## UNICODE and ASCII

Unicode is a universal character encoding standard. It defines the way individual characters are represented in text files, web pages and other types of documents.

Unlike ASCII(**American Standard code for information Interchange**), which was designed to represent only basic English characters, Unicode was designed to support characters from all languages around the world.

Difference between ASCII and Unicode is that ASCII represents lowercase letters (a-z), uppercase letters(A-Z), digits(0-9) and symbols such as punctuation marks while Unicode represents letters of English, Arabic, Greek etc., mathematical symbols, emoji covering a wide range of characters than ASCII. **Unicode is a superset of ASCII.**

ASCII supports 128 characters. Unicode supports a wide range of characters.

The ASCII uses 7 bits to represent a character while the Unicode uses 8 bit, 16 bit or 32 bit depending on the encoding type. Unicode require more space than ASCII.

ASCII codes of well known characters are (see page no 42):

| Characters | ASCII code |
|---|---|
| 0-9 | 48-57 |
| a-z | 97-122 |
| A-Z | 65-90 |
| Whitespace | 32 |

## Escape Sequences:

Non-graphic characters are those characters that cannot be typed directly from keyboard. Example, backspace, tabs, carriage return etc. These non-graphic characters can be represented by using escape sequences. An escape sequence character begins with a backslash(\) and it is followed by one or more characters.

| Escape Sequences | Non-Graphic Character |
|---|---|
| \t | Horizontal tab |
| \v | Vertical tab |
| \\ | Backslash |
| \’ | Single quote |
| \” | Double quotes |
| \b | Backspace |
| \0 | Null |
| \r | Carriage return |
| \n | New line feed |

e.g.

System.out.print(“Name: Anushi\n”+”Age: 4 yrs\n”+”Class: LKG”);

Output:

Name: Anushi

Age: 4 yrs

Class: LKG

e.g.

System.out.print(“Welcome to \”Computer\” world);

Output:

Welcome to “Computer” world

## Token

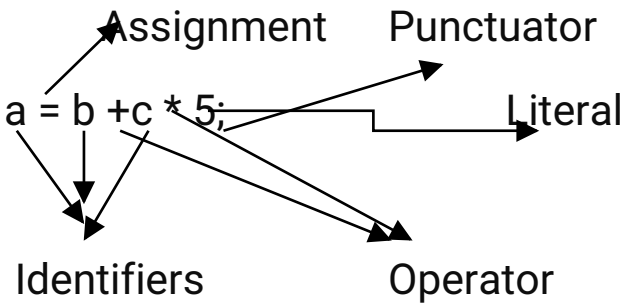
Each and every individual element present in a java expression is known as a token. A Token is the smallest element of a program that is meaningful to the compiler.

The various types of tokens available in Java are:

1) Literals 2) Assignment 3) Identifiers

4) Separators 5) Operators 7) Keywords

e.g



## Literals (Constants):

Literals in Java refer to fixed values that do not change during the execution of a program.

Java Support several types of constants

- Integer Literals: The number which are represented without decimal point. Whole numbers having positive and negative values. E.g. 17, -5, 396, 1682 etc
- Real or Float Literals: They represent number with decimal point. E.g. 17.0,24.6, -3.123, 1.0E-03 etc.
- Character Literals: A digit(0-9) or a character(a-z/A-Z) or a Symbol(@,#,$ etc). A character literal enclosed in single quotes.

  e.g. ‘a’, ‘X’, ‘@’ , ‘9’ etc.
- String Literals: A group of characters enclosed in double quotes.

  e.g. “ABC”, “Computer is fun” etc
- Boolean Literals: Boolean literals are special literals. They represent true or false and can be used in Java program to check whether a given logical condition is satisfied or not. Boolean constant are never enclosed within double quotes.
- Null Literal: This is another special purpose literal. It is represented as ‘\0’. It is used in a java program as a string terminator to mark the end of a string.

## Identifiers(Variables)

The name given to a package, interface, class, method or a

variable is known as identifier.

A variable is a named memory location, which contains a value. The value of a variable can change depending upon the circumstances and problems in a program.

```
int a; // variable declaration

a=5;  // variable initialization
```

## Rules for naming a Variable:

- Variable name only starts with a letter, an underscore (_) or a dollar sign($).
- Variable name cannot contain white space.
- Variable name cannot start with number.
- Java is a case sensitive language. The variable num is different from variable NUM.

## Static Initialization and Dynamic Initialization:

In Static initialization the variable is initialised at the time of its declaration. e.g.

```
int a=0;

float f= 0.0
```

In Dynamic Initialization the variable gets initialised during runtime (during execution of a program). e.g.

```
int a,b,c;

c=a+b;
```

## Punctuators:

Some of the punctuator are:

i) ? (Question mark)  ii) . (dot)  iii) ; (semi colon)

Question Mark (?): Represent the action to be taken when the given

condition is true while using the ternary operator.

e.g max=(a>b)? a : b;

Dot(.): Represent the scope of a function i.e. a function belonging to an object or class.

e.g. Math.max()- It represent that max function belongs to Math class.

