 *DEPARTMENT OF INFORMATION TECHNOLOGY*

Experiment No.15

|  |  |  |
| --- | --- | --- |
| Experiment No | 15 | |
| Experiment Title | Write the programs based on multithreading in java. | |
| Resources / Apparatus Required | Java SE(JDK)8u102 ,  gedit text editor | PC |
| Objectives | The objective of this experiment is to learn programs based on multithreading in java. | |
| Theory | What is Thread in java? A thread is a lightweight sub process, a smallest unit of processing. It is a separate path of execution.Threads are independent, if there occurs exception in one thread, it doesn't affect other threads. It shares a common memory area.  C:\Users\Mahesh\Desktop\multithreading.JPG As shown in the above figure, thread is executed inside the process. There is context-switching between the threads. There can be multiple processes inside the OS and one process can have multiple threads.Note: At a time one thread is executed only.Multithreading in Java: Multithreadingin java is a process of executing multiple threads. Simultaneously , thread is basically a lightweight sub-process, a smallest unit of processing. Multiprocessing and multithreading, both are used to achieve multitasking. But we use multithreading than multiprocessing because threads share a common memory area. They don't allocate separate memory area so saves memory, and context-switching between the threads takes less time than process. Java Multithreading is mostly used in games, animation etc. Advantages of Java Multithreading: 1) It doesn't block the user because threads are independent and you can perform multiple operations at same time.  2) You can perform many operations together so it saves time.  3) Threads are independent so it doesn't affect other threads if exception occur in a single thread. Multitasking: Multitasking is a process of executing multiple tasks simultaneously. We use multitasking to utilize the CPU. Multitasking can be achieved by two ways:   * Process-based Multitasking(Multiprocessing) * Thread-based Multitasking(Multithreading)   **1) Process-based Multitasking (Multiprocessing):**  Each process have its own address in memory i.e. each process allocates separate memory area. Process is heavyweight. Cost of communication between the process is high. Switching from one process to another require some time for saving and loading registers, memory maps, updating lists etc. 2) Thread-based Multitasking (Multithreading) Threads share the same address space. Thread is light weight. Cost of communication between the thread is low. Note: At least one process is required for each thread. | |
| Program & output | A)  **/\* W.A.P using Thread to print numbers from 1 to 10 by extending the "Thread" Class \*/**  import java.io.\*; class Numbers extends Thread { public void run() { int i; for(i=1;i<=10;i++) { System.out.println(i); }  }  }  class ThreadExtends { public static void main(String args[]) { Numbers n=new Numbers(); Thread t1=new Thread(n); t1.start();  }}  C:\Users\Mahesh\Desktop\New folder (2)\4.png  B)  **/\* W.A.P using Thread to print numbers from 1 to 10 by implementing the "Runnable" Interface \*/**  import java.io.\*;class Numbers implements Runnable{ public void run()  { int i; for(i=1;i<=10;i++) { System.out.println(i); } }  } class ThreadImplements { public static void main(String args[])  { Numbers n=new Numbers();  Thread t1=new Thread(n); t1.start(); }  }  C:\Users\Mahesh\Desktop\New folder (2)\3.png | |
| Conclusion | Thus, we have learnt programs based on multithreading in java. | |