

Practical: 4

AIM- Installation and Configuration of Hosted Based Hypervisor.

Submitted By: Adeshara Brijesh
Enrollment number: 21012021001



**Ganpat
University**

॥ विद्यया समाजोत्कर्षः ॥

**U.V. Patel
College of
Engineering**

**Department of
Information Technology**

❖ Introduction To CLOUD SIM:

- Cloud Sim is an open-source framework, which is used to simulate cloud computing infrastructure and services. It is developed by the CLOUDS Lab organization and is written entirely in Java. It is used for modeling and simulating a cloud computing environment as a means for evaluating a hypothesis prior to software development in order to reproduce tests and results.
- For example, if you were to deploy an application or a website on the cloud and wanted to test the services and load that your product can handle and also tune its performance to overcome bottlenecks before risking deployment, then such evaluations could be performed by simply coding a simulation of that environment with the help of various flexible and scalable classes provided by the Cloud Sim package, free of cost.

❖ Benefits of Simulation over the Actual Deployment:

1. No capital investment involved. With a simulation tool like Cloud Sim there is no installation or maintenance cost.
2. Easy to use and Scalable. You can change the requirements such as adding or deleting resources by changing just a few lines of code.
3. Risks can be evaluated at an earlier stage. In Cloud Computing utilization of real testbeds limits the experiments to the scale of the testbed and makes the reproduction of results an extremely difficult undertaking. With simulation, you can test your product against test cases and resolve issues before actual deployment without any limitations.
4. No need for try-and-error approaches. Instead of relying on theoretical and imprecise evaluations which can lead to inefficient service performance and revenue generation, you can test your services in a repeatable and controlled environment free of cost with Cloud Sim.

❖ Why use Cloud Sim?

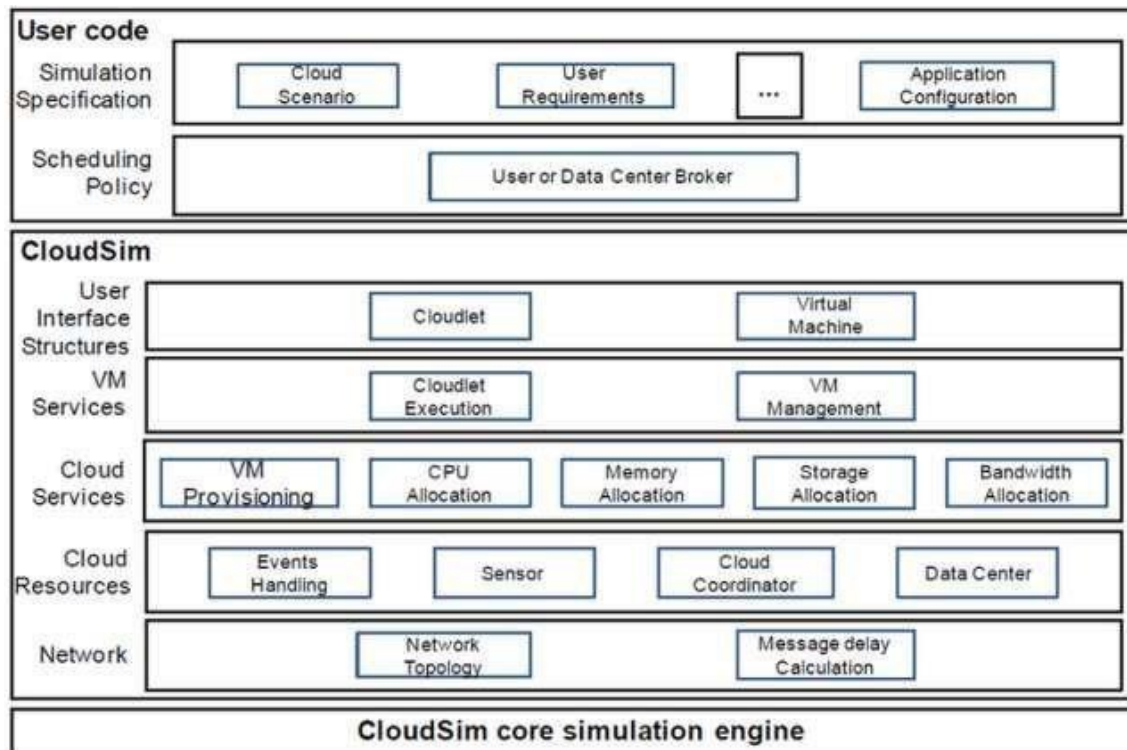
1. Open source and free of cost, so it favors researchers/developers working in the field.
2. Easy to download and set-up.
3. It is more generalized and extensible to support modeling and experimentation.
4. Does not require any high-specs computer to work on.
5. Provides pre-defined allocation policies and utilization models for managing resources, and allows implementation of user-defined algorithms as well.
6. The documentation provides pre-coded examples for new developers to get familiar with the