Semi colon(;): It is used in a Java Program as a statement terminator. It indicates the end of a statement.

e.g. int a=5;

System.out.println(a); are treated as two separate statements in java.

## Seperators:

They are special character in Java, which are use to separate the variables or the characters.

e.g. Comma(,), Brackets (), Curly Brackets {}, Square Brackets [ ] etc.

Comma(,): It is used to speperate multiple variables under the same declaration.

e.g int a,b,c;

Brackets () : Used to enclosed any arithmetical or relational expressions. e.g. (a+b)*2;

Curly brackets { }: Used to enclose a group of statements under a compound statement.

Square Brackets [ ]: Used to enclose subscript or cell number of a array.

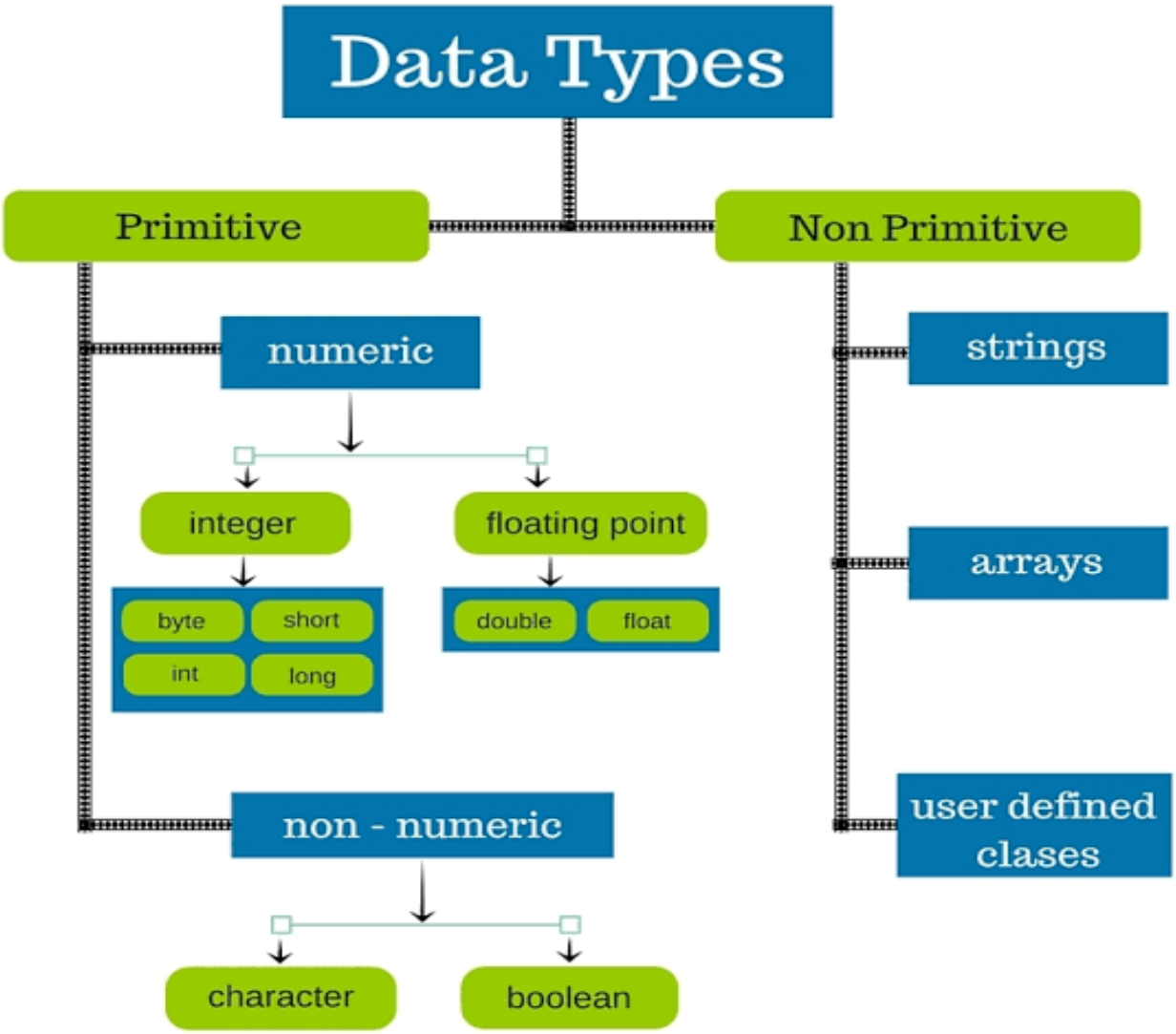
## Keywords:

Java reserved words or keywords are those words which carry special meaning to the java compiler. It can not be used as variables, methods, classes or any other identifiers.

