* Java
  + Object oriented programming

Everything in java is an object and hence can be extended

* + Platform independent

Java is platform independent because during the compilation the written code is converted into an intermediate code known as byte code. Which is converted into the respective format supported by the operating system by JVM installed.

* Data Types
  + Primitive Data types
    - Int

This data type is used to initialize integer data. Eg: int a;

Size: 4 bytes

* + - Long

This data type is used instead of int data type if more memory is required to store the integer data. Eg: long a;

Size: 8 bytes

* + - Double

Double is used to initialize bigger floating point numbers.

Size: 8 bytes

* + - Float

Float is used to initialize floating point numbers of smaller size

Size: 4 bytes.

* + - Byte

Byte data type is used to initialize smaller integer data of size 1 byte(8 bits)

* + - Boolean

Boolean initializes true or false value

Size: 1 byte

* + - Char

It is used to initialize character data types

* + Non primitive data types
    - String

It is used to initialize string data types

* Looping statements
  + For loop

It is used to repeat a block of instructions n number of times

Syntax:

for(data type variable ;condition ;increment){

Statements;

}

* + Advanced for loop

Modified version of for loop. Also used to access array elements in much more easy way.

Syntax:

for(data type variable : array\_name){

statements;

}

* + While loop

Same purpose of for loop but used in different scenarios depending upon the problem statement

Syntax:

Initialization;

While(condition){

Statements;

Increment;

}

* + Do while loop

This statements in this loop executes at least once

Syntax:

Initialization;

do{

statements;

increment;

}while(condition);

* + Break

It is used to break out of the loop

Syntax:

for(int i=0;i<n;i++){

If(condition){

Statements;

break;

}

}

* + Continue

It continues the iteration of the loop by skipping the remaining statements comes underneath it

Syntax:

for(int i=0;i<n;i++){

If(condition){

Statements;

continue;

statements;

}

}

* + Labelled break

Same function of break but instead it breaks towards the label.

Syntax:

First:

for(int i=0;i<n;i++){

If(condition){

Statements;

break First;

}

}

* + Labelled continue

Same function of continue but continue towards the label.

Syntax:

First:

for(int i=0;i<n;i++){

If(condition){

Statements;

continue First;

statements;

}

}

* Conditional statements
  + If

The statements inside the if block is executed if the condition is true.

Syntax:

if(condition){

Statements;

}

* + If else

The statements inside the if block is executed if the condition is true if not the statements inside the else block is executed

Syntax:

if(condition){

Statements;

}else {

Statements;

}

* + If else if

It is used to check multiple conditions

Syntax:

If(condition){

Statements;

}else if {

Statements;

}

.

.

.

}else if{

Statements;

}else{

Statements;

}

* Operators
  + Increment(a++,++a)
  + Decrement(a--,--a)
  + Less than(<)
  + Equal to(==)
  + Greater than(>)
  + Not equal to(!=)
  + And(&&)
  + Or(||)
* Arrays

Arrays are used to store a collection of data of same data type.

Arrays are initialized as follows:

int[] array\_name;

String[] array\_name;

array\_name.length gives size of the array

* OOPS
  + Data abstraction

Abstraction in oops is showing only the necessary information and hiding unwanted information

* + Data encapsulation

Encapsulation is the bundling of data and methods that works on the data within one unit

* + Inheritance

Inheritance is a child class can inherit the methods and attributes of the parent class. It can be single level or multiple level inheritance

* + Polymorphism

Polymorphism is in which something can take multiple forms. Overloading and overriding are some of the examples of polymorphism

* + Constructors

Constructor is a method that is called when a instance of the object is created.

* + Overloading
    - Method overloading

Method overloading happens when more than one method of same name exists with different number of parameters or different types of parameters.

* + Overriding
    - Method overriding

When a method in a child class is called which has same name as that of a method in parent class. The method in parent class is overridden by the method in child class

* + Class variables

These are the variables that are set for the classes. The value of these variables is mirrored in every instance created with theses classes. Any change made to them even in the one of the instances is reflected in every other instances. static keyword is used to declare them.

* + Instance variables.

Instance variables are as the name suggest variables initialized when the instance is created. These variables are limited to their instances.

* + This

this keyword is used to access the variables within a class

* + Super

super keyword is used to access methods of parent class

* Access modifiers
  + Default

Declarations are only visible inside the packages

* + Private

Declarations are only visible inside the classes

* + Protected

Declarations are visible within the package or subclasses

* + Public

Declarations are visible everywhere