

EX.NO:01	INSTALL VIRTUALBOX OR VMWARE ON WINDOWS 8 AND ABOVE, AND SET UP VIRTUAL MACHINES WITH DIFFERENT FLAVORS OF LINUX OR WINDOWS OS, YOU CAN FOLLOW THESE GENERAL STEPS

AIM:

To install VirtualBox or VMware on Windows 8 and above, and set up virtual machines with different flavors of Linux or Windows OS, you can follow these general steps.

PROCEDURE:

1. Choose a virtualization software:
 - VirtualBox: Visit the VirtualBox website (<https://www.virtualbox.org>) and download the latest version compatible with your Windows version.
 - VMware Workstation Player: Go to the VMware website (<https://www.vmware.com/products/workstation-player.html>) and download the latest version of VMware Workstation Player.
2. Install the virtualization software:
 - For VirtualBox: Run the downloaded installer and follow the on-screen instructions to complete the installation.
 - For VMware Workstation Player: Run the downloaded installer and follow the on-screen instructions to complete the installation.
3. Obtain the ISO files for the Linux or Windows OS versions you want to install:
 - For Linux: You can download ISO files from the official websites of the respective Linux distributions. For example, Ubuntu (<https://ubuntu.com/download>), Fedora (<https://getfedora.org>), or CentOS (<https://www.centos.org/download>).
 - For Windows: You will need a valid Windows ISO file, which you can obtain through official channels or use an existing ISO you have.
4. Create a new virtual machine:
 - Launch VirtualBox or VMware Workstation Player, depending on the software you installed.
 - Click on the "New" button to create a new virtual machine.
 - Follow the wizard to configure the virtual machine, including selecting the OS type, assigning memory and storage, and choosing the ISO file for installation.
5. Install the OS on the virtual machine:
 - Start the virtual machine you created.

- It will boot from the selected ISO file, and you can then proceed with the installation process as if you were installing on a physical machine.
6. Repeat steps 4 and 5 for each additional virtual machine with different OS flavors.
- Once you have installed the virtual machines, you can manage and run them within VirtualBox or VMware Workstation Player. You can also configure networking, shared folders, and other settings specific to each virtual machine.
 - Remember to allocate sufficient resources (CPU, memory, storage) to each virtual machine based on the requirements of the OS and the applications you plan to run.

RESULT:

The above experiment has been completed and successfully verified.

EX.NO: 02	INSTALL A C COMPILER IN THE VIRTUAL MACHINE CREATED USING A VIRTUAL BOX AND EXECUTE SIMPLE PROGRAMS

AIM:

To install a c compiler in the virtual machine created using a virtual box and execute simple programs.

PROCEDURE:

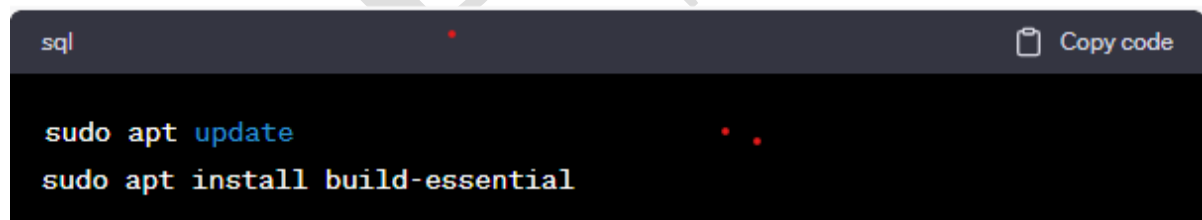
Start the virtual machine:

- Launch VirtualBox and select the virtual machine you created.
- Click on the "Start" button to power on the virtual machine.

Install the C compiler:

- Open a web browser within the virtual machine and download the C compiler package.
- For Linux:
 - If you're using Ubuntu, you can open the terminal and run the following command to install the GNU C Compiler (GCC):

sql



```
sql                                                                    Copy code

sudo apt update
sudo apt install build-essential
```

- For other Linux distributions, you can use the package manager specific to that distribution to install GCC.
- For Windows:
 - i. Download the MinGW (Minimalist GNU for Windows) installer from the official website (<https://osdn.net/projects/mingw/releases/>).
 - ii. Run the installer and follow the on-screen instructions to install MinGW.
 - iii. During the installation process, make sure to select the "C Compiler" component.