Some of the reserved words or keywords are:

| | | | |
|---|---|---|---|
| case | switch | else | break |
| static | main | do | float |
| for | double | public | int |
| new | import | class | default |

## Data Types in Java:



## Primitive Data Type:

Basic or Built in Data Type which are independent on any other data type.

There are exactly eight primitive data types in Java

- Four of them represent integers: byte, short, int, long
- Two of them represent floating point numbers: float, double
- One of them represent characters: char
- And one of them represent Boolean (logical) values: boolean

## Non-Primitive Data Type

The data type that are derived from primary data types are known as non-primitive data type. Non-primitive data types are not defined by the programming language, but are instead created by the programmer. They are sometimes called “reference variable” or “object reference”, since they reference a memory location, which stores the data.Example class, array, interface.

## Pure Expression:

A pure expression in java is an expression containing variables and constants of the same data type only, separated by arithmetic operators (like +,-,*,/ etc). example 5 + 5, 10.2 +5.7 etc

## Impure or Mixed Expression:

An expression in which the two operands are not the same type is called a mixed or impure expression. Example of mixed expressions:

2 + 3.5

6/4 + 2.5

## Type Conversion:

In a mixed expression the result can be obtained in any one form its data types. Hence, it is needed to convert the various data types into a single type. Such conversion is termed as Type Conversion.

In java type conversion done in two different ways:

1) Implicit. 2) Explicit

## Implicit Type Conversion(Coercion):

In a mixed expression, the data type of the result gets automatically converted into the higher most type available in the expression without any user intervention.

e.g.

```
int  num = 5 + 12.75;        // num=17
double num = 5 + 12.75;  // num=17.75
int  x = 'A';                    // x = 65
```

## Explicit Type Conversion(Type Casting):

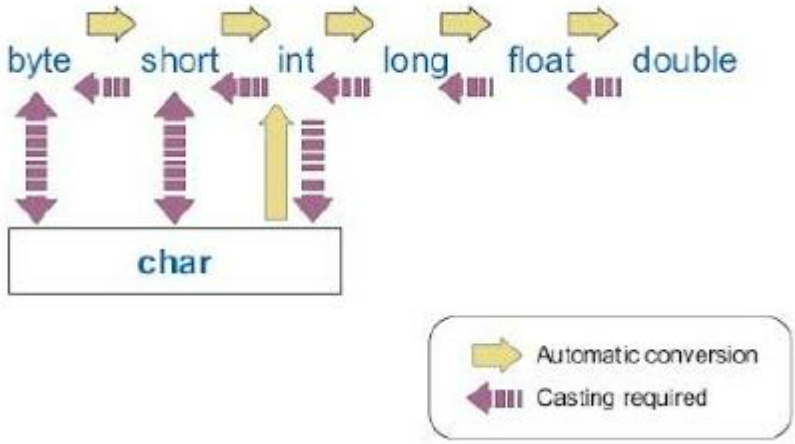
In case of explicit type conversion, the data gets converted to another type as per the user's choice and requirement.

e.g.

```
double  d = 10;
int i;
i  = (int) d;
```

Type Cast Operator

## Hierarchy of Data Types:




## Primitive Data Types With their Sizes:

**Primitive Data Types with their sizes and the ranges at a glance:**

| Data Type | Size | Range | Description |
|---|---|---|---|
| byte | 8 bits (1 byte) | –128 to +127 | Bit-wise operations |
| short | 16 bits (2 bytes) | –32768 to +32767 | To represent short integers |
| int | 32 bits (4 bytes) | –231 to 231 – 1 | To represent integers |
| long | 64 bits (8 bytes) | –263 to 263 –1 | To represent long integers |
| float | 32 bits (4 bytes) | –3.4 E+38 to 3.4E+38 | To represent values up to 6 significant digits |
| double | 64 bits (8 bytes) | –1.7E +308 to 1.7E+308 | To represent double value up to 15 significant digits |

**Answer the following questions in your copy….**

1. Define variable with an example.
2. What is literal? Explain briefly.
3. What is the difference between Unicode and ASCII.
4. What do you understand by token? Name different types of tokens.

5. What do you mean by type conversion. How is implicit conversion different from explicit conversion.
6. What is the significance of dot (.) operator in java.
7. What is the difference between true and “true”?
8. Distinguish between:

    i) Integer and floating constant

    ii) String and character constant

    iii) Static and dynamic initialization.

    iv) Floating point literal and double type literal
9. What are the resultant data types if the following implicit conversion are performed? Show the result with flow lines.

    int i; float f; double d; char c; byte b;

    a) i + c/b; b) f/d + c*f; c) i + f – c + b/d;
10. State the value of n and c:

    int x= ‘A’;

    int n = x + 32;

    char c = (char) n;

**Book**: Understanding Computer Applications with BlueJ ICSE Class IX

**Chapter**: 3 (Values an Data Types)

**Teacher**: Mr.Sabyasachi Mukherjee

**Date**: 22/05/2020