Java Basics

- Program – set of instruction
- Collection of every single thing called tokens
- Tokens could be –An
- Identifier
- Keyword
- Operator
- Datatype

Java Program

- 1. class classname
- 2.{
- 3.Public static void main(String [] a)
- 4.{
- 5. //a statement ends with semicolon
- 6. System.out.println(“Hello”);
- 7. }
- }

Identifier

- Name of a variable ,function, array, class
- Rules- Never a keyword,
- Never start with number
- No symbols except underscore in middle
- Ex temp_1

keyword

- Set of words that are define by Language with specific meaning
- Ex: main, public, class, void , int, char, float, long, double, if else…
- The identifiers like class, function variable name cannot be keyword

Operators

- Alll arithmetic, relational, condition,logical
- Depending on number of values/operands taken there are
- 1)Unary
- 2)Binary

Arithmetic

Operator	Description
+	adds two operands
-	subtract second operands from first
*	multiply two operand
/	divide numerator by denumerator
%	remainder of division
++	Increment operator increases integer value by one
--	Decrement operator decreases integer value by one

Relational

Operator	Description
==	Check if two operand are equal
!=	Check if two operand are not equal.
>	Check if operand on the left is greater than operand on the right
<	Check operand on the left is smaller than right operand
>=	check left operand is greater than or equal to right operand
<=	Check if operand on left is smaller than or equal to right operand

Logical

Operator	Description	Example
&&	Logical AND	(a && b) is false
||	Logical OR	(a || b) is true
!	Logical NOT	(!a) is false

Bitwise

Operator	Description
&	Bitwise AND
|	Bitwise OR
^	Bitwise exclusive OR
<<	left shift
>>	right shift

Assignment

Operator	Description	Example
=	assigns values from right side operands to left side operand	a = b
+=	adds right operand to the left operand and assign the result to left	a+=b is same as a=a+b
-=	subtracts right operand from the left operand and assign the result to left operand	a-=b is same as a=a-b
*=	mutiply left operand with the right operand and assign the result to left operand	a*=b is same as a=a*b
/=	divides left operand with the right operand and assign the result to left operand	a/=b is same as a=a/b
%=	calculate modulus using two operands and assign the result to left operand	a%=b is same as a=a%b

Condition

- ?:

Variables

- Define the type of value used
- When we want to store any information, we store it in an address of the computer. Instead of remembering the complex address where we have stored our information, we name that address.The naming of an address is known as variable. Variable is the name of memory location.
- In other words, variable is a name which is used to store a value of any type during program execution.
- To declare the variable in Java, we can use following syntax
- datatype variableName; Here, **datatype** refers to type of variable which can any like: **int, float** etc.
  and **variableName** can be any like: **empId, amount, price** etc.
-

