Java Basic Fundamentals

Java Statements

Statement	*Description*
Empty Statement	These do nothing and are used during program development as a place holder.
Labelled Statement	Any Statement may begin with a label. Such labels must not be keywords, already declared local variables or previously used labels in this module. Labels in Java are used as the arguments of Jump statements, which are described later in this list.
Expression Statement	Most statements are expression statements. Java has seven types of Expression statements: **Assignment, Pre-Increment, Pre-Decrement, Post-Increment, Post-Decrement, Method Call and Allocation Expression.**
Selection Statement	These select one of several control flows. There are Three types of selection statements in Java: **if, if-else,** and **switch**.

In Java, we can give a label to a block of statements. A label is any valid Java variable name. To give a label to a loop, place it before the loop with a colon at the end. Example:

```
loop1:    for (.........)
          {
               .........
               .........
          }
          .........
```

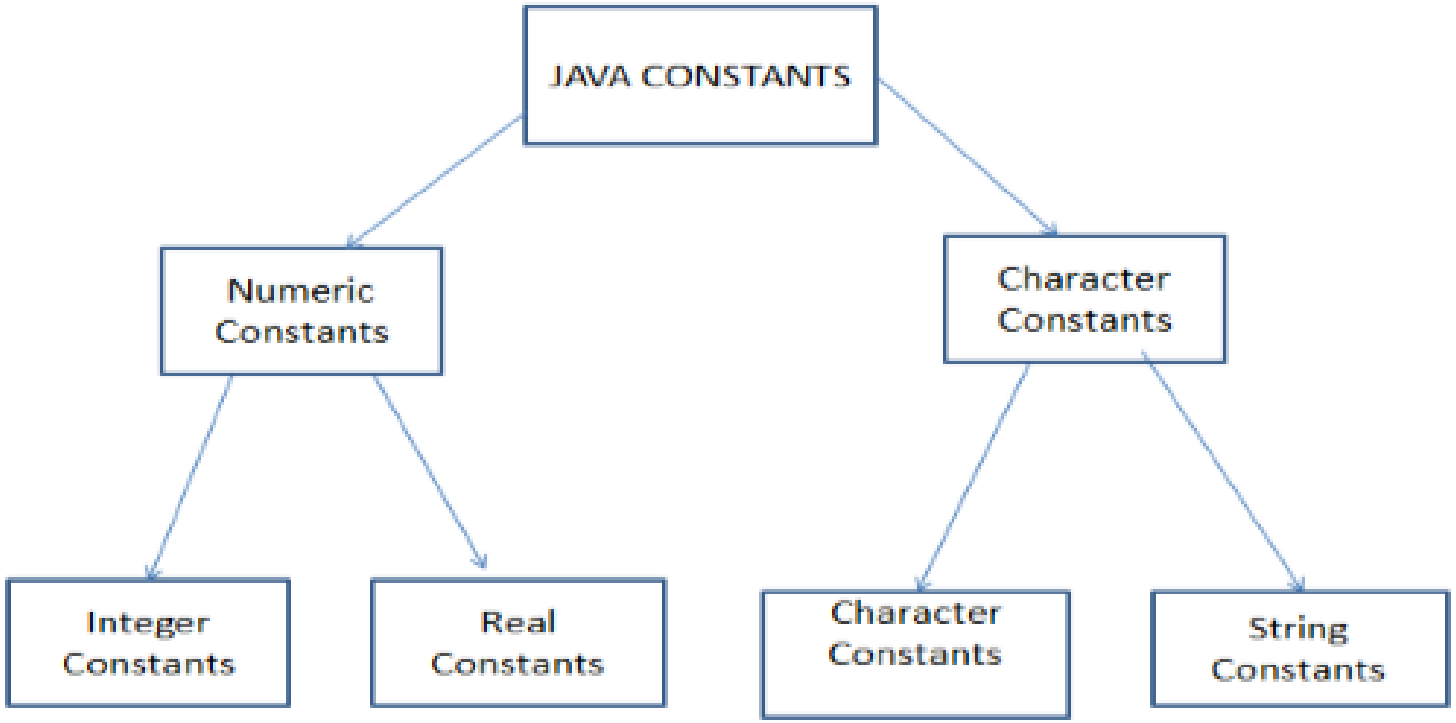
Iteration Statement	These specify how and when looping will take place. There are three types of iteration statements; **while, do** and **for**.
Jump Statement	Jump Statements pass control to the beginning or end of the current block, or to a labeled statement. Such labels must be in the same block, and **continue** labels must be on an iteration statement. The four types of Jump statement are **break, continue, return** and **throw**.
Synchronization Statement	These are used for handling issues with multithreading.
Guarding Statement	Guarding statements are used for safe handling of code that may cause exceptions (such as division by zero). These statements use the keywords **try, catch,** and **finally**.