Practical: 4

basic classes and functions.

7. Tackle bottlenecks before deployment to reduce risk, lower costs, increase performance, and raise revenue.

❖ Cloud Sim Architecture:



- Cloud Sim Core Simulation Engine provides interfaces for the management of resources such as VM, memory and bandwidth of virtualized Datacenters.
- Cloud Sim layer manages the creation and execution of core entities such as VMs, Cloudlets, Hosts etc. It also handles network-related execution along with the provisioning of resources and their execution and management.
- User Code is the layer controlled by the user. The developer can write the requirements of the hardware specifications in this layer according to the scenario.

❖ Some of the most common classes used during simulation are:

- **Datacenter**: used for modelling the foundational hardware equipment of any cloud environment, that is the Datacenter. This class provides methods to specify the functional requirements of the Datacenter as well as methods to set the allocation policies of the VMs etc.
- **Host**: this class executes actions related to management of virtual machines. It also defines policies for provisioning memory and bandwidth to the virtual machines, as well as allocating CPU cores to the virtual machines.

Practical: 4

- VM: this class represents a virtual machine by providing data members defining a VM's bandwidth, RAM, mips (million instructions per second), size while also providing setter and getter methods for these parameters.
- Cloudlet: a cloudlet class represents any task that is run on a VM, like a processing task, or a memory access task, or a file updating task etc. It stores parameters defining the characteristics of a task such as its length, size, mi (million instructions) and provides methods similarly to VM class while also providing methods that define a task's execution time, status, cost and history.
- DatacenterBroker: is an entity acting on behalf of the user/customer. It is responsible for functioning of VMs, including VM creation, management, destruction and submission of cloudlets to the VM.
- Cloud Sim: this is the class responsible for initializing and starting the simulation environment after all the necessary cloud entities have been defined and later stopping after all the entities have been destroyed.

❖ Features of Cloud Sim:

1. Large scale virtualized Datacenters, servers and hosts.
2. Customizable policies for provisioning host to virtual machines.
3. Energy-aware computational resources.
4. Application containers and federated clouds (joining and management of multiple public clouds).
5. Datacenter network topologies and message-passing applications.
6. Dynamic insertion of simulation entities with stop and resume of simulation.
7. User-defined allocation and provisioning policies.

❖ Scope: -

With the flexibility and generalizability of the Cloud Sim framework, it is easy to model heavy cloud environments which would otherwise require experimentation on paid computing infrastructures. Extensible capabilities of scaling the infrastructure and resources to fit any scenario helps in fast and efficient research of several topics in cloud computing.

❖ Cloud Sim has been used in several areas of research such as:

1. Task Scheduling
2. Green Computing

3. Resource Provisioning

4. Secure Log Forensics.

❖ **Advantages of Cloud Sim:**

1. Cloud Sim provides a layered architecture that allows for the simulation of cloud computing infrastructure and services.
2. It has a flexible and scalable core simulation engine that can manage resources such as virtual machines, memory, and bandwidth.
3. Cloud Sim also includes support for modeling and simulating various entities such as virtual machines, cloudlets, and hosts.
4. It can be used to evaluate a hypothesis prior to software development and can help reproduce tests and results.
5. Cloud Sim is open-source and written in Java, making it a popular choice for researchers and developers in the field of cloud computing.

