BACHELORS OF SCIENCE IN COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

IOST, TRIBHUVAN UNIVERSITY

Thimi, Bhaktapur

Affiliated to Tribhuvan University

Lab Report

CSC 467: Introduction to Cloud Computing

SUBMITTED BY

NAME: PARASH THAPA
SYMBOL NO/ROLL NO: 24380
PROGRAM: BSc. CSIT

SUBMITTTED TO

SUSHIL NEPAL

Lab 1: Install Virtual Machine and configuring Linux on it (Type 2 virtualization).

Theory:

Virtualization can increase IT agility, flexibility and scalability while creating significant cost savings. Greater workload mobility, increased performance and availability of resources, and automated operations – they're all benefits of virtualization that make IT simpler to manage and less costly to own and operate.

Procedure:

Steps to install and configure VMWare:

1. Download VMWare workstation trial version setup file from here and make sure the latest version is being downloaded and installed.

2. Install VMWare on your machine. Setup is simple and requires to click Next button couple of times.

3. After installation open VMWare workstation by using either start menu or shortcut created on the desktop.

4. Click on —Create a New Virtual Machine‖.

5. With default —Typical‖ selected click on Next button.

6. Specify the path of the operating system set up file.

7. In the Next step you need to specify a Key or a serial number of operating system. If you are using trial version then that part can be

skipped.

8. Enter the name for the virtual machine and specify a path to the directory where you want to create your virtual machine. It is recommended that the drive you‘re selecting to install virtual machine should have sufficient space.

9. Specify an amount of disk space you want to allocate for a virtual machine. Allocate disk space according to the size of software you are going to install on the virtual machine.

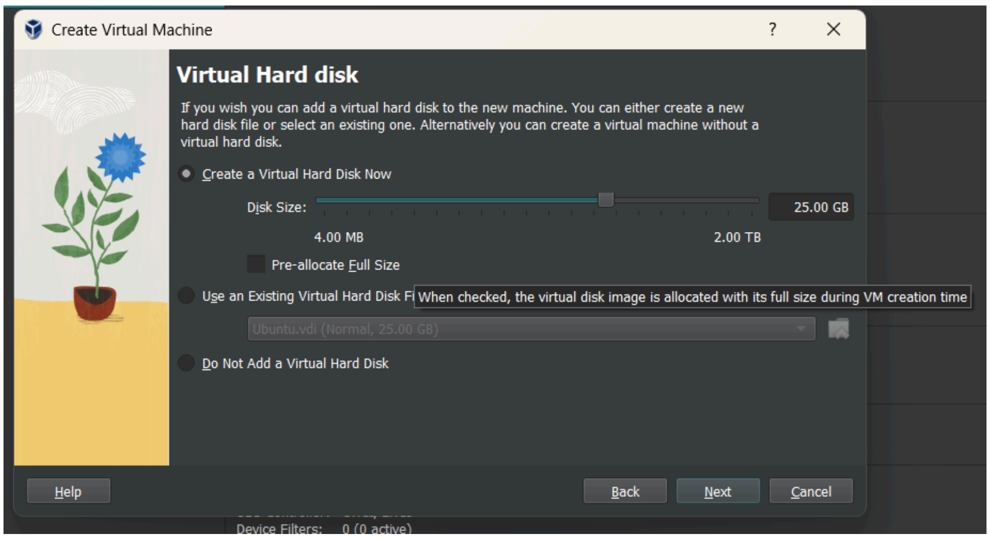
10. On the next screen it will show configuration you selected for a virtual machine.

11. It will allocate Hardware according to the default settings but you can change it by using Customize Hardware button in the above screen. You can specify what amount of RAM, a processor has to be allocated for a virtual machine. Do not allocate complete RAM or complete Processor for a virtual machine. Also, do not allocate very less RAM or processor. Leave default settings or allocate in such way that your application should be able to run on the virtual machine. Else it will result in a slow virtual machine.

12. Click on the Finish button to create the virtual machine at the specified location and with specified resources.

Output:

Allocating Disk Space for Virtual Machine:


Create Virtual Machine

Virtual Hard disk

If you wish you can add a virtual hard disk to the new machine. You can either create a new hard disk file or select an existing one. Alternatively you can create a virtual machine without a virtual hard disk.