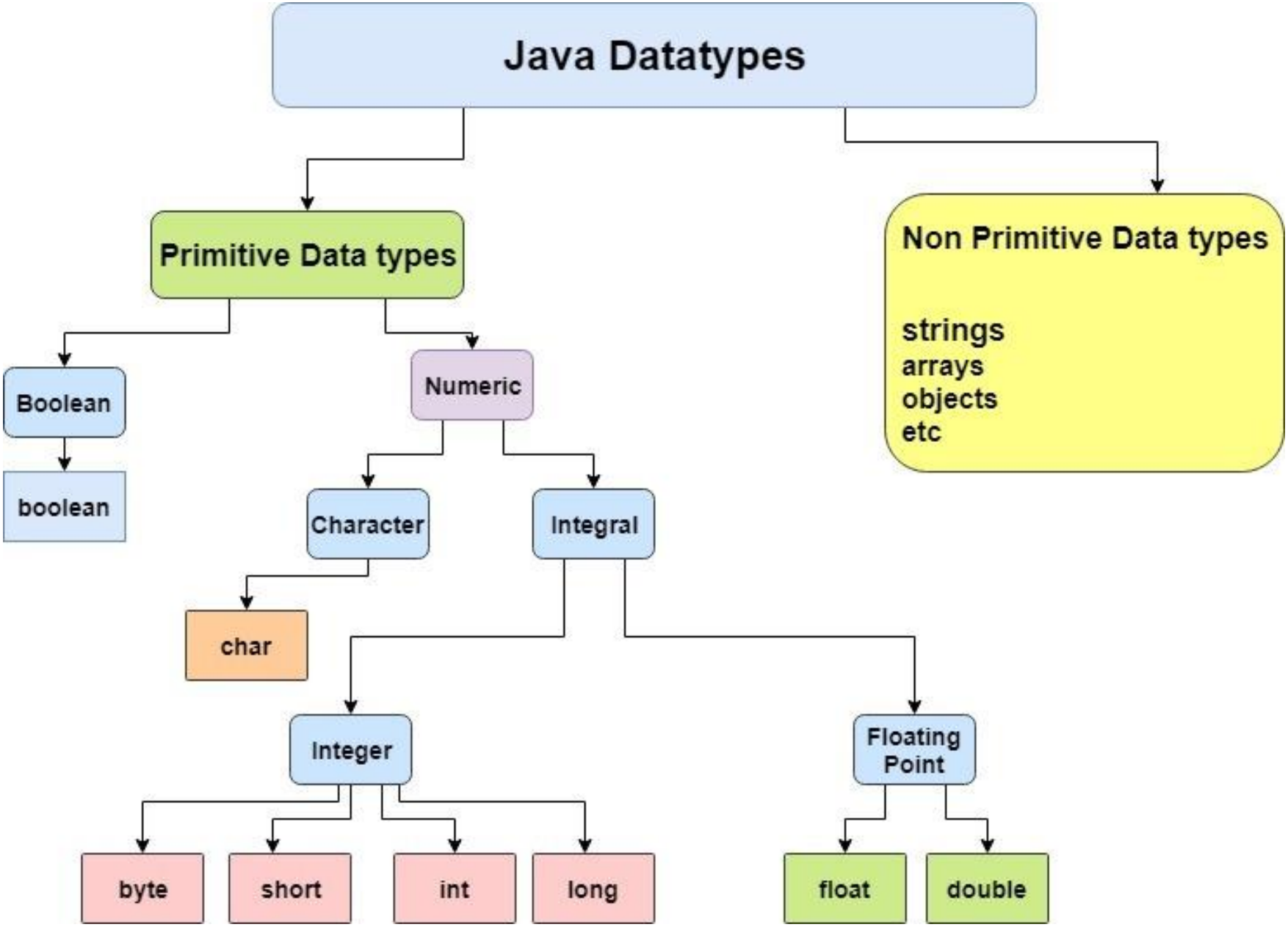
- Java Programming language defines mainly three kind of variables.
- Instance Variables
- Static Variables (Class Variables)
- Local Variables

Variable Scope in Java

- Scope of a variable decides its accessibility throughout the program. As we have seen variables are different types so they have their own scope.
- **Local variable**: Scope of local variable is limited to the block in which it is declared. For example, a variables declared inside a function will be accessible only within this function.
- **Instance variable**: scope of instance variable depends on the access-modifiers **(public, private, default)**. If variable is declared as **private** then it is accessible within class only.
- If variable is declared as **public** then it is accessible for all and throughout the application.
- If variable is declared as **default** the it is accessible with in the same package.
-

```
class Student{
int a;
static int id = 35;
void change()
{
System.out.println(id);
 }
 }
 public class Study
 {
public static void main(String[] args)
 {
 Student o1 = new Student();
 Student o2 = new Student();
o1.change();
 Student.id = 1;
 o2.change();
 }
 }
```

- class Student
- {
- String name;
- int age;
- static int instituteCode=1101;
- }


Java Datatypes
Primitive Data types
Non Primitive Data types
strings
arrays
objects
etc
Boolean
boolean
Numeric
Character
Integral
char
Integer
Floating Point
byte
short
int
long
float
double

Range of Datatypes

- **Integer**
- This group includes byte, short, int, long
- **byte :** It is 1 byte(8-bits) integer data type. Value range from -128 to 127. Default value zero. example: byte b=10;
- **short :** It is 2 bytes(16-bits) integer data type. Value range from -32768 to 32767. Default value zero. example: short s=11;
- **int :** It is 4 bytes(32-bits) integer data type. Value range from -2147483648 to 2147483647. Default value zero. example: int i=10;
- **long :** It is 8 bytes(64-bits) integer data type. Value range from -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807. Default value zero. example: long l=100012;
-