❖ **Disadvantages of Cloud Sim:**

1. Cloud Sim requires a good understanding of cloud computing concepts and simulation techniques to use effectively.
2. The framework can be complex to set up and use, with a steep learning curve for new users.
3. Cloud Sim simulations may not always accurately reflect real-world cloud computing environments, as they are based on simulations and assumptions.
4. Cloud Sim may not support all cloud computing technologies or platforms, which can limit its usefulness in some scenarios.
5. The accuracy and reliability of Cloud Sim simulations may depend on the quality and completeness of the input data and assumptions used in the simulation.

❖ **Versions of Cloud Sim:**

1. **Cloud Sim 1.0:**

- Released in 2009, this was the first version of Cloud Sim.
- It provided a basic simulation framework for cloud computing environments.
- It supported the simulation of data centers, virtual machines, and cloudlets.
- It used a simple allocation policy for resource allocation.
- It supported only a single data center and a single user.

2. Cloud Sim 1.1:

- Released in 2010, this version of Cloud Sim introduced several improvements over the previous version.
- It added support for multiple data centers and multiple users.
- It introduced a new resource allocation policy called Best Fit.
- It provided a more realistic simulation of cloud computing environments.
- It supported the simulation of data center failures and recovery.

3. Cloud Sim 1.2:

- Released in 2011, this version of Cloud Sim introduced several new features and improvements.
- It added support for data center failures and recovery.
- It introduced a new scheduling policy called Round Robin.
- It provided a more detailed simulation of cloud computing environments.
- It supported the simulation of multiple data centers and multiple users.

4. Cloud Sim 1.3:

- Released in 2012, this version of Cloud Sim introduced several new features and improvements.
- It added support for data center failures and recovery.
- It introduced a new scheduling policy called First Come First Served.
- It provided a more detailed simulation of cloud computing environments.
- It supported the simulation of multiple data centers and multiple users.

5. Cloud Sim 2.0:

- Released in 2013, this version of Cloud Sim introduced significant improvements over the previous versions.
- It added support for data center failures and recovery.
- It introduced a new scheduling policy called Least Loaded.
- It provided a more detailed simulation of cloud computing environments.
- It supported the simulation of multiple data centers and multiple users.
- It introduced a new architecture for the simulation framework, making it more flexible and scalable.

6. Cloud Sim 2.1:

- Released in 2014, this version of Cloud Sim introduced several new features and improvements.
- It added support for data center failures and recovery.
- It introduced a new scheduling policy called Most Loaded.
- It provided a more detailed simulation of cloud computing environments.
- It supported the simulation of multiple data centers and multiple users.
- It introduced a new architecture for the simulation framework, making it more flexible and scalable.

