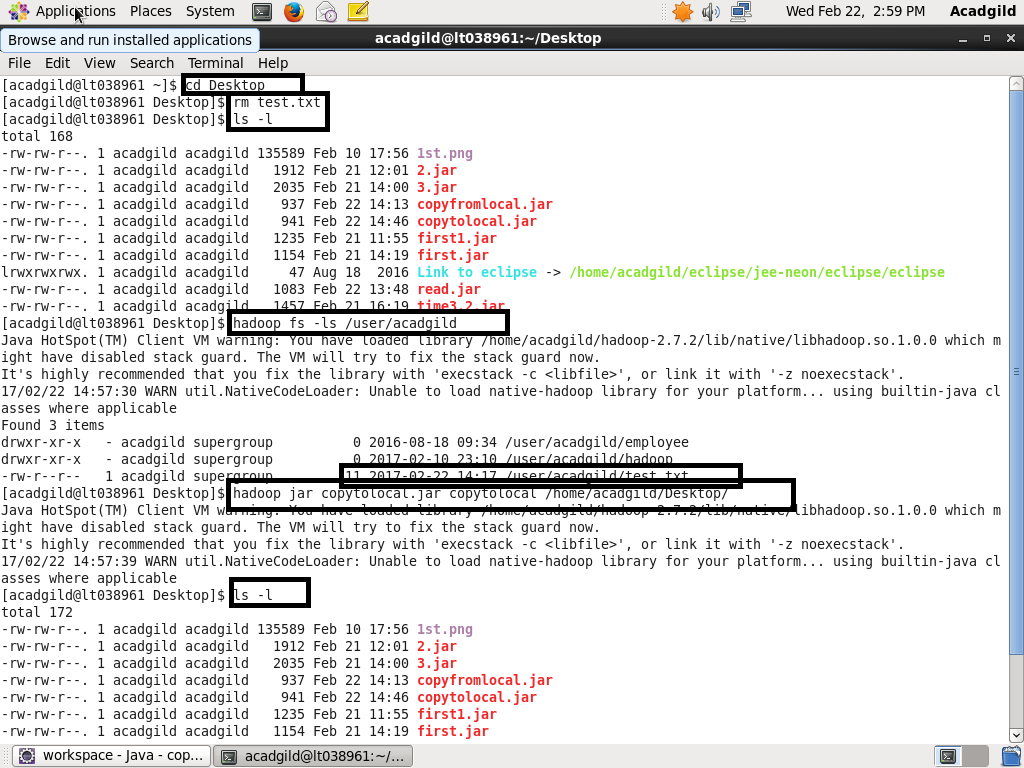
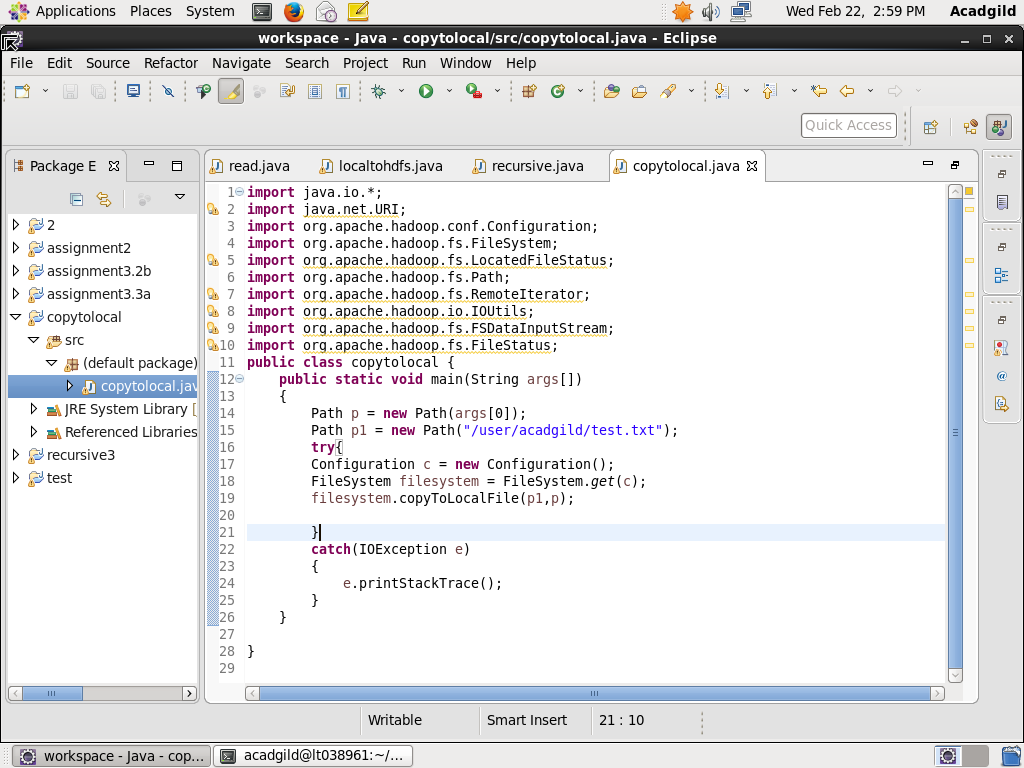
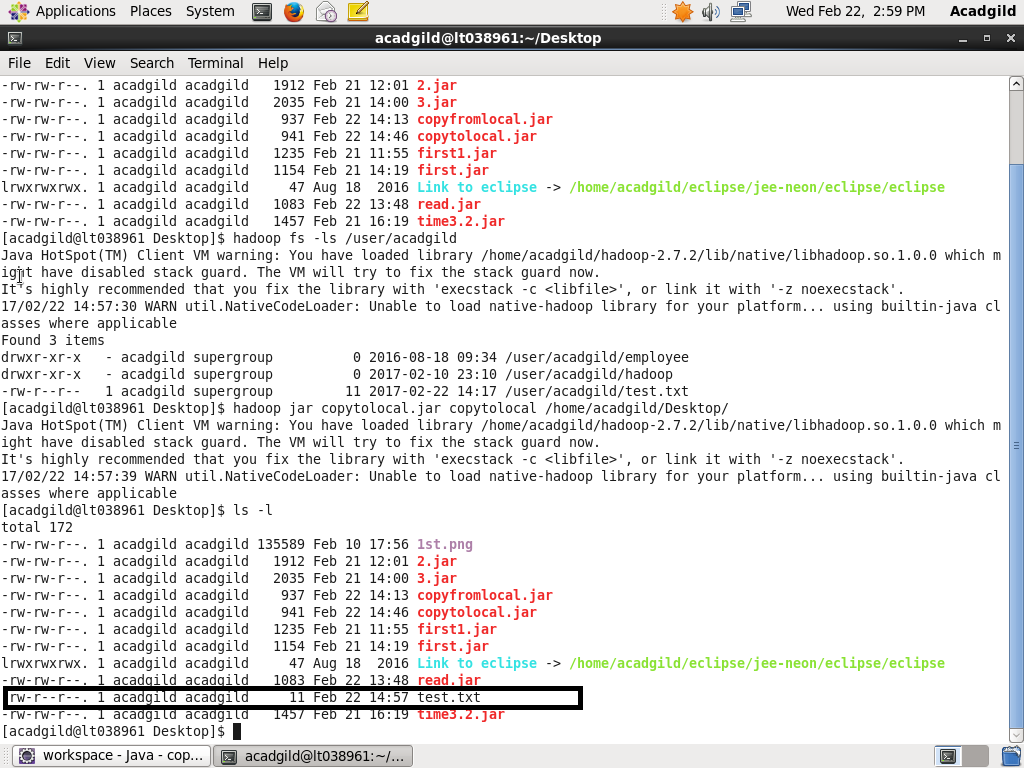
1. Write a Java program to copy a file from HDFS to LFS