- Verify the C compiler installation:
 - i. Open a terminal or command prompt within the virtual machine.
 - ii. Type `gcc --version` and press Enter.
 - iii. If the C compiler is installed correctly, it will display the version information.
- Write and compile a simple C program:
 - i. Open a text editor or integrated development environment (IDE) within the virtual machine.
- Write a simple C program, such as the following example:

```
c Copy code  
  
#include <stdio.h>  
  
int main() {  
    printf("Hello, World!\n");  
    return 0;  
}
```

Save the file with a .c extension, for example, `hello.c`.

- Compile and run the C program:
 - i. Open a terminal or command prompt within the virtual machine.
 - ii. Navigate to the directory where you saved the C program file (e.g., `cd /path/to/file`).
- Use the C compiler to compile the program by running the following command:

For Linux:

```
Copy code  
  
gcc hello.c -o hello
```

For Windows with MinGW:

```
Copy code  
  
gcc hello.c -o hello.exe
```

- The output, "Hello, World!" in this example, should be displayed in the terminal or command prompt.

RPSTF

RESULT:

The above experiment has been completed and successfully verified.

EX.NO:03

Install google App Engine. Create a hello world App and other simple web applications using python/java

AIM:

To Install google App Engine. Create a hello world App and other simple web applications using python/java.

PROCEDURE:

1. Install the necessary tools:
 - Install Python: If you don't have Python installed, visit the Python website (<https://www.python.org>) and download the latest version compatible with your operating system. Follow the installation instructions provided.
 - Install Java Development Kit (JDK): If you don't have Java installed, visit the Oracle Java SE Downloads page (<https://www.oracle.com/java/technologies/javase-jdk11-downloads.html>) and download the JDK for your operating system. Follow the installation instructions provided.
2. Set up the Google Cloud SDK:
 - Visit the Google Cloud SDK documentation (<https://cloud.google.com/sdk/docs/install>) and follow the instructions to download and install the Google Cloud SDK for your operating system.
3. Install the App Engine component:
 - Open a terminal or command prompt.
 - Run the following command to install the App Engine component:

```
gcloud components install app-engine-python
```

For Java applications, use `app-engine-java` instead of `app-engine-python`.

4. Create a new App Engine project:

- Run the following command to create a new project:

```
csharp  
gcloud projects create [PROJECT_ID] --set-as-default
```

Replace `[PROJECT_ID]` with your desired project ID.

5. Initialize the App Engine app:

- Run the following command to initialize the app:

```
lua
```

[Copy code](#)

```
gcloud app create
```

Create a "Hello, World!" app:

- Create a new directory for your app and navigate into it.
- Create a new file named `app.yaml` and add the following content:

```
yaml
```

[Copy code](#)

```
runtime: python39 # For Python app
# runtime: java11 # For Java app

handlers:
- url: /*
  script: auto
```

- Adjust the runtime to `java11` if you're developing a Java app.
- For Python:
 - Create a new file named `main.py` and add the following content:

```
python
```

[Copy code](#)

```
from flask import Flask

app = Flask(__name__)

@app.route('/')
def hello():
    return 'Hello, World!'

if __name__ == '__main__':
    app.run()
```

- For Java:

- Create a new file named `HelloWorldServlet.java` and add the following content:

```
java Copy code  
  
import javax.servlet.annotation.WebServlet;  
import javax.servlet.http.HttpServlet;  
import javax.servlet.http.HttpServletRequest;  
import javax.servlet.http.HttpServletResponse;  
import java.io.IOException;  
  
@WebServlet(name = "HelloWorldServlet", value = "/")  
public class HelloWorldServlet extends HttpServlet {  
    @Override  
    protected void doGet(HttpServletRequest request, HttpServletResponse response)  
        response.setContentType("text/html");  
        response.getWriter().println("Hello, World!");  
    }  
}
```

Deploy the app:

- Run the following command to deploy the app:

```
Copy code  
  
gcloud app deploy
```

- Follow the prompts to choose the region and confirm the deployment.
2. Access the deployed app:
 - After successful deployment, you will receive a URL where your app is accessible.
 3. Create additional web applications:
 - To create additional web applications, repeat the steps from creating the app (step 6) onward, adjusting the code and configuration as needed.

RESULT:

The above experiment has been completed and successfully verified.