Create a Virtual Hard Disk Now

Disk Size: 25.00 GB

4.00 MB 2.00 TB

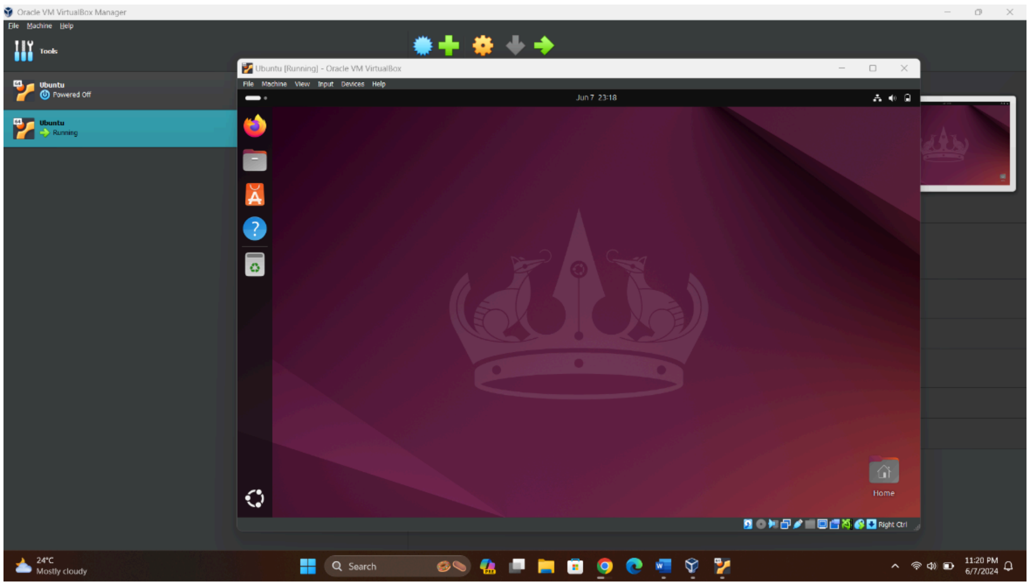
Pre-allocate Full Size

Use an Existing Virtual Hard Disk Fi When checked, the virtual disk image is allocated with its full size during VM creation time

Ubuntu.vdi (Normal, 25.00 GB)

Do Not Add a Virtual Hard Disk

Virtual Machine Created:



Conclusion:

Cloud computing provides measured service to the users and that can be achieved by using virtualization. VMWare- a popular application that can be used to configure virtual machines in the same computer and make them work as separate entities, which is foundational to the very concept of cloud computing.

Lab 2: Run a Simple C Program in Virtual Machine using Type2 Virtualization.

Theory:

As the VMWare can be installed and utilized as an entirely separate computing unit, different programs can be run on the virtual machine as if it was a real computer. In this lab, a simple program in C is executed in the Guest OS of the virtual machine thus solidifying the usability of virtual machine.

Procedure:

1. Create a .C extension file and write program

2. Install gcc compiler

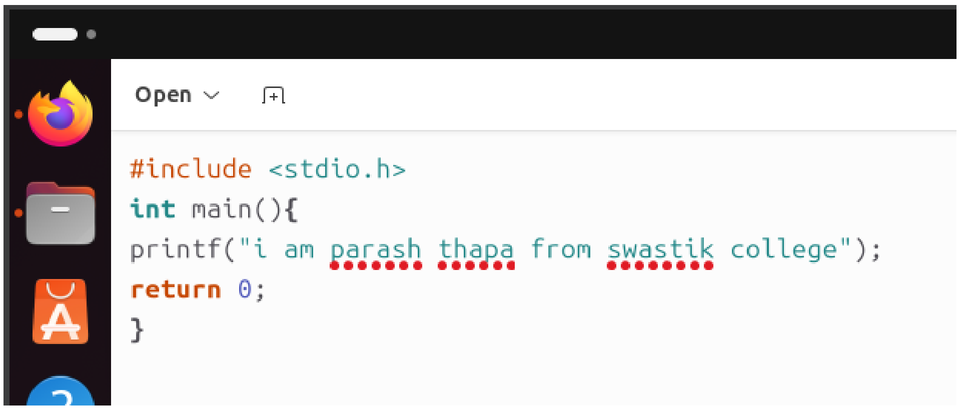
3. sudo apt-get install libncurses5-dev libncursesw5-dev for curses.h (use curses.h instead of conio.h)

