**Core Java Notes**

**Java:**

Java is a **programming language** and a **platform**. Java is a high level, robust, object-oriented and secure programming language.

Java was developed by Sun Microsystems (which is now the subsidiary of Oracle) in the year 1995. James Gosling is known as the father of Java. Before Java, its name was Oak. Since Oak was already a registered company, so James Gosling and his team changed the name from Oak to Java.

**Java Virtual machine:**

JVM (Java Virtual Machine) is an abstract machine. It is called a virtual machine because it doesn't physically exist. It is a specification that provides a runtime environment in which Java bytecode can be executed. It can also run those programs which are written in other languages and compiled to Java bytecode.

**Java Development Kit:**

The Java Development Kit (JDK) is a software development environment which is used to develop Java applications and applets. It physically exists. It contains JRE + development tools.

**Java Runtime Environment:**

The Java Runtime Environment is a set of software tools which are used for developing Java applications. It is used to provide the runtime environment. It is the implementation of JVM. It physically exists. It contains a set of libraries + other files that JVM uses at runtime.

**Object Oriented Programming:**

**Object-Oriented Programming** is a methodology or paradigm to design a program using classes and objects. It simplifies software development and maintenance .

**Object:**

Any entity that has state and behaviour is known as an object. An Object can be defined as an instance of a class.

**Class:**

Collection of objects is called class. It is a logical entity.

A class can also be defined as a blueprint from which you can create an individual object.

**Inheritance:**

When one object acquires all the properties and behaviours of a parent object, it is known as inheritance. It provides code reusability.

**Polymorphism:**If one task is performed in different ways, it is known as polymorphism. we use method overloading and method overriding to achieve polymorphism.

**Abstraction:**

Hiding internal details and showing functionality is known as abstraction.

**Encapsulation:**

Binding (or wrapping) code and data together into a single unit are known as encapsulation.

**Interface:**

An **interface in Java** is a blueprint of a class. The interface in Java is a mechanism to achieve abstraction. There can be only abstract methods in the Java interface, not method body.

**Abstract class:**

A class which is declared with the abstract keyword is known as an abstract class in java. It can have abstract and non-abstract methods (method with the body).

**Exception handling:**

The Exception Handling in Java is one of the powerful mechanism *to handle the runtime errors* so that the normal flow of the application can be maintained.

**Exception:**

In Java, an exception is an event that disrupts the normal flow of the program.  Exception is an abnormal condition.

**Checked Exception:**

Compile time exception.

**Unchecked Exception:**

Runtime exception.

**Try:**

The "try" keyword is used to specify a block where we should place an exception code.

**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.

**Finally:**

The "finally" block is used to execute the necessary code of the program. It is executed whether an exception is handled or not.

**Concurrency:**

Concurrency refers to the parallel processing. It is the ability to execute more than one programs or more than one parts of the program in parallel without affecting the final outcome.

**Thread:**

A thread is a lightweight process. Thread uses process’s execution environment like memory area. Context switch time is less in case of threads because switch is done within the same process’s memory area.

If your class is intended to be executed as a thread then you can achieve this by implementing a Runnable interface.

 you need to implement a run() method provided by a Runnable interface.

Once a Thread object is created, you can start it by calling start() method.

**Generics:**

Generics means parameterized types. The idea is to allow type to be a parameter to methods, classes, and interfaces. Using Generics, it is possible to create classes that work with different data types.

**JDBC:**

JDBC stands for Java Database Connectivity. JDBC is a Java API to connect and execute the query with the database. JDBC API uses JDBC drivers to connect with the database.

The **forName()** method of Class is used to register the driver class. This method is used to dynamically load the driver class.

The getConnection() method of DriverManager class is used to establish connection with the database.

|  |
| --- |
| The createStatement() method of Connection interface is used to create statement.  The object of statement is responsible to execute queries with the database.  The executeQuery() method of Statement interface is used to execute queries to the database. |