Practical: 4

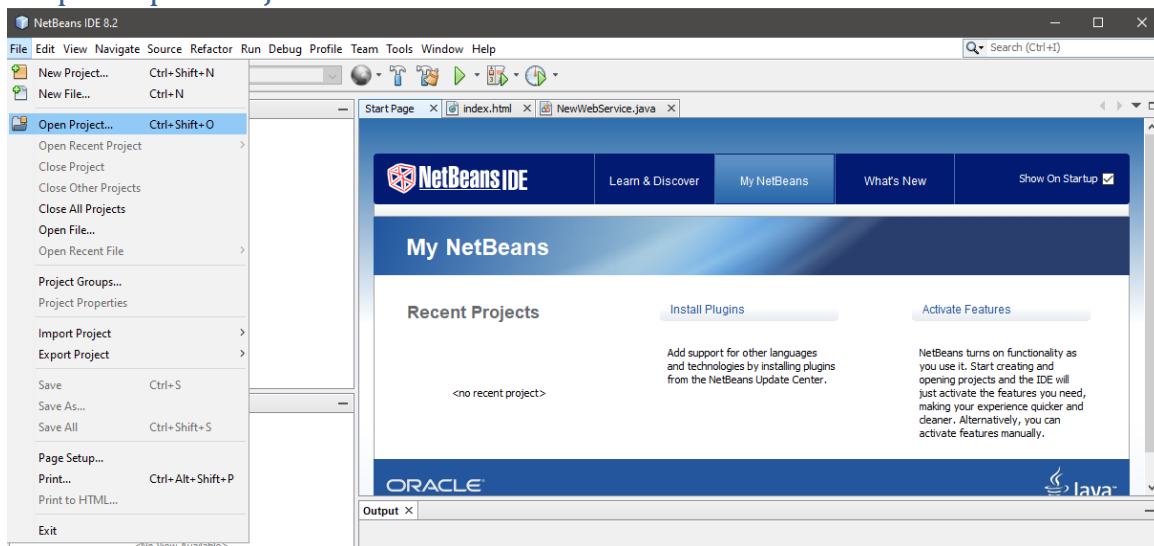
7. Cloud Sim 3.0:

- Released in 2015, this version of Cloud Sim introduced significant improvements over the previous versions.
- It added support for data center failures and recovery.
- It introduced a new scheduling policy called Random.
- It provided a more detailed simulation of cloud computing environments.
- It supported the simulation of multiple data centers and multiple users.
- It introduced a new architecture for the simulation framework, making it more flexible and scalable.

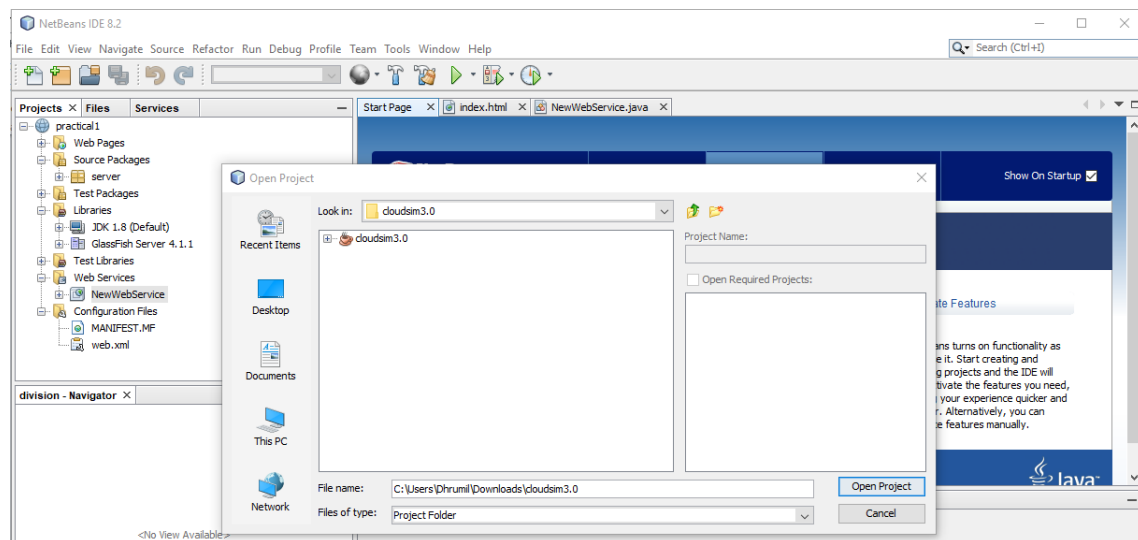
8. CLOUD SIM STEP BY STEP INSTALLATION USING NETBEANS:

Step 1: Open NetBeans Software

Step 2: Open Project from File Menu

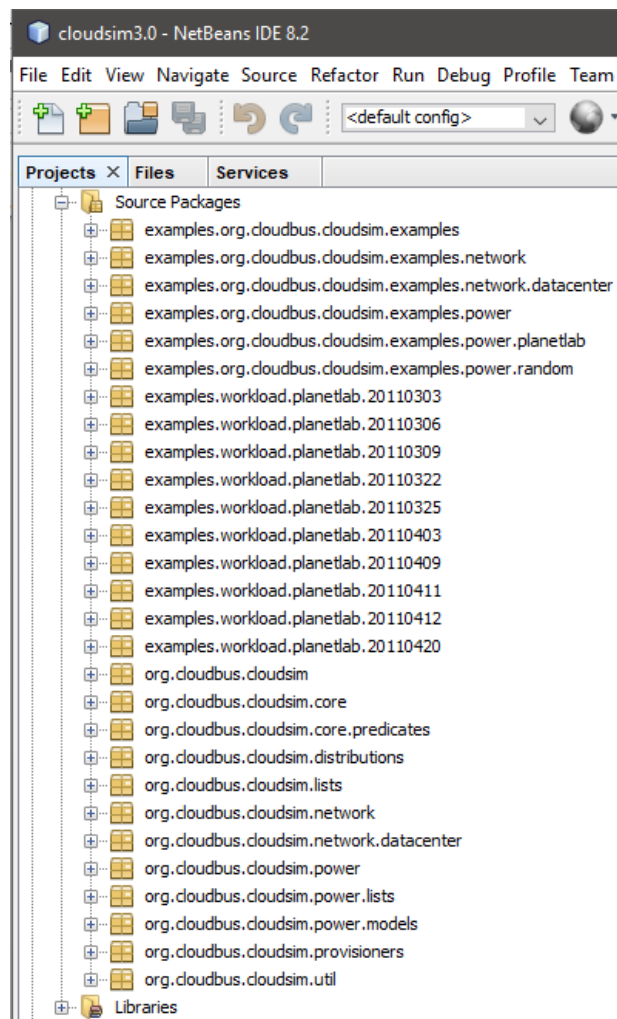


Step 3: Select The Cloud Sim Project and click on open



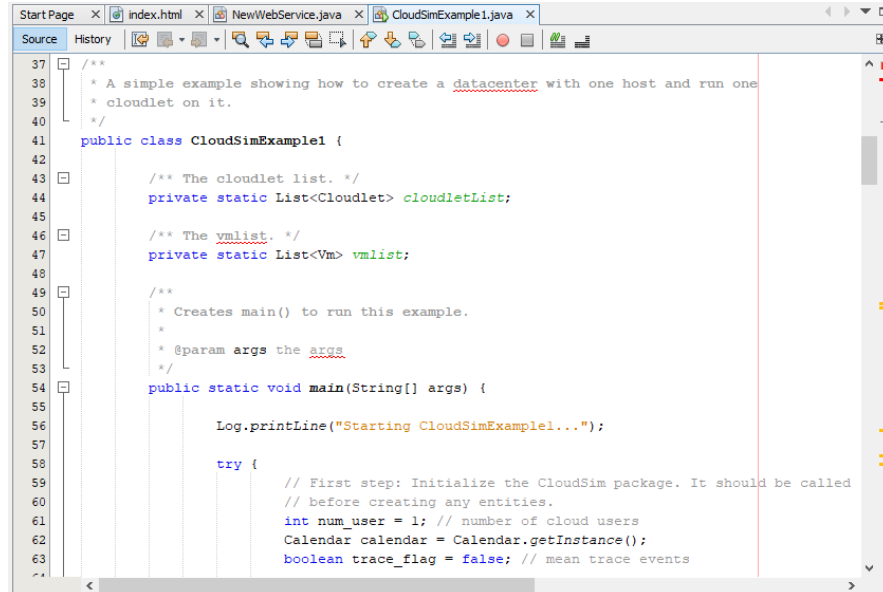
Practical: 4

Step 4: You will see the existing projects at left side



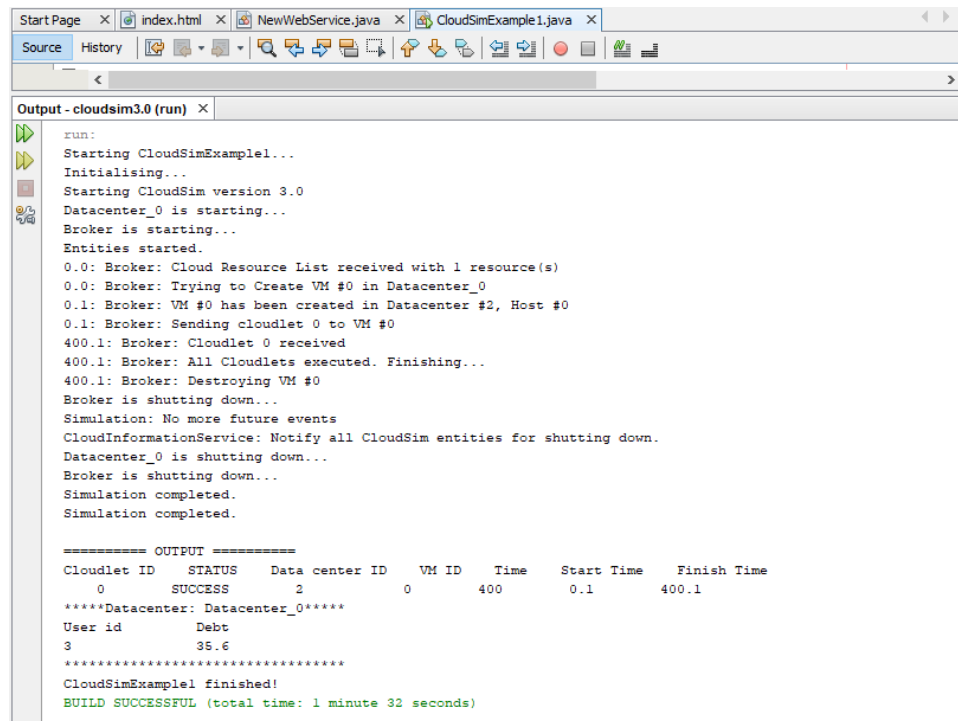
Practical: 4

Step 5: Open any project you want to run



```
37  /**
38   * A simple example showing how to create a datacenter with one host and run one
39   * cloudlet on it.
40   */
41  public class CloudSimExample1 {
42
43      /** The cloudlet list. */
44      private static List<Cloudlet> cloudletList;
45
46      /** The vmlist. */
47      private static List<Vm> vmList;
48
49      /**
50       * Creates main() to run this example.
51       *
52       * @param args the args
53       */
54      public static void main(String[] args) {
55
56          Log.println("Starting CloudSimExample1...");
57
58          try {
59              // First step: Initialize the CloudSim package. It should be called
60              // before creating any entities.
61              int num_user = 1; // number of cloud users
62              Calendar calendar = Calendar.getInstance();
63              boolean trace_flag = false; // mean trace events
```