4. Compile the c program using gcc pathtofile -o filename

5. Execute ./filename

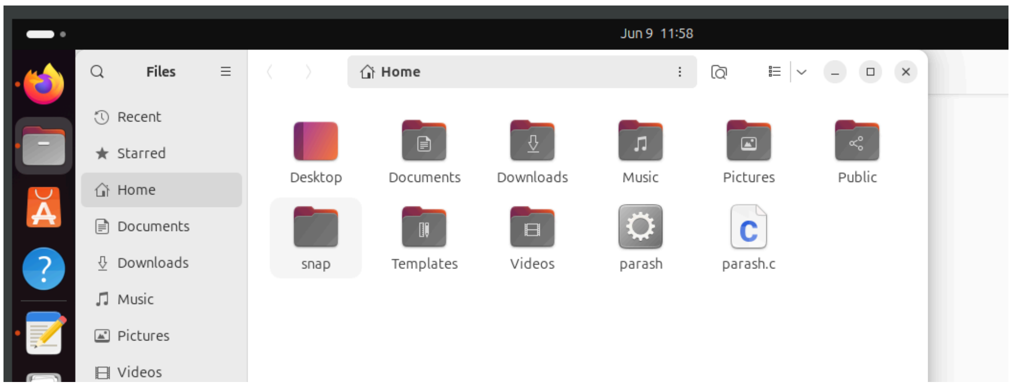
Output:

C program source code file:

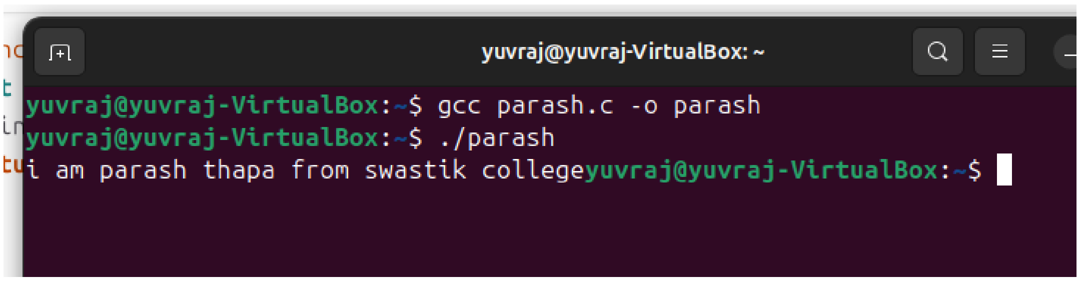


```
#include <stdio.h>
int main(){
printf("i am parash thapa from swastik college");
return 0;
}
```

Source code file hello.c and compiled executable file test:



Running executable file and source code output:



```
yuvraj@yuvraj-VirtualBox:~$ gcc parash.c -o parash
yuvraj@yuvraj-VirtualBox:~$ ./parash
i am parash thapa from swastik collegeyuvraj@yuvraj-VirtualBox:~$ 
```

Conclusion:

Therefore, using VMWare, separate programs can be executed in them making it possible for multiple devices to be allocated for different users in cloud computing where these users can have their own workspace.

Lab 3: Run a Simple Java Program in Virtual Machine using Type2 Virtualization.

Theory:

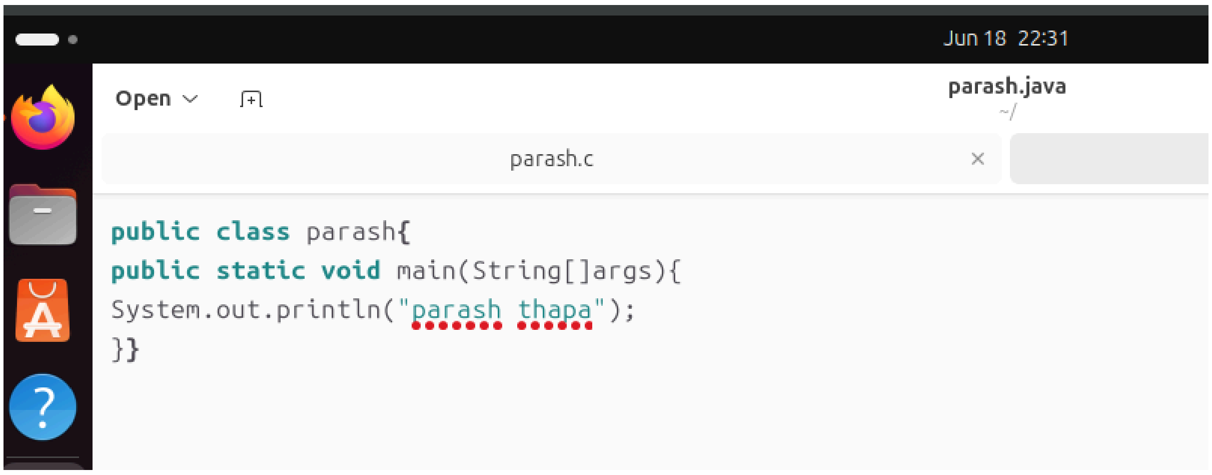
As the VMWare can be installed and utilized as an entirely separate computing unit, different programs can be run on the virtual machine as if it was a real computer. In this lab, a simple program in java is executed in the Guest OS of the virtual machine thus solidifying the usability of virtual machine.