Java Constants

- Constants are referred as a fixed value that do not change during the execution of the program.

- Java Supports following constants

1. Integer Constants
2. Real Constants
3. Single Character Constants
4. String Constants
5. Backslash Character Constants


JAVA CONSTANTS
Numeric Constants
Character Constants
Integer Constants
Real Constants
Character Constants
String Constants

Integer Constants

- It is referred as sequence of digits. There are three type of digits.

1. Decimal Integer :It consist of set of digit 0 to 9 preceding with optional minus sign. No embedded spaces, commas, non-digit characters
2. Octal Integer : It consist of any combination of 0 to 7 with leading 0. eg. 037, 0, 047
3. Hexadecimal Integer :It consist of digit preceding with 0x or 0X.They may also include alphabets A to F or a to f to represent 10 to 15.

Real Constant

- It is nothing but floating point constant. These numbers have a whole number followed by decimal point and fractional part which is an integer.
- Example: 325.65 --> 3.2565e2

  e2 means multiply by 10^2

General format is,

mantissa e exponent

3.2565e2

- Mantissa can be either real number in decimal notation or integer.
- Exponent is an integer with an optional plus or minus sign.
- Letter e can be either in lowercase or uppercase.

Therefore four parts as

- A whole number
- A decimal point
- Fractional part
- An exponent

Single Character Constant

- A single character constant contains a single character enclosed within a pair of single quote marks.

- Eg. ‘5’, ‘X’, ‘ ’

String Constants

- Sequence of characters enclosed in double quote is nothing but String Constants

