Javac: is the standard JDK compiler found on nearly every machine*
- Byte code is translated by the interpreter (JVM): byte code translates to machine code thus it can be executed by CPU
    - Machine code: 0's and 1's*
    - JVM: Java Virtual Machine is another application that translates byte code. It is good because the JVM is available on many different platforms (windows, OS, browsers etc) thus can be customized and working for any particular platform*

Low level PsuedoCode:

- Lowest level contains small indivisible steps

High level PsuedoCode: (Procedure Abstraction)

- Basic steps are not single indivisible steps of machine code. Complex methods can be summarize by a single word
- Use words/sentences to describe
- EG- (*firstchar* != q)
- EG- get an input number d from the user

Abstraction: describes the essential features of something while leaving out the details. Types

- **Procedure abstraction:** purpose is to allow for the use of simple names to capture complex procedure (which are methods). So the basic steps are not small indivisible steps of machine code. Used in *high level pseudo-code*
- **Data abstraction:** Data abstraction allows the data in the program to be organised by putting related data together and having a simple name for the big lump of data that we call
    - EG: String

***Java:**

- It is an object-oriented (OO) language
- Allows re-use of software written before
- Java is case sensitive
- Java is made up of one or more classes. Each class is normally a separate file
- Class contains one or more methods which perform task in program
- Java application executes main method first
- Allow for inheritance, polymorphism and dynamic binding

C vs Java:

- C programming has no Boolean (false/true) data type. Java has Boolean
- C uses ASCII character. Java uses Unicode character set
- C used scanf() and printf(). Java uses methods of library classes like Scanner and System.out
- C needs to specific whether pass by reference or value. Java automatically does it.
- C is procedural language. Java is object-oriented language
- C contains a main() function. Java its called a main() method

To use classes we need to instantiation/How to create an object for the class:

Every object is an instance of a class Object = Class but Every class is not an object

```
[Class] [class variable] = new [Class name]([parameters]);
```

- Scanner [class variable] = new Scanner(system.in);
- String [class variable] = new String("Hello"); → String s = "Hello";

Java API:

- Java Application Programming Interface (API) is a collection of classes (class libraries) that can be used to support program development

- Classes in a class hierarchy is often related to inheritance
- Classes in the API are separated into packages which can be nested
- Each package contains a set of classes that are intertwined

Infinite loop: when a while (true) is used and you need to press control c to quit.

Design methodology:

- Systematic approach to design, and supports good design by helping with design task and possibly with description of it
- Suggest a tried and tested general way of coming up with a good design
- Allow a programmer to reuse other designs by allowing easier understanding and communication of designs

Procedure:

- Used to capture a common and well defined task
- Can decide what input(s) is needed for the procedure to complete the task.
    - Via Procedure parameters
- Can decide the output the procedure returns to either the main program or another procedure although sometimes there are no outputs
    - Via Return value
- Consist of a procedure definition and procedure body

Structured design:

- The main structured design used is top down refinement. This method involves breaking the problem into steps involving the major sub tasks and then implementing each sub task as a procedure in the same way
- Call graph show which procedures call which procedures

<u>Type casting:</u>

Type cast can be used to change the data type of a value from its declared → another data type

```
Variable = ([DataType]) value;
```

Pseudo-code procedure definition:

- Any parameters means that output from other procedures are taken inside this procedure

```
Procedure [Data type] Method([Parameters])
…
Return
End Procedure
```