Procedure:

1. Create a .java extension file and write program
2. Install java using sudo apt install openjdk-8-jdk
3. export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-amd64/ and source the .bashrc
4. Compile the java program using javac filename.java
5. Execute using java filename

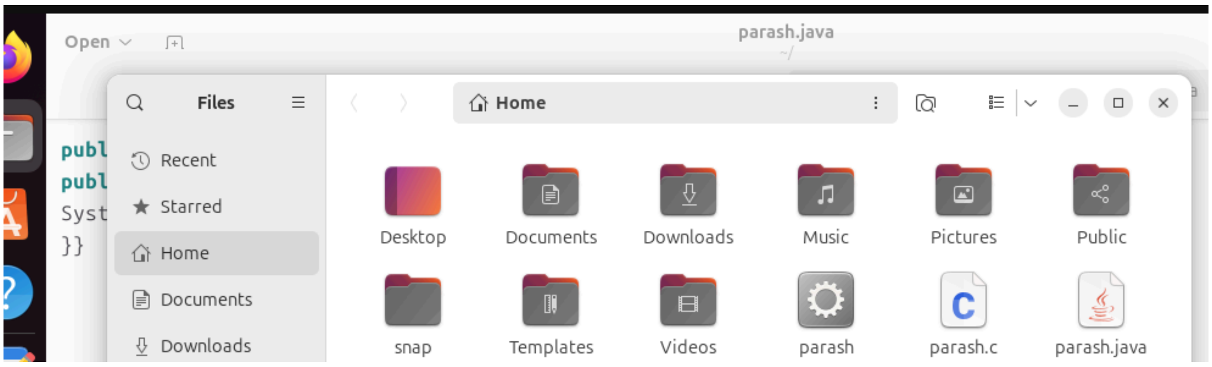
Output:

Java program source code file:



```java
public class parash{
public static void main(String[]args){
System.out.println("parash thapa");
}}
```

Source code file Hello.java and compiled executable file Hello.class:



Running the compiled Java program and output:

Conclusion:

Therefore, using VMWare, separate programs can be executed in them making it possible for multiple devices to be allocated for different users in cloud computing where these users can have their own workspace.

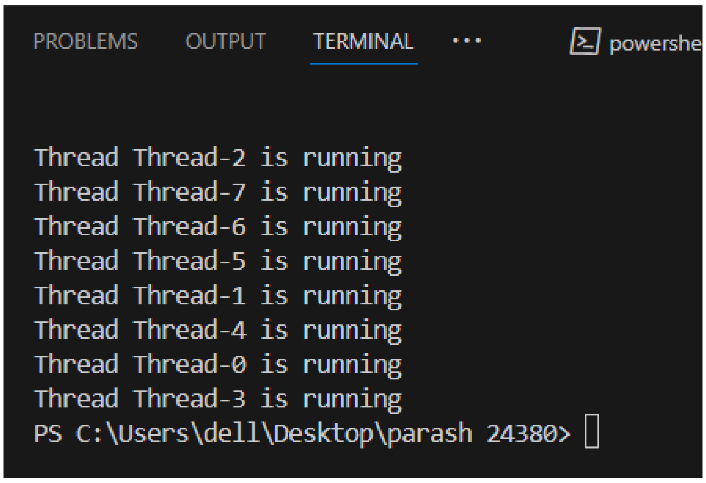
Lab 4: Implement Multithread programming using java.