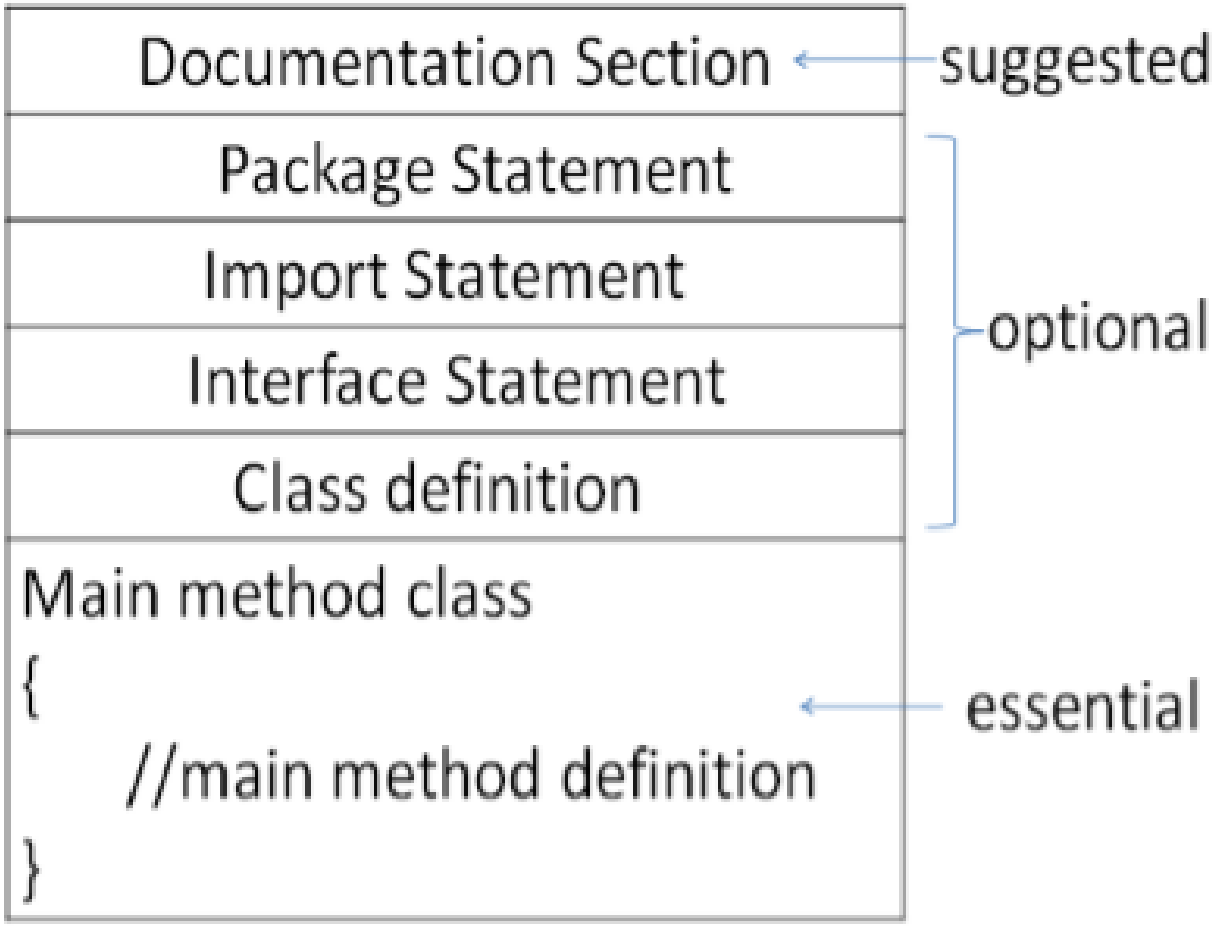
- Eg. “Hello Java ”, “1988”.

Backslash Character Constants

- These are used in output methods.
- Each one of them represent one characters although they consist of two characters.
- These are known as Escape Sequences.
- Example:-

