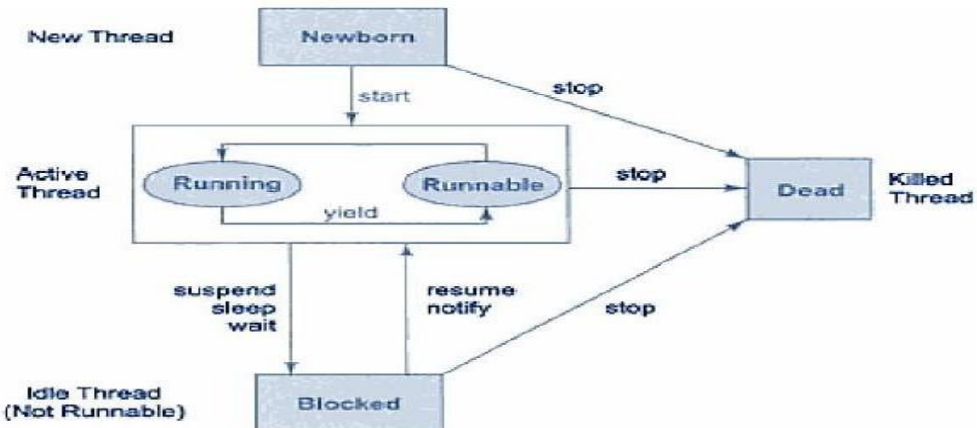


UNIT III EXCEPTION HANDLING AND MULTITHREADING Exception Handling basics - Multiple catch Clauses- Nested try Statements - Java's built-in exceptions-User defined exception. Multithreaded Programming: Java Thread Model, Creating a thread and multiple threads- Priorities- Synchronization- Inter-Thread communication-Suspending-Resuming and Stopping Threads- Multithreading. Wrappers- Auto boxing.											
UNIT-III/ PART-A											
1	What are the types of errors? • Compile time errors • Run time errors										
2	Define Java Exception. An Exception is an event that occurs during program execution which disrupts the normal flow of a program. It is an object which is thrown at runtime.										
3	State the five keywords in exception handling. Java exception handling is managed via five keywords: try, catch, throw, throws, and finally.										
4	Draw the exception hierarchy. The top-level exception hierarchy is shown here: <pre> graph BT Throwable[Throwable] --> Error[Error] Throwable --> Exception[Exception] Exception --> RuntimeException[RuntimeException] Exception --> IOException[IOException] RuntimeException --> NullPointerException[NullPointerException] RuntimeException --> NumberFormatException[NumberFormatException] RuntimeException --> ClassCastException[ClassCastException] RuntimeException --> IndexOutOfBoundsException[IndexOutOfBoundsException] IOException --> SQLException[SQLException] IOException --> MalformedURLException[MalformedURLException] </pre>										
5	Name any four-java built in exceptions. <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: center;">Exception</th><th style="text-align: center;">Meaning</th></tr> </thead> <tbody> <tr> <td>Arithmetic Exception</td><td>Arithmetic error, such as divide-by-zero.</td></tr> <tr> <td>ArrayIndexOutOfBoundsException</td><td>Arithmetic Exception Array index is out-of-bounds.</td></tr> <tr> <td>ArrayStoreException</td><td>Assignment to an array element of an incompatible type.</td></tr> <tr> <td>ClassCastException</td><td>Invalid cast</td></tr> </tbody> </table>	Exception	Meaning	Arithmetic Exception	Arithmetic error, such as divide-by-zero.	ArrayIndexOutOfBoundsException	Arithmetic Exception Array index is out-of-bounds.	ArrayStoreException	Assignment to an array element of an incompatible type.	ClassCastException	Invalid cast
Exception	Meaning										
Arithmetic Exception	Arithmetic error, such as divide-by-zero.										
ArrayIndexOutOfBoundsException	Arithmetic Exception Array index is out-of-bounds.										
