Ex.No - 06 Date - 03/09/2024

# Implementation of Round Robin Task Scheduling in Both Time Shared and Space Shared CPU

### AIM:

To implement the round robin task scheduling in both time shared and space shared CPU using CloudSim.

#### **PROCEDURE:**

- 1. Create a new project by selecting java console lineapplication template and JDK 18.
- **2.** Open project settings from the file menu of the optionswindow.
- **3.** Navigate to project dependencies and select on add externaljars and then click on 'Browse' to open the path where you have unzipped the Cloudsim Jars and click on apply.
- **4.** Create a java file with the cloudsim code to implement theround robin scheduling algorithm.
- **5.** Run the application as a java file to see the output in the console below.

#### CODE:

```
import org.cloudbus.cloudsim.*;
import org.cloudbus.cloudsim.core.CloudSim;import java.util.*;
public class RoundRobinScheduler { public static
    void main(String[] args) {
```

```
try {
       int numUser = 1; // number of cloud users Calendar calendar =
       Calendar.getInstance(); boolean traceFlag = false; // mean trace
       events
       CloudSim.init(numUser, calendar, traceFlag);Datacenter
       datacenter0=
createDatacenter("Datacenter_0");
       DatacenterBroker broker = createBroker();int brokerId =
       broker.getId();
       List<Vm> vmList = new ArrayList<>();int vmId =
       0;
       int mips = 1000;
       long size = 10000; // image size (MB)int ram =
       512; // vm memory (MB) long bw = 1000;
       int pesNumber = 1; // number of CPUsString
       vmm = "Xen"; // VMM name
       for (int i = 0; i < 3; i++) {
         vmList.add(new Vm(vmId++, brokerId, mips, pesNumber, ram,bw, size, vmm,
new CloudletSchedulerTimeShared()));
       broker.submitVmList(vmList);
```

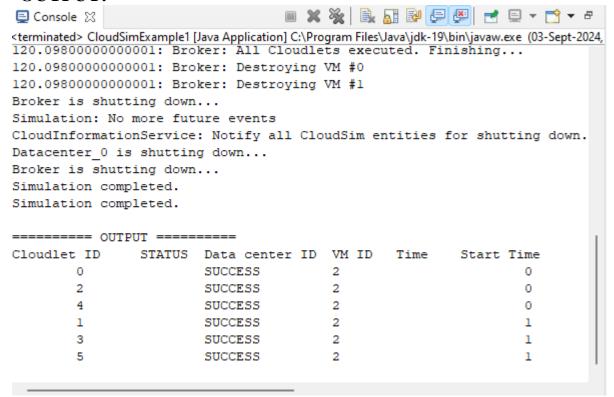
```
List<Cloudlet> cloudletList = new ArrayList<>();int cloudletId =
       0;
       long length = 40000; long
       fileSize = 300; long
       outputSize = 300;
       UtilizationModel utilizationModel = new UtilizationModelFull();
       for (int i = 0; i < 6; i++) {
         Cloudlet cloudlet = new Cloudlet(cloudletId++, length, pesNumber, fileSize,
outputSize, utilizationModel, utilizationModel, utilizationModel);
         cloudlet.setUserId(brokerId);
         cloudletList.add(cloudlet);
       broker.submitCloudletList(cloudletList); CloudSim.startSimulation();
       List<Cloudlet> newList = broker.getCloudletReceivedList();
       CloudSim.stopSimulation();
       printCloudletList(newList);
     } catch (Exception e) {
       e.printStackTrace();
```

```
private static Datacenter createDatacenter(String name) {List<Host> hostList =
    new ArrayList<>();
    int mips = 1000;
    int ram = 2048; // host memory (MB) long storage =
    1000000; // host storageint bw = 10000;
    for (int i = 0; i < 2; i++) {
       List<Pe> peList = new ArrayList<>();
       peList.add(new Pe(0, new PeProvisionerSimple(mips)));
       hostList.add(new Host(i, new RamProvisionerSimple(ram),new
BwProvisionerSimple(bw), storage, peList, new
VmSchedulerTimeShared(peList)));
    String arch = "x86"; String os =
    "Linux"; String vmm = "Xen";
    double time_zone = 10.0;double
    cost = 3.0;
    double costPerMem = 0.05; double
    costPerStorage = 0.001; double
    costPerBw = 0.0;
```

DatacenterCharacteristics characteristics = new
DatacenterCharacteristics(arch, os, vmm, hostList, time\_zone,cost,costPerMem,
costPerStorage,costPerBw);

```
Datacenter datacenter = null;try {
       datacenter = new Datacenter(name, characteristics, new
VmAllocationPolicySimple(hostList), new LinkedList<Storage>(), 0);
    } catch (Exception e) {
       e.printStackTrace();
    return datacenter;
                      DatacenterBroker
                                            createBroker()
  private
             static
    DatacenterBroker broker = null;
    try {
       broker = new DatacenterBroker("Broker");
     } catch (Exception e) {
       e.printStackTrace(); return
       null;
    return broker;
  private static void printCloudletList(List<Cloudlet> list) {String indent
    = "
    System.out.println();
```

## **OUTPUT:**



## **RESULT:**

Thus, to implement the round robin task scheduling using CloudSim is done successfully.