1. \n new line,
2. \b back space
3. \f form feed
4. \t horizontal tab

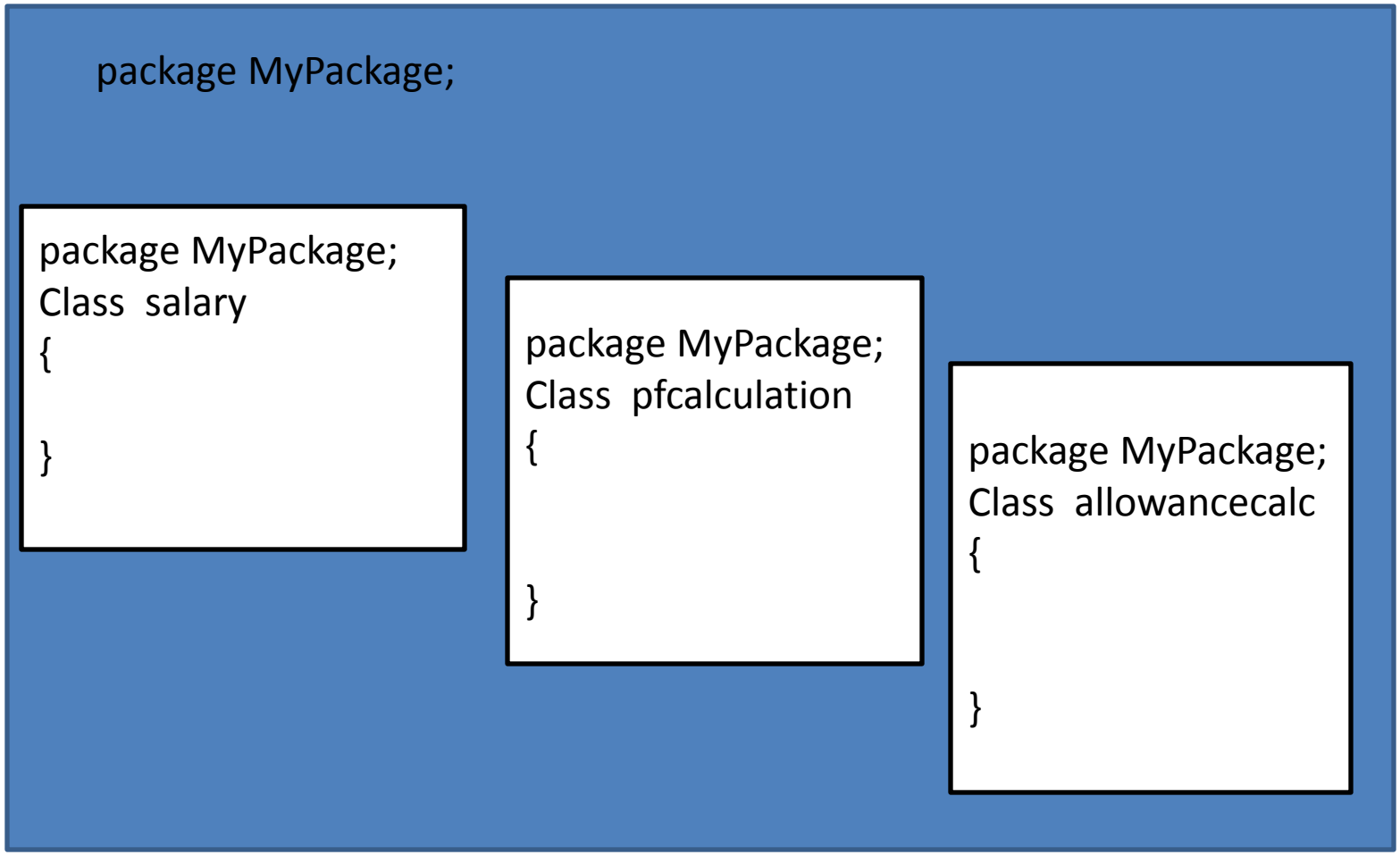
Java Program Structure


Documentation Section
suggested
Package Statement
Import Statement
Interface Statement
Class definition
optional
Main method class
{
//main method definition
}
essential

- Documentation Section: It comprises a set of comment lines. Java supports
- single line comment : //
- multi-line comments : /* and ends with */ .
- Java supports third type of comment /**……*/ known as documentation comment.

Package Statement

- The first statement allowed in a Java file is a package statement.
- This statement declares a package name and informs the compiler that the classes defined here belong to this package.
- Example: package mypackage;
- The package statement is optional.


package MyPackage;
package MyPackage;
Class salary
{
}
package MyPackage;
Class pfcalculation
{
}
package MyPackage;
Class allowancecalc
{
}

```
import MyPackage.*;
Class employee
{
public static void main(String args[])
{



}
}
```

Import Statements:

- Next thing after package statement may be number of import statements.
- This is similar to #include statement in C.
- For example, import mypackage.operation; This statement instructs the interpreter to load operation class contained in mypackage class.

Interface Statements

- An interface is like a class but includes a group of abstract methods.

- This is an optional section and is used only when we have to implement multiple inheritance feature in the program.

Class Definition

- A Java program may contain multiple class definitions.
- Classes are primary and essential elements of the Java program.
- The number of classes used depends on the complexity of the program.