EX.NO: 04	USE THE GAE LAUNCHER TO LAUNCH THE WEB APPLICATIONS

AIM:

Use the gae launcher to launch the web applications.

PROCEDURE:**1. Install the necessary tools:**

- Install Python: If you don't have Python installed, visit the Python website (<https://www.python.org>) and download the latest version compatible with your operating system. Follow the installation instructions provided.
- Install Java Development Kit (JDK): If you don't have Java installed, visit the Oracle Java SE Downloads page (<https://www.oracle.com/java/technologies/javase-jdk11-downloads.html>) and download the JDK for your operating system. Follow the installation instructions provided.
- Install the Google Cloud SDK: Visit the Google Cloud SDK documentation (<https://cloud.google.com/sdk/docs/install>) and follow the instructions to download and install the Google Cloud SDK for your operating system.

2. Set up the Google Cloud SDK:

- Open a terminal or command prompt.
- Run the following command to initialize the Google Cloud SDK:



```
csharp
gcloud init
```


Follow the prompts to log in with your Google account, select your project, and configure the SDK.

Navigate to the project directory:

- Open a terminal or command prompt.
- Use the **cd** command to navigate to the directory where your App Engine project files are located.


Deploy the application:

- For Python applications:

 Copy code

```
gcloud app deploy app.yaml
```

- Follow the prompts to choose the region and confirm the deployment.
- For Java applications:
 - Run the following command to deploy the app:

 Copy code

```
gcloud app deploy
```

- Follow the prompts to choose the region and confirm the deployment.

Access the deployed app:

- After successful deployment, you will receive a URL where your app is accessible.

You have now deployed and launched your web application on Google App Engine using the command-line interface. You can continue to manage and deploy your applications using the Google Cloud SDK commands.

Simulate a cloud scenario using cloudsim and run a scheduling algorithm that is not present in cloudsim

RESULT

The above experiment has been completed and successfully verified.

EX.NO:05	SIMULATE A CLOUD SCENARIO USING CLOUDSIM AND RUN A SCHEDULING ALGORITHM THAT IS NOT PRESENT IN CLOUDSIM, YOU CAN FOLLOW THESE STEPS.

AIM:

To simulate a cloud scenario using CloudSim and run a scheduling algorithm that is not present in CloudSim, you can follow these steps.

PROCEDURE:

1. Set up the CloudSim environment:
 - Download the CloudSim library from the official CloudSim website (<https://www.cloudbus.org/cloudsim/>) and extract it to a directory.
 - Create a new Java project in your preferred Integrated Development Environment (IDE).
 - Add the CloudSim JAR file to your project's classpath.
2. Design and implement the scheduling algorithm:
 - Decide on the scheduling algorithm you want to implement. Consider researching and selecting an algorithm suitable for your cloud simulation scenario.
 - Create a new Java class representing your scheduling algorithm.
 - Implement the logic of your scheduling algorithm within the class, considering factors such as task allocation, load balancing, and resource utilization.
 - Make sure to adhere to the CloudSim architecture and interfaces.
3. Create a cloud simulation scenario:
 - Define the characteristics of your cloud simulation scenario, including the number of data centers, virtual machines, tasks, and their properties.
 - Create a Java class representing your simulation scenario.
 - Implement the logic of your simulation scenario, such as creating data centers, allocating resources, generating tasks, and specifying their requirements.
4. Integrate your scheduling algorithm with the simulation scenario:
 - Instantiate your scheduling algorithm class within the simulation scenario class.
 - Utilize your scheduling algorithm to schedule tasks and allocate resources in the simulation.
 - Customize the behavior of your simulation scenario by invoking your scheduling algorithm's methods appropriately.
5. Run the cloud simulation:
 - Configure the simulation parameters, such as the simulation length, logging options, and output preferences.
 - Execute the simulation scenario, triggering the simulation of the cloud environment and the execution of your scheduling algorithm.
 - Monitor the simulation progress and analyze the results generated by the simulation.

RESULT

RESULT:

The above experiment has been completed and successfully verified.

EX.NO:06	FIND A PROCEDURE TO TRANSFER THE FILES FROM ONE VIRTUAL MACHINE TO ANOTHER VIRTUAL MACHINE


AIM:

To find a procedure to transfer the files from one virtual machine to another virtual machine.