Step 6: Run the particular project and see the output

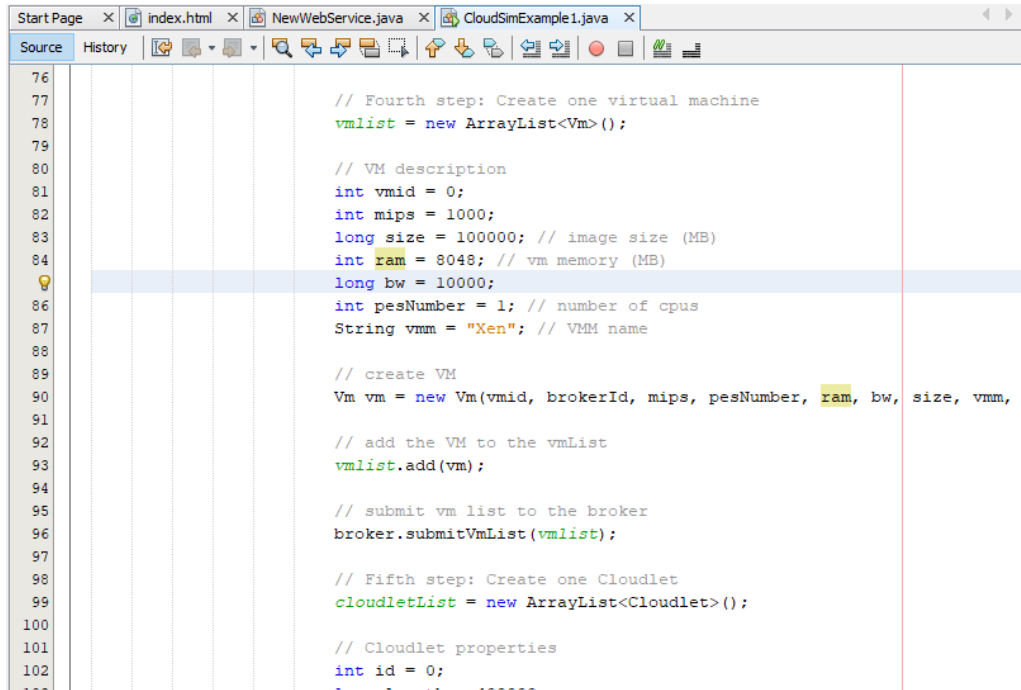


```
run:
Starting CloudSimExample1...
Initialising...
Starting CloudSim version 3.0
Datacenter_0 is starting...
Broker is starting...
Entities started.
0.0: Broker: Cloud Resource List received with 1 resource(s)
0.0: Broker: Trying to Create VM #0 in Datacenter_0
0.1: Broker: VM #0 has been created in Datacenter #2, Host #0
0.1: Broker: Sending cloudlet 0 to VM #0
400.1: Broker: Cloudlet 0 received
400.1: Broker: All Cloudlets executed. Finishing...
400.1: Broker: Destroying VM #0
Broker is shutting down...
Simulation: No more future events
CloudInformationService: Notify all CloudSim entities for shutting down.
Datacenter_0 is shutting down...
Broker is shutting down...
Simulation completed.
Simulation completed.

===== OUTPUT =====
Cloudlet ID   STATUS   Data center ID   VM ID   Time   Start Time   Finish Time
0            SUCCESS    2              0       400     0.1          400.1
****Datacenter: Datacenter_0****
User id      Debt
3            35.6
*****
CloudSimExample1 finished!
BUILD SUCCESSFUL (total time: 1 minute 32 seconds)
```