Program 1:

Source Code:

```
class MultithreadingDemo extends Thread {
   public void run()
   {
      try {
         // Displaying the thread that is running
         System.out.println(
            "Thread " + Thread.currentThread().getId()
            + " is running");
        }
        catch (Exception e) {
            // Throwing an exception
            System.out.println("Exception is caught");
        }
    }
}

// Main Class
public class Multithread {
    public static void main(String[] args)
    {
        int n = 8; // Number of threads
        for (int i = 0; i < n; i++) {
            MultithreadingDemo object
                = new MultithreadingDemo();
            object.start();
        }
    }
}
```

Output:



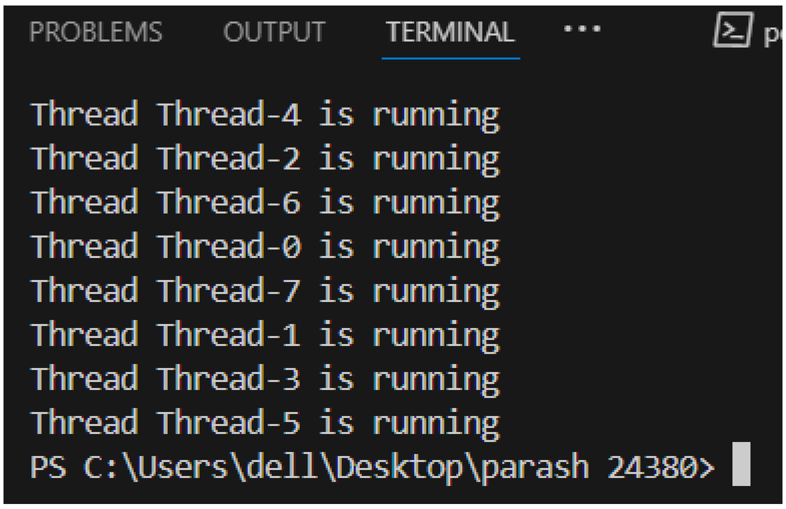
```
Thread Thread-2 is running
Thread Thread-7 is running
Thread Thread-6 is running
Thread Thread-5 is running
Thread Thread-1 is running
Thread Thread-4 is running
Thread Thread-0 is running
Thread Thread-3 is running
PS C:\Users\dell\Desktop\parash 24380>
```

Program 2:

Source Code:

```
class MultithreadingDemo implements Runnable {

    public void run()

    {

        try {

            // Displaying the thread that is running

            System.out.println(

                "Thread " + Thread.currentThread().getId()

                + " is running");

        }

        catch (Exception e) {

            // Throwing an exception

            System.out.println("Exception is caught");

        }

    }

}



// Main Class

class Multithread {

    public static void main(String[] args)
    {
        int n = 8; // Number of threads
        for (int i = 0; i < n; i++) {
            Thread object
                = new Thread(new MultithreadingDemo());
            object.start();
        }
    }
}
```

Output:



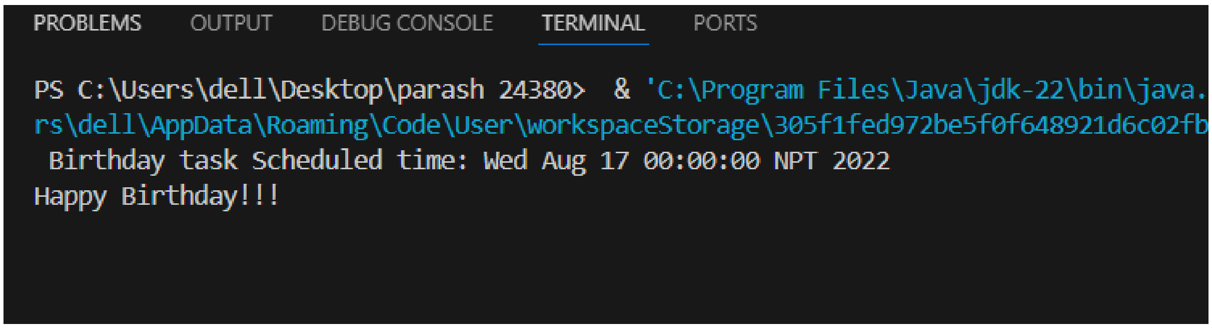
```
Thread Thread-4 is running
Thread Thread-2 is running
Thread Thread-6 is running
Thread Thread-0 is running
Thread Thread-7 is running
Thread Thread-1 is running
Thread Thread-3 is running
Thread Thread-5 is running
PS C:\Users\dell\Desktop\parash 24380>
```

