PRACTICAL : 15

PAGE No: ____

Aim: Create, debug and run Java program based on threads by implementing runnable interfrance.

Theory:

What are Java Threads?

A thread is a:

- Facility to allow multiple activities within a single process
- Referred as light weight process.
- A thread is a series of executed statement
- Each thread has it's own program counter, stack and local variables.
- A thread is a nested sequence of methods calls.
- It share memory, fires and per-process state.

Every Java program creates at least one thread [main() thread]. Additional thread are created through the Thread constructor or by instantiating classes that extend the Thread class.

Thread implementation in Java can be achieved in two ways:

DATE: __/__/__ PAGE No: ____

Runnable interface

Java runnable is an interface used to execute code on a Concurrent thread It is an interface which is implemented by any class if we want that that the instances of that class should sho be executed by a thread.

public void run() :- This method takes inno argument. when the object of a class implementing Runnable class is used to Create a thread, then the run method is invoked in The run thread which executes Sep.anetely.

A class that implements Runnable run on a different thread without Subclassing. Thread as it instantiates a Thread instance and passes itself in an the tenget. This become important as classes should not be Subclassed unless there is an intention of modifying or enhaving the fundamental behavior of the class.

DATE : __/__/__ PAGE No : ____

Runnable class is extensively used in Networ programming as each thread represent a separate flow of control. Also in multi-threaded programming, Runnable class is used. This interface is present in Java.lang.package.

Conclusion : Hence, we successfully create, debug and run Java program based on threads by implementing runnable interface.

Program:-

```
Class ChildThread implements Runnable {
        Thread t;
        ChildThread (String s) {
            t = new Thread (this, s);
                run ();
            System.out.println ("Constructor existing :- ");
    }
public void run (){
    String child = t.getName ();
  System.out.println ("Thread name : " + child );
      int i = 1;
    if (child.equals ("add"))
                i = 1;
    else if (child.equals ("Even"))
                i = 0;
     else return ;
   try {
       for (i<=10; i = i+2) {
     System.out.println ("Child thread : " + child + "->"
                                    +i);
          t.sleep (500);
       }
   }
Catch (Exception e) {}
    System.out.println (child + "Thread is exiting--");
}
```

PAGE :
DATE : / /

```
public class Assignment8 {
    public static void main (String args [] ) {
        String name = Thread. CurrentThread ().
                      getName ();
    ChildThread odd = new ChildThread ("odd");
    ChildThread even = new ChildThread ("even");
}
}
```

```
C:\Users\Public\Java>javac practical15.java

C:\Users\Public\Java>java practical15
Thread[main,5,main]
Thread : main 5
Thread Name :One
Child Thread :One->5
Child Thread :One->4
Thread : main 4
Child Thread :One->3
Child Thread :One->2
Thread : main 3
Child Thread :One->1
Child thread Exiting ...
Thread : main 2
```