***Pseudo-code calling methods:**

```
Method([class variable],[Parameter])
```

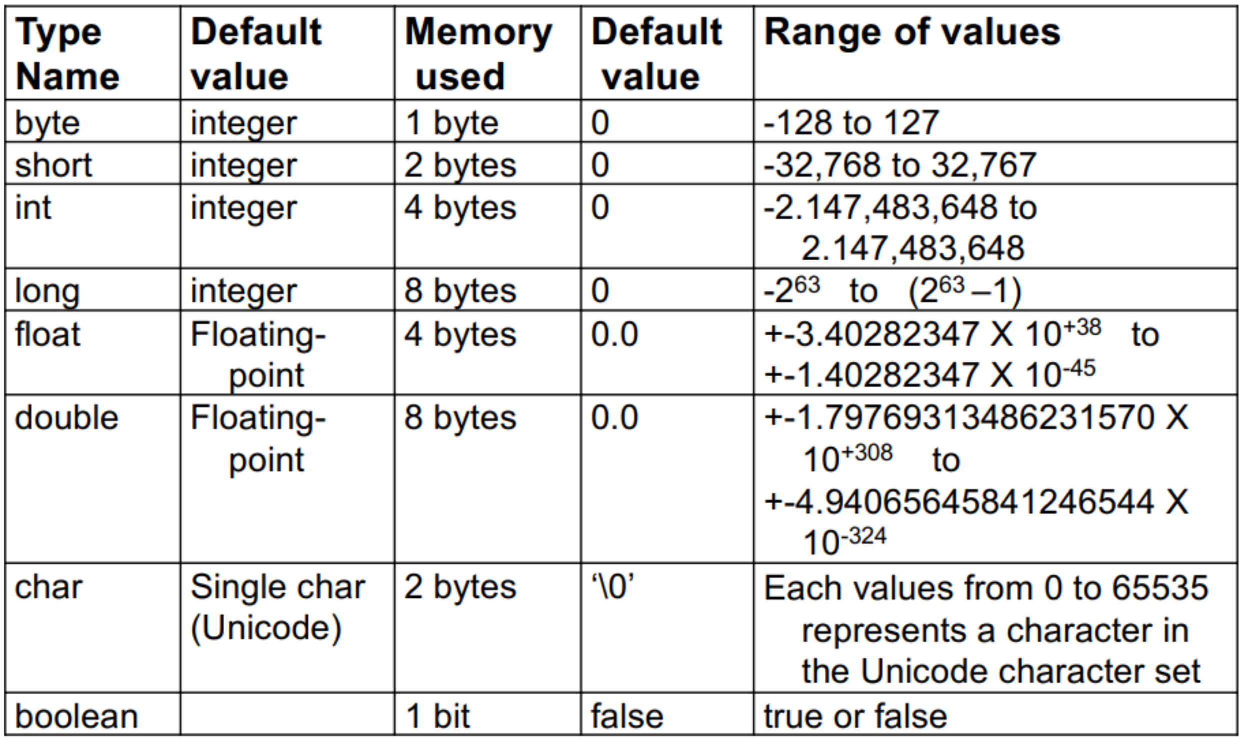
***Create constant:**

```
Final [Datatype] [variable] = ….;
```

File name: SmallIO.java

Must be the same as class name

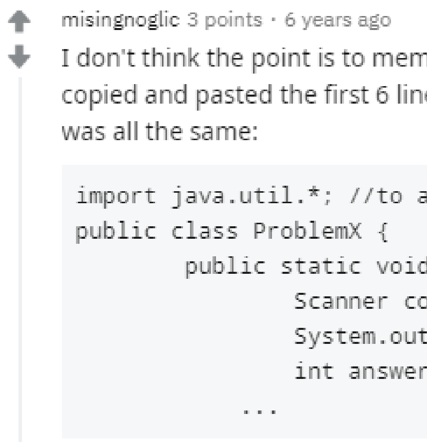
Java Data types:



Type Name	Default value	Memory used	Default value	Range of values
byte	integer	1 byte	0	-128 to 127
short	integer	2 bytes	0	-32,768 to 32,767
int	integer	4 bytes	0	-2.147,483,648 to 2.147,483,648
long	integer	8 bytes	0	-2^{63} to $(2^{63}-1)$
float	Floating-point	4 bytes	0.0	+-3.40282347 X 10^{+38} to +-1.40282347 X 10^{-45}
double	Floating-point	8 bytes	0.0	+-1.79769313486231570 X 10^{+308} to +-4.94065645841246544 X 10^{-324}
char	Single char (Unicode)	2 bytes	'\0'	Each values from 0 to 65535 represents a character in the Unicode character set
boolean		1 bit	false	true or false

Declaring →

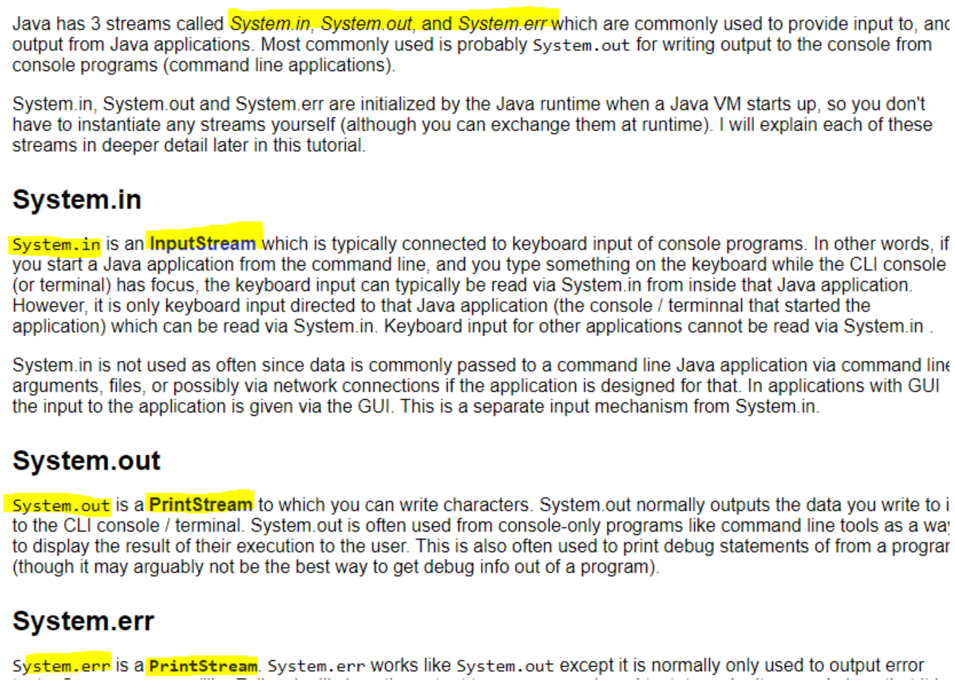
Boolean [variable name] = true/false;