Floating-Point Number

- This group includes float, double
- **float :** It is 4 bytes(32-bits) float data type. Default value 0.0f.
- example: float ff=10.3f;
- **double :** It is 8 bytes(64-bits) float data type. Default value 0.0d.
- example: double db=11.123d;
-

Characters

- This group represent char, which represent symbols in a character set, like letters and numbers.
- **char :** It is 2 bytes(16-bits) unsigned unicode character. Range 0 to 65,535.
- Default value: '\u0000'
- example: char c='a';
- unicode provides unique code to represent alphabet , currency or other characters that are comes under the unicode set.

String- Array or collection of chars

- It is Class
- Immutable
- Default value is Null

Public static void main(String a[])

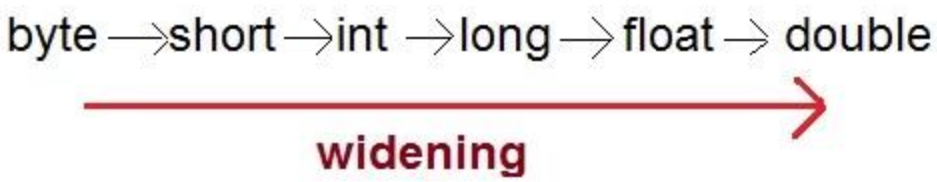
- Public – accesibility
- Static – independent , common
- Void fumction main return no value
- Main- function , indicates start of execution for System

Typecasting

- Casting is a process of changing one type value to another type.
- Cast one type of value to another type.
- It is known as type casting.
- int x = 10;
- byte y = (byte)x;

Upcasting

- In Java, type casting is classified into two types,
- Widening Casting(Implicit)
-



Downcasting

- Narrowing Casting(Explicitly done)

Example

- Widening or Automatic type conversion
- Automatic Type casting take place when,
- the two types are compatible
- the target type is larger than the source type
-

Widening/auto

- public class Test { public static void main(String[] args) { int i = 100; long l = i; //no explicit type casting required float f = l; //no explicit type casting required System.out.println("Int value "+i); System.out.println("Long value "+l); System.out.println("Float value "+f); }

narrowing

```
class Demo2 {
 public static void main(String args[])
 {
 byte b; int i = 355;
 double d = 423.150;
b = (byte) i;
 System.out.println("Conversion of int to byte: i = " + i + " b = " + b);
System.out.println("**********************************************
*****");
 b = (byte) d;
 System.out.println("Conversion of double to byte: d = " + d + " b= " + b);
}
```

Input values to variable

- 3 ways =>
- 1 initialisation
- 2 Scanner class
- 3 Command line Argument

Command line arguments

- Argument that passed to a program during runtime.
- It is the way to pass argument to the main method .
- Store into the String type **args** parameter which is main method parameter.
- To access these arguments, you can simply traverse the args parameter in the loop or use direct index value because args is an array of type String.
-

```
class cmd1
{
public static void main(String s[])
{
String s1= s[0];
String s2= s[1];
System.out.println("Arg1 "+s[0]);
System.out.println("Arg2 "+s[1]);
}
}
```

C:\Windows\System32\cmd.exe

```
Microsoft Windows [Version 10.0.18363.1082]
(c) 2019 Microsoft Corporation. All rights reserved.

E:\CDAC_Course\Lab1_Assgn>javac cmd1.java

E:\CDAC_Course\Lab1_Assgn>java cmd1
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 0
        at cmd1.main(cmd1.java:5)

E:\CDAC_Course\Lab1_Assgn>java cmd1 abc pqr
Arg1 abc
Arg2 pqr

E:\CDAC_Course\Lab1_Assgn>java cmd1 abc
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 1
        at cmd1.main(cmd1.java:6)

E:\CDAC_Course\Lab1_Assgn>_
```

```
class cmd1
 {
public static void main(String[] args)
{
 for(int i=0;i< args.length;i++)
{
 System.out.println(args[i]);
 }
 }
```

- **Execute this program as**
- **Javac cmd1.java**
- **java cmd1 10 20 30**