First removed the test.txt from local system. Then assured it with ls command . Checked the test.txt in Hadoop file system. Copied test.txt from Hadoop fs to local file system. Again checked in the local file system using ls –l command.

1. Explain the importance and usage of the below terms in details **● DFSInputStream ● DFSOutputStream ● FSDataInputStream ● FSDataOutputStream**

Answer-

**DfsoutputStream** - It is one of the important class of the hadoop architecture and it creates files from stream of bytes.

Following Steps tell us how the file is prepared

1) Client app written data is cached by this stream, this data is broken down into packets. Each packet is of 64KB.

2) These packets are made of data chunks which are of the size 512bytes and each one has a checksum associated with it.

3) Whenever the packet gets full the packet is sent to the data queue where the data streamer takes the packet from the queue and puts into the datanode for this action the datanode sends and acknowledgment.

**DFSInputStream**-This class is used while reading the data from HDFS by a client. When the name node returns the address of the block to the DFS, the DFS returns DFSInputStream (stream of bytes which support seek)

FSDataInputStream in turn wraps a DFSInputStream, which manages the datanode and namenode I/O. When the client calls read() operation the DFSInputStream which stores the address of the data nodes connect to one of the

nearest datanode and then with repeated read() calls the data is read from the data node. When the block reaches the end of the file it closes the connection with the data node and finds next data node for next block of data.

**FSDataInputStream**- FSDataInputStream wraps the DataInputStream and provides us with method like seek(),getPOS()

FileSystem have open() method which return FSDataInputStream as below:

/\*

URI uri = URI.create (“/ ”);

Configuration conf = new Configuration ();

FileSystem file = FileSystem.get (uri, conf);

FSDataInputStream in = file.open(new Path(uri));

\*/

here the buffer size is default of 4KB but it can be modified according to user.

FSDataOutputStream- FileSystem returns FSDataOutputStream on call of create() which is used to create new file or write at EOF. No seek method is provided for write as it has the limitation to write on the EOF.

Create method on FileSystem create file e.g.

/\*

public FSDataOutputStream create(Path f) create empty file.

public FSDataOutputStream append(Path f) will append existing file

\*/