misingnoglic 3 points · 6 years ago

I don't think the point is to mem
copied and pasted the first 6 lin
was all the same:

```
import java.util.*; //to a
public class ProblemX {
        public static void
                Scanner co
                System.out
                int answer
        ...
```



Java has 3 streams called System.in, System.out, and System.err which are commonly used to provide input to, and output from Java applications. Most commonly used is probably `System.out` for writing output to the console from console programs (command line applications).

System.in, System.out and System.err are initialized by the Java runtime when a Java VM starts up, so you don't have to instantiate any streams yourself (although you can exchange them at runtime). I will explain each of these streams in deeper detail later in this tutorial.

System.in

`System.in` is an **InputStream** which is typically connected to keyboard input of console programs. In other words, if you start a Java application from the command line, and you type something on the keyboard while the CLI console (or terminal) has focus, the keyboard input can typically be read via System.in from inside that Java application. However, it is only keyboard input directed to that Java application (the console / terminnal that started the application) which can be read via System.in. Keyboard input for other applications cannot be read via System.in .

System.in is not used as often since data is commonly passed to a command line Java application via command line arguments, files, or possibly via network connections if the application is designed for that. In applications with GUI the input to the application is given via the GUI. This is a separate input mechanism from System.in.

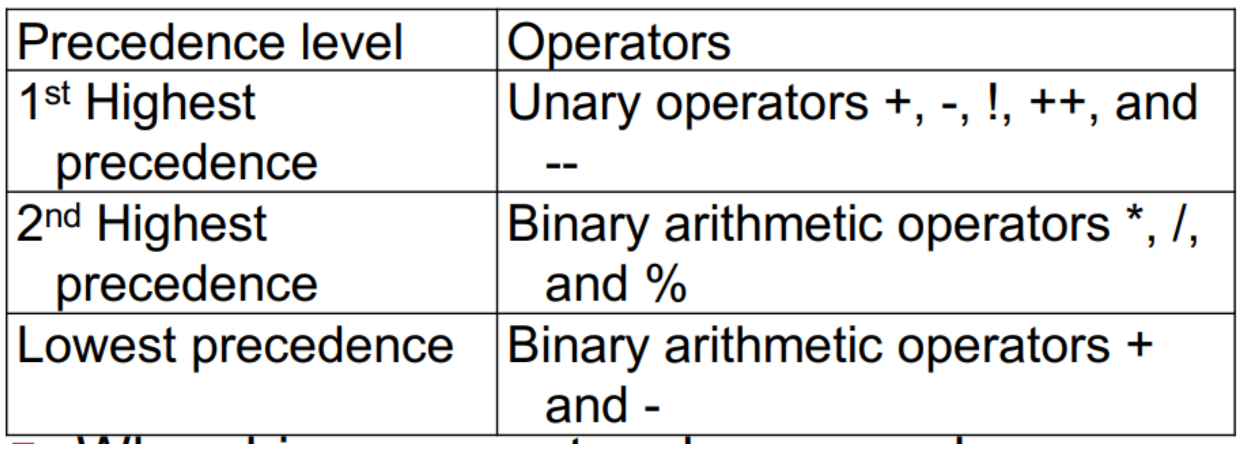
System.out

`System.out` is a **PrintStream** to which you can write characters. System.out normally outputs the data you write to i to the CLI console / terminal. System.out is often used from console-only programs like command line tools as a wa to display the result of their execution to the user. This is also often used to print debug statements of from a prograr (though it may arguably not be the best way to get debug info out of a program).

System.err

`System.err` is a **`PrintStream`**. `System.err` works like `System.out` except it is normally only used to output error

<u>Precedence:</u>



Precedence level	Operators
1st Highest precedence	Unary operators +, -, !, ++, and --
2nd Highest precedence	Binary arithmetic operators *, /, and %
Lowest precedence	Binary arithmetic operators + and -

Questions

1. Why do we need to declare new Scanner() to use scanner class but not new System() to use system class