Lab 5: Task Programming

Source Code:

```
public class TestTask {

   public static void main(String[] args) {

      Timer T=new Timer();

      TimerTask birthday=new TimerTask() {

         @Override

         public void run() {

            System.out.println("Happy Birthday!!!");

         }

      } ;

      Calendar date=Calendar.getInstance();

      date.set(2022,Calendar.AUGUST,17,0,0,0);


      T.schedule(birthday,date.getTime());

   }

}
```

Output:



```
PS C:\Users\dell\Desktop\parash 24380>  & 'C:\Program Files\Java\jdk-22\bin\java.
rs\dell\AppData\Roaming\Code\User\workspaceStorage\305f1fed972be5f0f648921d6c02fb
 Birthday task Scheduled time: Wed Aug 17 00:00:00 NPT 2022
Happy Birthday!!!
```

Lab 5: Implementation of MapReduce programming.

Algorithm:

1. Splitting – The splitting parameter can be anything, e.g. splitting by space, comma, semicolon,

or even by a new line (‘\n’).

2. Mapping – as explained above.

3. Intermediate splitting – the entire process in parallel on different clusters. In order to group them

in “Reduce Phase” the similar KEY data should be on the same cluster.

4. Reduce – it is nothing but mostly group by phase.

5. Combining – The last phase where all the data (individual result set from each cluster) is

combined together to form a result.

6. Open Eclipse> File > New > Java Project >(Name it – MRProgramsDemo) > Finish.

7. Right Click > New > Package (Name it - PackageDemo) > Finish.

8. Right Click on Package > New > Class (Name it - WordCount).

9. Add Following Reference Libraries:

10. Right Click on Project > Build Path> Add External

11. /usr/lib/hadoop-0.20/hadoop-core.jar

12. Usr/lib/hadoop-0.20/lib/Commons-cli-1.2.jar

Source Code:

```
package PackageDemo;

import java.io.IOException;

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Job;

import org.apache.hadoop.mapreduce.Mapper;

import org.apache.hadoop.mapreduce.Reducer;

import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

import org.apache.hadoop.util.GenericOptionsParser;

public class WordCount {

public static void main(String [] args) throws Exception

{

Configuration c=new Configuration();

String[] files=new GenericOptionsParser(c,args).getRemainingArgs();

Path input=new Path(files[0]);
Path output=new Path(files[1]);

Job j=new Job(c,"wordcount");

j.setJarByClass(WordCount.class);

j.setMapperClass(MapForWordCount.class);

j.setReducerClass(ReduceForWordCount.class);

j.setOutputKeyClass(Text.class);

j.setOutputValueClass(IntWritable.class);

FileInputFormat.addInputPath(j, input);

FileOutputFormat.setOutputPath(j, output);
System.exit(j.waitForCompletion(true)?0:1);

}

public static class MapForWordCount extends Mapper<LongWritable, Text, Text,
IntWritable>{

public void map(LongWritable key, Text value, Context con) throws IOException,

InterruptedException

{

String line = value.toString();

String[] words=line.split(",");

for(String word: words )

{

 Text outputKey = new Text(word.toUpperCase().trim());

 IntWritable outputValue = new IntWritable(1);
 con.write(outputKey, outputValue);

}

}

}

public static class ReduceForWordCount extends Reducer<Text, IntWritable, Text,
IntWritable>

{

public void reduce(Text word, Iterable<IntWritable> values, Context con) throws
IOException,

InterruptedException

{

int sum = 0;

 for(IntWritable value : values)

 {

 sum += value.get();

 }

 con.write(word, new IntWritable(sum));

}

}

}
```

The above program consists of three classes:

- 🎬 Driver class (Public, void, static, or main; this is the entry point).
- 🎬 The Map class which extends the public class

Mapper<KEYIN,VALUEIN,KEYOUT,VALUEOUT> and implements the Map function.

🎬 The Reduce class which extends the public class

Reducer<KEYIN,VALUEIN,KEYOUT,VALUEOUT> and implements the Reduce

function.

Make a jar file

Right Click on Project> Export> Select export destination as Jar File > next> Finish