**If**

**If else**

**Nested if**

**SWITCH STATEMENT**

1.SWITCH statement is used for one to one mapping and not for any conditions

2. the default statement of switch can be mentioned anywhere, if we put first or in between cases we should use break statement

3.if default statement is put last , not mandatory to use break statement

4.’BREAK’ keyword exits from the case

5. we can mention single output statement for multiple cases without break

**switch**(age) {

**case** 18 :

**case** 19:

**case** 20:

**case** 50:

**case** 60:

System.***out***.println("your are eligible for driving license ");

**break**;

**case** 15:

**case** 16:

**case** 17:

System.***out***.println(" your are eleigible for learner license");

**FOR LOOP**

**1.simple for loop**

1. Initialization only once
2. verify condition
3. execute statements
4. increment/decrement
5. repeat step b and then c followed by d

**2.advanced for loop – arrays and collections**

**1.for(;;) --- This is infinite loop**

2.for**(int i=0;i>=10;i--) This will end up infinite loop**

**WHILE LOOP : entry check**

**This will end up infinite loop**

**int** i= 1;

**while**(i<=10)

{

System.***out***.println("hello");

}

**DO WHILE LOOP : EXIT check**

**int** i= 10;

do

{

System.***out***.println("hello");

} **while**(i<=10)

OUTPUT: the print statement is executed only once

**NOTE:**

Diff between while and do while and for loop

For while and do while loop below statements are mentioned in 3 different lines where as in for loop **all below statements are written in same line**

1. Initialization
2. Conditions
3. Increment/decrement

**WHICH LOOP TO PREFER TO USE**

1.If we know the number if iteration in advance, the for loop is often most suitable

2.if we want to execute code atleast once use do –while loop

3.if we are uncertain about the number of iterations and loop termination is based on conditions, while loop is good choice

**RELATING EXAMPLE**

BIKE – source, destination, fuel -- FOR LOOP

BUS –GET INTO BUS THEN TICKET – DO WHILE

FLIGHT – FIRST BOOK TICKET CONDITION, WHILE LOOP

**JUMPING STATEMENTS**

break –conditional and looping statement

NOTE: statement mentioned after break, will be syntax error or unreachable code

continue – mostly used in looping statements

STATIC/NON STATIC

1. NON STATIC members are accessed using objects

**CLASS**

TEST t1= new TEST();

1. One java file can have only one public class
2. Here **TEST() is the constructor** of the class
3. Constructor

* is used to create a new object of the class
* to initialize non static variables
* name of constructor is same as class name
* constructor does not have return value
* for every object constructor will be called
* constructor does not have return type
* we can have more than one constructor for a class with params / without params
* constructor overloading – type, number of arguments and order of params
* **TEST()**
* **TEST(int a , int b, char c)**
* **TEST(int a , int b)**
* **TEST(float a , int b, char c)**
* **TEST(float a , char b, char c)**

1. **method name** can also same as **class name**

INHERITANCE- Inheriting

1. SIMPLE
2. MULTI LEVEL
3. HIERARCHICAL –one parent two child
4. MULTIPLE – 2 parent and one child

NOTE

THIS

1. Its used to refer global variable of class
2. This refers to current class object

Package

1. Used to avoid name conflict
2. Same class name can be put in different package but cant be inside same package

ACCESS MODIFIER

1.PRIVATE- ONLY IN CLASS

2.DEFAULT- inside all classes of same JAVA file

3.Protected- inside all classes of same PACKAGE file

4.PUBLIC- inside all classes of ALL PACKAGE file

NOTE: CLASS cant be private or default, it can be only public, abstract or final

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **private** | **default** | **Protected** | **public** |
| Same class |  |  |  |  |
| Different class same java file |  |  |  |  |
| Different class same package file |  |  |  |  |
| Different class different package file |  |  |  |  |
|  |  |  |  |  |

== : compare values of primitive data type

==: compare address of object

**Exception**

**Throw**

**Throwable**

**Throws**

|  |  |
| --- | --- |
| catch | The "catch" block is used to handle the exception. It must be preceded by try block which means we can't use catch block alone. It can be followed by finally block later. |
| finally | The "finally" block is used to execute the necessary code of the program. It is executed whether an exception is handled or not. |
| throw | The "throw" keyword is used to throw an exception. |
| throws | The "throws" keyword is used to declare exceptions. It specifies that there may occur an exception in the method. It doesn't throw an exception. It is always used with method signature. |

COLLECTION

1.LIST

2.QUEU

LIST

1.ARRAYLIST:can hold any type of data, all values are considered as object

ArrayList a1= new ArrayList();

Here data type is not mentioned hence can hold all types of data

a1.add(10);

a1.add(10.89);

a1.add(“hello”);

a1.add(‘b’);

get – to retrieve data

String s1= a1.get(2);//index 2 has string so we should use string type and explicitly cadt to string type

String s1= (String)a1.get(2)

If a1 is declared as String then this explicit string typecasting is not required

**File handling**

1.File class just creates a file if file does not exist, but it does not create folder.

2.