PROCEDURE:

1. Determine the file transfer method:
 - There are several methods you can use to transfer files between virtual machines, including:
 - File transfer protocols such as FTP (File Transfer Protocol) or SFTP (SSH File Transfer Protocol).
 - Shared folders or network file sharing.
 - Secure copy (SCP) command-line utility.
 - Cloud storage services.
2. Set up the file transfer method:
 - Depending on the chosen method, you may need to configure the virtual machines accordingly:
 - For FTP or SFTP: Install and set up an FTP or SFTP server on the destination virtual machine and an FTP or SFTP client on the source virtual machine.
 - For shared folders or network file sharing: Enable shared folders or network file sharing on both virtual machines.
 - For SCP: Ensure that the SSH service is enabled on both virtual machines.
 - For cloud storage services: Sign up for a cloud storage service, install the required client or software on both virtual machines, and configure access to the cloud storage.
3. Transfer files using the chosen method:
 - Follow the specific instructions for the selected file transfer method:
 - For FTP or SFTP: Connect to the destination virtual machine from the source virtual machine using an FTP or SFTP client, and transfer the files using standard file transfer commands.
 - For shared folders or network file sharing: Copy the files to the shared folder or network location accessible by both virtual machines.
 - For SCP: Use the SCP command-line utility to securely copy files between the virtual machines. For example:

ruby

 Copy code

```
scp /path/to/source/file user@destination:/path/to/destination/file
```

- For cloud storage services: Upload the files to the cloud storage service from the source virtual machine and then download them from the destination virtual machine.

Verify the file transfer:

- Once the file transfer is completed, verify that the files have been successfully transferred to the destination virtual machine.
- Confirm the integrity and completeness of the transferred files by checking their size and content.

Remember to consider security aspects, such as encrypting sensitive data during file transfers and configuring appropriate access controls and firewall rules to protect the virtual machines.

RESULT:

The above experiment has been completed and successfully verified.

EX.NO:07	INSTALL HADOOP SINGLE NODE CLUSTER AND RUN SIMPLE APPLICATIONS LIKE WORDCOUNT

AIM:

To install hadoop single node cluster and run simple applications like word count.

PROCEDURE:

1. Set up a single-node Hadoop cluster:

- Download the Hadoop distribution from the Apache Hadoop website (<https://hadoop.apache.org/releases.html>) and extract it to a directory.
- Configure the necessary environment variables in your operating system:
 - Open the `bashrc` or `bash_profile` file in a text editor:

```
bash
```

[Copy code](#)

```
nano ~/.bashrc
```

Add the following lines at the end of the file:

```
bash
```

[Copy code](#)

```
export HADOOP_HOME=/path/to/hadoop/directory
export PATH=$PATH:$HADOOP_HOME/bin:$HADOOP_HOME/sbin
```

Save and close the file.

Run the following command to load the updated environment variables:

```
bash
```

[Copy code](#)

```
source ~/.bashrc
```

Configure Hadoop:

- Open the `hadoop-env.sh` file located in the `etc/hadoop` directory of your Hadoop installation.
- Set the `JAVA_HOME` variable to the path of your Java Development Kit (JDK) installation. For example:

javascript

Copy code

```
export JAVA_HOME=/path/to/java/directory
```

Save and close the file.

Configure Hadoop single-node cluster settings:

Open the `core-site.xml` file located in the `etc/hadoop` directory.

Add the following configuration:

xml

Copy code

```
<configuration>
  <property>
    <name>fs.defaultFS</name>
    <value>hdfs://localhost:9000</value>
  </property>
</configuration>
```

- Save and close the file.
- Open the `hdfs-site.xml` file located in the same directory.
- Add the following configuration:

xml

Copy code

```
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>
</configuration>
```

- Save and close the file.

Format the Hadoop file system:

- Open a terminal or command prompt and run the following command:

lua

Copy code

```
hdfs namenode -format
```

Start the Hadoop services:

- Run the following command to start the Hadoop services:


```
sql Copy code  
  
start-dfs.sh  
start-yarn.sh
```

1. Verify the Hadoop cluster setup:
 - Open a web browser and visit the Hadoop cluster web interface by accessing <http://localhost:9870>.
 - Verify that the Hadoop cluster services are running correctly.
2. Run the WordCount application:
 - Copy the input file for the WordCount application to the Hadoop distributed file system (HDFS). For example:

```
css Copy code  
  
hdfs dfs -mkdir -p /input  
hdfs dfs -put /path/to/input/file /input
```

Run the WordCount application using the following command:

```
bash Copy code  
  
hadoop jar /path/to/hadoop/directory/share/hadoop/mapreduce/hadoop-mapred
```

Once the job is completed, you can view the output using the following command:

```
bash Copy code  
  
hdfs dfs -cat /output/part-r-00000
```

RESULT:

The above experiment has been completed and successfully verified.