ArrayStoreException	Assignment to an array element of an incompatible type.										
ClassCastException	Invalid cast										
6	What is chained exception? Chained Exceptions allows to relate one exception with another exception, i.e one exception describes cause of another exception. For example, consider a situation in which a method throws an Arithmetic Exception because of an attempt to divide by zero but the actual cause of exception was an I/O error which caused the divisor to be zero.										
6	What does java.lang.StackTraceElement represent? The java.lang.StackTraceElement class element represents a single stack frame. All stack frames except for the one at the top of the stack represent a method invocation. The frame at the top of the stack represents the execution point at which the stack trace was generated.										
7	What are the useful methods of throwable classes • public String getMessage()										

	• public synchronized Throwable getCause() • public String toString() • public void printStackTrace()
8	Compare throw and throws. • Throw is used to throw an exception & throws is used to declare an exception. • Throw is used in method implementation & throws is used in method signature. • Using throw keyword, we can throw only 1 exception at a time & throws can declare multiple exceptions at a time.
9	What is the use of try and catch exception? Try-catch block is used for exception handling in the program code. try is the start of the block and catch is at the end of try block to handle the exceptions. A Program can have multiple catch blocks with a try and try-catch block can be nested also. catch block requires a parameter that should be of type Exception.
10	What is the use of finally exception? Finally block is optional and can be used only with try-catch block. Since exception halts the process of execution, we might have some resources open that will not get closed, so we can use finally block. finally block gets executed always, whether exception occurs or not.
11	What is OutOfMemoryError in Java? OutOfMemoryError in Java is a subclass of java.lang.VirtualMachineError and it's thrown by JVM when it ran out of heap memory.
12	What is difference between final, finally and finalize in Java? Final and finally are keywords in java whereas finalize is a method. Final keyword can be used with class variables so that they can't be reassigned, with class to avoid extending by classes and with methods to avoid overriding by subclasses. Finally keyword is used with try-catch block to provide statements that will always get executed even if some exception arises, usually finally is used to close resources. finalize() method is executed by Garbage Collector before the object is destroyed, it's a great way to make sure all the global resources are closed. Out of the three, only finally is related to java exception handling.
13	What happens when exception is thrown by main method? When exception is thrown by main() method, Java Runtime terminates the program and prints the exception message and stack trace in system console.
14	Can we have an empty catch block? We can have an empty catch block but it's the example of worst programming. We should never have an empty catch block because if the exception is caught by that block, we will have no information about the exception and it will be a nightmare to debug it.
15	How Java Exception Hierarchy is categorized? Java Exceptions are hierarchical and inheritance is used to categorize different types of exceptions. Throwable is the parent class of Java Exceptions Hierarchy and it has two child objects – Error and Exception. Exceptions are further divided into checked exceptions and runtime exception.

16	How Threads are created in Java? Threads are created in two ways. They are by extending the Thread class and by implementing Runnable interface.	
17	Define Thread? A thread is a single sequential flow of control within program. Sometimes, it is called an execution context or light weight process. A thread itself is not a program. A thread cannot run on its own. Rather, it runs within a program. A program can be divided into a number of packets of code, each representing a thread having its own separate flow of control.	
18	What is Multi-threading? Multithreading is a conceptual programming concept where a program(process) is divided into two or more subprograms(process), which can be implemented at the same time in parallel. A multithreaded program contains two or more parts that can run concurrently. Each part of such a program is called a thread, and each thread defines a separate path of execution.	
19	What is meant by Multitasking? Multitasking , in an operating system, is allowing a user to perform more than one computer task (such as the operation of an application program) at a time. The operating system is able to keep track of where you are in these tasks and go from one to the other without losing information. Multitasking is also known as multiprocessing.	
20	Difference between multi-threading and multi-tasking?	
	Multi-threading In any single process, multiple threads is allowed and again, can run simultaneously.	Multi-tasking It refers to having multiple (programs, processes, tasks, threads) running at the same time.