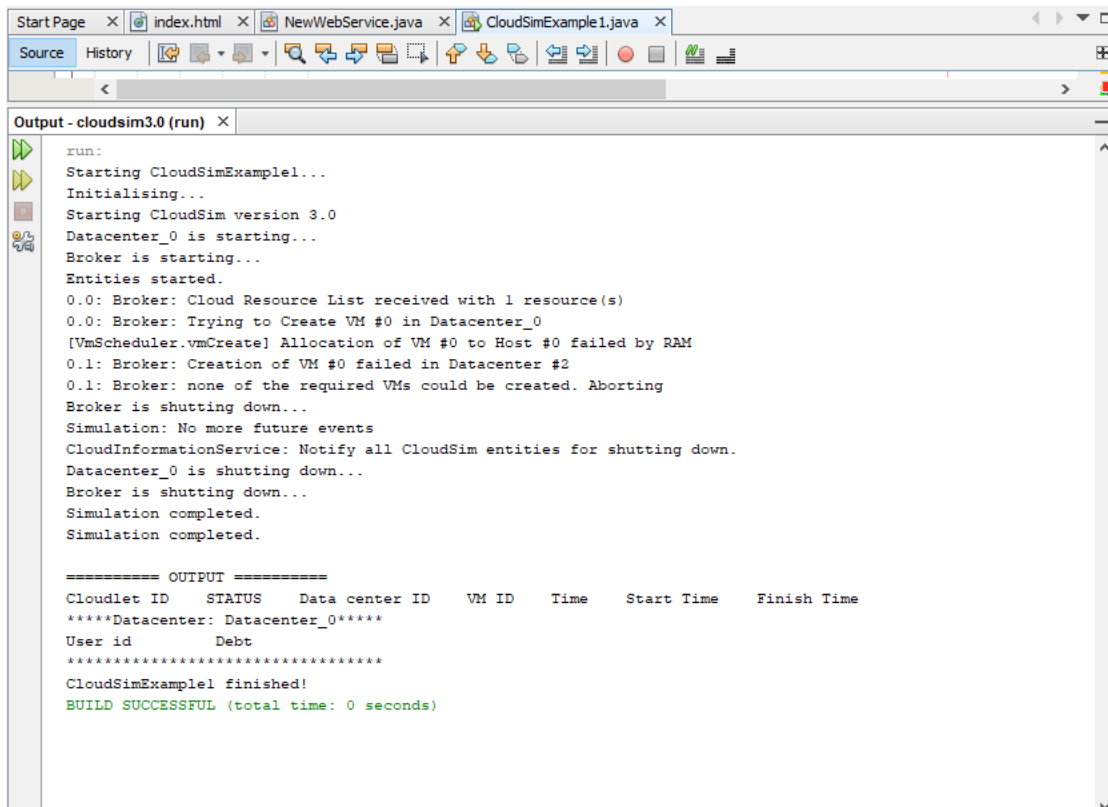
Practical: 4

Step 7: Change the value of Ram, Bandwidth, Mips, Bw



```
76
77 // Fourth step: Create one virtual machine
78 vmList = new ArrayList<Vm>();
79
80 // VM description
81 int vmid = 0;
82 int mips = 1000;
83 long size = 100000; // image size (MB)
84 int ram = 8048; // vm memory (MB)
85 long bw = 10000;
86 int pesNumber = 1; // number of cpus
87 String vmm = "Xen"; // VMM name
88
89 // create VM
90 Vm vm = new Vm(vmid, brokerId, mips, pesNumber, ram, bw, size, vmm);
91
92 // add the VM to the vmList
93 vmList.add(vm);
94
95 // submit vm list to the broker
96 broker.submitVmList(vmList);
97
98 // Fifth step: Create one Cloudlet
99 cloudletList = new ArrayList<Cloudlet>();
100
101 // Cloudlet properties
102 int id = 0;
```

Step 8: Observe the output again we will see the change in execution time



```
run:
Starting CloudSimExample1...
Initialising...
Starting CloudSim version 3.0
Datacenter_0 is starting...
Broker is starting...
Entities started.
0.0: Broker: Cloud Resource List received with 1 resource(s)
0.0: Broker: Trying to Create VM #0 in Datacenter_0
[VmScheduler.vmmCreate] Allocation of VM #0 to Host #0 failed by RAM
0.1: Broker: Creation of VM #0 failed in Datacenter #2
0.1: Broker: none of the required VMs could be created. Aborting
Broker is shutting down...
Simulation: No more future events
CloudInformationService: Notify all CloudSim entities for shutting down.
Datacenter_0 is shutting down...
Broker is shutting down...
Simulation completed.
Simulation completed.

===== OUTPUT =====
Cloudlet ID   STATUS   Data center ID   VM ID   Time   Start Time   Finish Time
****Datacenter: Datacenter_0****
User id      Debt
*****
CloudSimExample1 finished!
BUILD SUCCESSFUL (total time: 0 seconds)
```