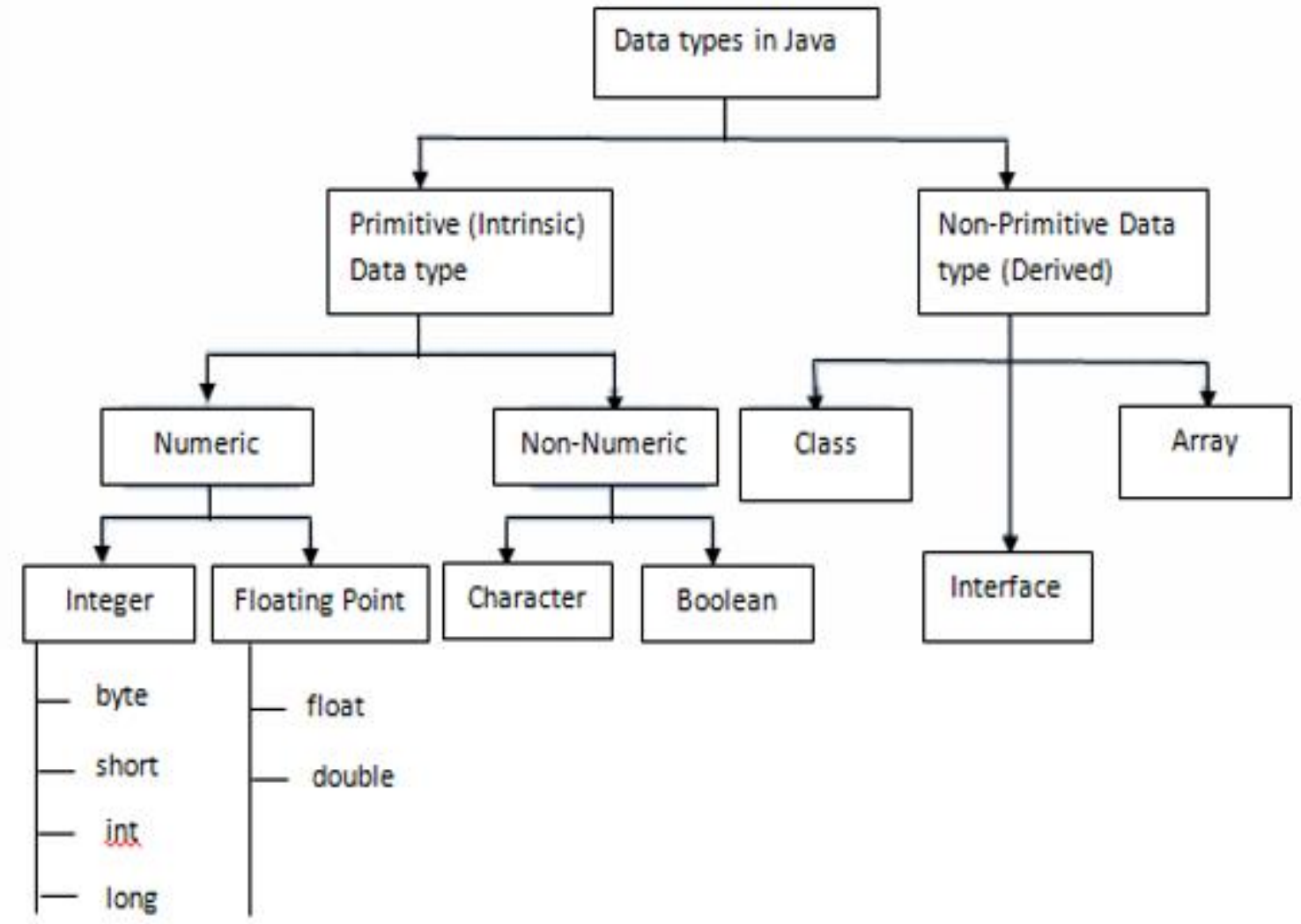
Main Method Class

- Every Java stand-alone program requires a class containing main method that is the starting point of execution.
- Therefore this class is the essential part of a Java program.
- In main method we create objects of various class and establish communication between them.