	It is sharing of computing resources among threads of a single process.	It is sharing of computing resources (CPU, memory, devices, etc.) among processes
21	What do you mean by Thread Scheduling? Execution of multiple threads on a single CPU in some order is called scheduling. The Java runtime environment supports a very simple, deterministic scheduling algorithm called fixed-priority scheduling. This algorithm schedules threads on the basis of their priority relative to other Runnable threads.	
22	What is Thread Pool? A thread pool is a managed collection of threads that are available to perform tasks. Thread pools usually provide: • Improved performance when executing large numbers of tasks due to reduce per-task invocation overhead • A means of bounding the resources, including threads, consumed when executing a collection of tasks.	
23	What is Synchronization thread? When two or more threads need access to a shared resource, they need some way to ensure that the resource will be used by only one thread at a time. The process by which this synchronization is achieved is called thread synchronization.	
24	What is thread priority? Every Java thread has a priority that helps the operating system determine the order in which threads are scheduled. Java priorities are in the range between MIN_PRIORITY (a constant of 1) and MAX_PRIORITY(a constant of 10). By default, every thread is given priority NORM_PRIORITY(a constant of 5) Threads with higher priority are more important to a program and should be allocated processor time before lower-priority threads. However, thread priorities cannot guarantee the order in which threads execute and very much platform independent.	

25	List out the methods of object class to perform inter thread communication? ➤ wait() – This method make the current thread to wait until another thread invokes the notify() method. ➤ notify() – This method wakes up a thread that called wait() on same object. ➤ notifyAll() – This method wakes up all the thread that called wait() on same object. Wakes up all threads that are waiting on this object's monitor. Above all three methods have been implemented as final method in Object class, so that they are available in all the classes in java world.
26	What are the various states of a thread? The following figure shows the states that a thread can be in during its life and illustrates which method calls cause a transition to another state.  <pre> graph TD Newborn[Newborn] -- start --> Running([Running]) Newborn -- stop --> Dead[Dead] Running -- yield --> Runnable([Runnable]) Runnable -- yield --> Running Running -- "suspend, sleep, wait" --> Blocked[Blocked] Blocked -- "resume, notify" --> Runnable Running -- stop --> Dead Blocked -- stop --> Dead subgraph Active_Thread [Active Thread] Running Runnable end style Newborn fill:#d9e1f2,stroke:#333,stroke-width:1px style Running fill:#d9e1f2,stroke:#333,stroke-width:1px style Runnable fill:#d9e1f2,stroke:#333,stroke-width:1px style Blocked fill:#d9e1f2,stroke:#333,stroke-width:1px style Dead fill:#d9e1f2,stroke:#333,stroke-width:1px </pre>
27	Why do we need run() and start() method both? Can we achieve it with only run method? • The separate start() and run() methods in the Thread class provide two ways to create threaded programs. The start() method starts the execution of the new thread and calls the run() method. The start() method returns immediately and the new thread normally continues until the run() method returns. • The Thread class' run() method does nothing, so sub-classes should override the method with code to execute in the second thread. If a Thread is instantiated with a Runnable argument, the thread's run() method executes the run() method of the Runnable object in the new thread instead. • Depending on the nature of your threaded program, calling the Thread run() method directly can give the same output as calling via the start() method, but in the latter case the code is actually executed in a new thread.