EX.NO:08

CREATING AND EXECUTING YOUR FIRST CONTAINER USING DOCKER

AIM:

Creating and executing your first container using docker

PROCEDURE:

1. Install Docker:

- Visit the Docker website (<https://docs.docker.com/get-docker>) and follow the instructions to download and install Docker for your operating system.

2. Verify Docker installation:

- Open a terminal or command prompt.
- Run the following command to verify that Docker is installed correctly:

```
docker version
```

Copy code

Pull a Docker image:

- Docker images are the building blocks for containers. To start, you need to pull an existing Docker image from the Docker Hub or a private registry.
- Run the following command to pull a basic image like "hello-world":

```
docker pull hello-world
```

Copy code

Create a Docker container:

- Containers are instances of Docker images. You can create a container using the **docker run** command.
- Run the following command to create and run a container based on the "hello-world" image:

```
arduino
```

```
docker run hello-world
```

Copy code

1. View container output:

- Docker will download the "hello-world" image if it's not already available locally. The container will run and display its output, which should include a message confirming the successful execution of the container.

2. List running containers:

- To see a list of running containers, use the following command:

```
docker ps
```

7. View container logs:

- If you want to view the logs of a specific container, note down its container ID or name from the previous step, and run the following command:

```
css
```

```
docker logs [CONTAINER_ID/CONTAINER_NAME]
```

8. Stop and remove the container:

- To stop a running container, use the following command:

```
arduino
```

```
docker stop [CONTAINER_ID/CONTAINER_NAME]
```

- To remove a container, use the following command:

```
bash
```

```
docker rm [CONTAINER_ID/CONTAINER_NAME]
```

Congratulations! You have created and executed your first Docker container. You can now explore and experiment with different Docker images and containers to run various applications and services.

RESULT:

The above experiment has been completed and successfully verified.

EX.NO:09	RUN A CONTAINER FROM DOCKER HUB

AIM:

Run a container from docker hub

PROCEDURE:


1. Search for the desired image:

- Visit the Docker Hub website (<https://hub.docker.com>) or use the Docker CLI to search for the image you want to run.
- For example, if you want to run an Nginx web server, you can search for the "nginx" image.

2. Pull the Docker image:

- Once you have identified the image you want to run, use the **docker pull** command to download it.
- For example, to pull the "nginx" image, run the following command:

```
docker pull nginx
```


 Copy code

Run the container:

- After pulling the image, you can run a container based on it using the **docker run** command.
- Specify any additional configuration or runtime options as needed.
- For example, to run an Nginx container and expose port 80, use the following command:

```
arduino
```

```
docker run -d -p 80:80 nginx
```


 Copy code

1.

-
- The **-d** flag runs the container in detached mode (in the background).
- The **-p** flag maps port 80 of the host to port 80 of the container.

2. Verify the running container:

- Use the **docker ps** command to list the running containers and ensure that the container you just started is running:

 Copy code

```
docker ps
```


1. Access the containerized application:

- Open a web browser and visit `http://localhost` or `http://<host-ip>` to access the containerized application running inside the Nginx container.
- If you exposed a different port during the `docker run` command, use that port in the URL.

2. Stop and remove the container:

- To stop the container, use the following command, replacing `[CONTAINER_ID/CONTAINER_NAME]` with the appropriate value:


arduino

 Copy code

```
docker stop [CONTAINER_ID/CONTAINER_NAME]
```

To remove the container, use the following command:

bash

 Copy code

```
docker rm [CONTAINER_ID/CONTAINER_NAME]
```

By following these steps, you can run a container from Docker Hub and access the containerized application. Docker Hub provides a vast repository of pre-built images for various applications, frameworks, and services, allowing you to easily run and deploy containers for your specific needs.

RESULT:

The above experiment has been completed and successfully verified.