Simple Java Program

```
class Sampleone
{
public static void main(String args[])
{
System.out.println("Hello BCA");
}
 }
```

Java Datatypes


Data types in Java
Primitive (Intrinsic) Data type
Non-Primitive Data type (Derived)
Numeric
Non-Numeric
Class
Array
Integer
Floating Point
Character
Boolean
Interface
byte
short
int
long
float
double

- Derived Data type is nothing but collection of same data type. eg: Array

- User defined data type is collection of different data type eg: class

Integer Type

Type	*Size*	*Minimum value*	*Maximum value*
byte	One byte	–128	127
short	Two bytes	–32, 768	32, 767
int	Four bytes	–2, 147, 483, 648	2, 147, 483, 647
long	Eight bytes	–9, 223, 372, 036, 854, 775, 808	9, 223, 372, 036, 854, 775, 807

We can make integers **long** by appending the letter L or l at the end of the number. Example:

```
123L    or    123l
```

Floating Point Type

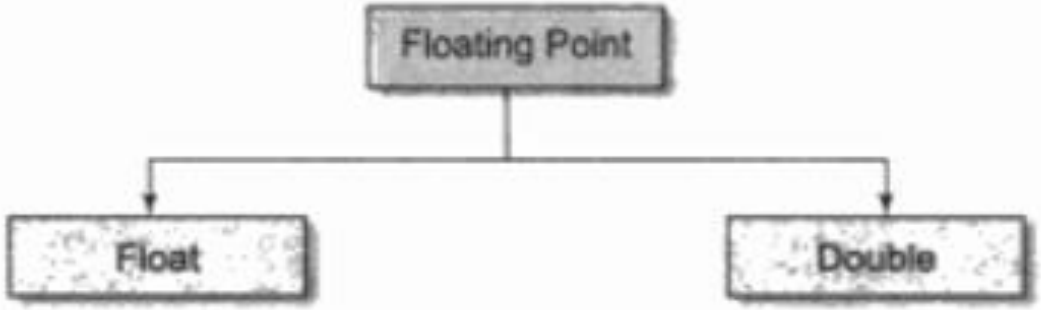
Integer types can hold only whole numbers and therefore we use another type known as *floating point* type to hold numbers containing fractional parts such as 27.59 and –1.375 (known as floating point constants). There are two kinds of floating point storage in Java as shown in Fig.

The **float** type values are *single-precision* numbers while the **double** types represent *double-precision* numbers. Table gives the size and range of these two types.

Floating point numbers are treated as double-precision quantities. To force them to be in single-precision mode, we must append f or F to the numbers. Example:

```
1.23f
7.56923e5F
```

Type	*Size*	*Minimum value*	*Maximum value*
float	4 bytes	3.4e-038	3.4e+038
double	8 bytes	1.7e-308	1.7e+308



All mathematical functions, such as sin, cos and sqrt return **double** type values.

Floating point data types support a special value known as Not-a-Number (NaN). NaN is used to represent the result of operations such as dividing zero by zero, where an actual number is not produced. Most operations that have NaN as an operand will produce NaN as a result.

Data Type	**Default Value (for fields)**
byte	0
short	0
int	0
long	0L
float	0.0f
double	0.0
char	'\u0000'
String (or any object)	null
boolean	false

Character Type

In order to store character constants in memory, Java provides a character data type called **char**. The char type assumes a size of 2 bytes but, basically, it can hold only a single character.

Boolean Type

Boolean type is used when we want to test a particular condition during the execution of the program. There are only two values that a boolean type can take: **true** or **false**. Boolean type is denoted by the keyword **boolean** and uses only one bit of storage.