28	Write short note on isAlive() and join(). isAlive() and join() methods are used to determine whether a thread has finished or not. First, you can call isAlive() on the thread. This method is defined by Thread, and its general form is: <i>final Boolean isAlive()</i> The isAlive() method returns true if the thread upon which it is called is still running. It returns false otherwise. While isAlive() is occasionally useful, the method that you will more commonly use to wait for a thread to finish is called join(). The general form is: <i>final void join() throws InterruptedException</i> This method waits until the thread on which it is called terminates
29	Define Suspend(). The suspend () method of thread class puts the thread from running to waiting state. This method is used if you want to stop the thread execution and start it again when a certain event occurs. This method allows a thread to temporarily cease execution. The suspended thread can be resumed using the resume() method. public final void suspend()

30	Define Resume(). The resume() method of thread class is only used with suspend() method. This method is used to resume a thread which was suspended using suspend() method. This method allows the suspended thread to start again. public final void resume()
31	What is Wrapper class? A Wrapper class in Java is a class whose object wraps or contains primitive data types. When we create an object to a wrapper class, it contains a field and, in this field, we can store primitive data types.
32	What is Autoboxing and Unboxing? Autoboxing: The automatic conversion of primitive types to the object of their corresponding wrapper classes is known as autoboxing. For example – conversion of int to Integer, long to Long, double to Double, etc. Unboxing: It is just the reverse process of autoboxing. Automatically converting an object of a wrapper class to its corresponding primitive type is known as unboxing. For example – conversion of Integer to int, Long to long, Double to double, etc.
33	Define Arithmetic Exception with example ArithmeticException: An arithmetic exception in java is a Runtime exception . JVM throws Arithmetic Exception when a wrong mathematical expression occurs in a java program. such as divide by zero <pre> class Simple { public static void main(String args[]) {int data=50/0; System.out.println("rest of the code..."); } } </pre> Output: Exception in thread main java.lang.ArithmeticException:/ by zero
34	"Thread is a lightweight process" – Comment Threads do not require separate address for its execution. It runs in the address space of the process to which it belongs to. Hence thread is a lightweight process.
35	What are the three ways by which the thread can enter in waiting stage? Waiting state: Sometimes one thread has to undergo in waiting state because another thread starts executing. This can be achieved by using wait() state. ii) Timed waiting state: There is a provision to keep a particular threading waiting for some time interval. This allows to execute high prioritized threads first. After the timing gets over, the thread in waiting state enters in runnable state. This can be achieved by using the sleep() command. iii) Blocked state : When a particular thread issues an input/output request then operating system sends it in blocked state until the I/O operation gets completed. After the I/O completion the thread is sent back to the runnable state. This can be achieved by using suspend() method.

	Unit –III/Part B
1.	Explain in detail the important methods of Java Exception Class.
2.	Explain the different scenarios causing “Exception in thread main”.
3.	How will you create your Own Exception? (user defined Exception)
4.	Explain in detail the various exception types with its hierarchy.
5.	Write programs to illustrate arithmetic exception, ArrayIndexOutOfBoundsException Exception and NumberFormat Exception.
6.	Write a calculator program using exceptions.
7.	Create two exception classes that can be used by the stack classes developed by TRY.
8.	What are the two ways of thread creation? Explain with suitable examples.
9.	With illustrations explain multithreading, interrupting threads, thread states and thread properties.
10.	Describe the life cycle of thread and various thread methods.
11.	Explain in detail about suspending, resuming and stopping threads.
12.	Write a java program that synchronizes three different threads of the same program and displays the contents of the text supplies through the threads.
13.	Write a java program for inventory problem to illustrate the usage of thread synchronized keyword and inter thread communication process. They have three classes called consumer, producer and stock.
14.	i)What is the purpose of thread priorities? What is the different thread priorities that exist? ii)Difference between throw and throws. Give example for both.
15.	i)Describe the criteria for creating single and multithread using an example. ii)Using an example, explain inter-thread communication in java.
16.	Create a software for departmental stores to maintain the following details like item no, item description , requested quantity, cost price. Provide the options to update the stock. Calculate the selling price ($SP = CP * 20\%$). Create an exception whenever the selling price of item exceeds the given amount
17.	Discuss about try , catch and finally keywords in Exception handling with example
18.	List and explain datatypes and their corresponding Wrapper class
19.	Explain Exception hierarchy in detail
20.	Write about pre-defined Exceptions (Built in Exceptions)