EX.NO:1.1 # CAESAR CIPHER

DATE:

AIM:

To write a java program to perform encryption and decryption using Caesar cipher algorithm.

ALGORITHM:

- Read the plaintext
- Invoke the method for encryption
- The Caesar cipher encryption involves replacing each letter of the alphabet with the letter standing three places further down the alphabet.
    - C: E(P,3) = (P + 3) mod 26
- Invoke decryption method
    - P: D(C, 3) = (C + 3) mod 26

PROGRAM:

```
import java.util.Scanner;
  class cipher
{
    String alphabet = "abcdefghijklmnopqrstuvwxyz";
     String encrypt(String plainText, int shiftKey)
    {
        plainText = plainText.toLowerCase();
        String cipherText = "";
        for (int i = 0; i < plainText.length(); i++)
        {
            int charPosition =
alphabet.indexOf(plainText.charAt(i));
            int keyVal = (shiftKey + charPosition) % 26;
            char replaceVal = alphabet.charAt(keyVal);
            cipherText += replaceVal;
        }
        return cipherText;
    }

    String decrypt(String cipherText, int shiftKey)
    {
        cipherText = cipherText.toLowerCase();
        String plainText = "";
        for (int i = 0; i < cipherText.length(); i++)
        {
            int charPosition =
alphabet.indexOf(cipherText.charAt(i));
            int keyVal = (charPosition - shiftKey) % 26;
            if (keyVal < 0)
            {
                keyVal = alphabet.length() + keyVal;
```

```
            }
            char replaceVal = alphabet.charAt(keyVal);
            plainText += replaceVal;
        }
        return plainText;
    }
}
class Ccipher
{
    public static void main(String[] args)
    {
       cipher c=new  cipher();
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the String for Encryption: ");
        String message = new String();
        message = sc.next();
       String cipherText = new String();
        cipherText=c.encrypt(message, 3);
        System.out.println("Encryption:");
        System.out.println(cipherText);
        System.out.println("Decryption:");
        System.out.println(c.decrypt(cipherText, 3));
        sc.close();
    }
}
```

OUTPUT:

D:\Java\jdk1.6.0\bin>javac Ccipher.java

D:\Java\jdk1.6.0\bin>java Ccipher

Enter the String for Encryption:

computer

Encryption:

frpsxwhu

Decryption:

Computer

RESULT:

Thus a java program to perform encryption and decryption using Caesar cipher algorithm was executed successfully.

EX.NO: 1.2

PLAYFAIR CIPHER

DATE:

AIM:

To write a java program to perform encryption and decryption using Playfair cipher technique.

ALGORITHM:

- The Playfair algorithm is based on the use of a 5 x 5 matrix of letters constructed using a keyword. The matrix is constructed by filling in the letters of the keyword (minus duplicates) from left to right and from top to bottom, and then filling in the remainder of the matrix with the remaining letters in alphabetic order. The letters I and J count as one letter.
- Plaintext is encrypted two letters at a time, according to the following rules:
- Repeating plaintext letters that are in the same pair are separated with a filler letter, such as x.
- Two plaintext letters that fall in the same row of the matrix are each replaced by the letter to the right, with the first element of the row circularly following the last.
- Two plaintext letters that fall in the same column are each replaced by the letter beneath, with the top element of the column circularly following the last.
- Otherwise, each plaintext letter in a pair is replaced by the letter that lies in its own row and the column occupied by the other plaintext letter.

PROGRAM:

```
import java.util.*;
class Playfaircipher1
{
 public static void main(String[] args)
  {
   int j,i,m=0,k=0,val,z=0,flag=0,count=0;
   char T[][]=new char[5][5];
   String enmsg="";
   String plainmsg="";
   String alpha="abcdefghiklmnopqrstuvwxyz";
   Scanner sc=new Scanner(System.in);
   System.out.println("\n\nenter the msg:\t");
   String msg=sc.next();
   System.out.println("\n\nenter the key:\t");
   String key=sc.next();
  char p[]=new char[26];
  for(i=0;i<alpha.length();i++)
  {
   for(j=0;j<key.length();j++)
   {
    if(alpha.charAt(i)==key.charAt(j))
    {
     flag=1;
```

```
      break;
     }
    }
    if(flag==0)
    {
     p[z]=alpha.charAt(i);
     z++;
    }
    flag=0;
   }
  z=0;
   for(i=0;i<5;i++)
   {
    for(j=0;j<5;j++)
    {
     if(count==key.length())
     {
      T[i][j]=p[z];
      z++;
     }
     else
     {
      T[i][j]=key.charAt(m);
      m++;
      count++;
     }
    }
   }
            System.out.println("\n\nThe matrix:\n");
            for(i=0;i<5;i++)
            {
             for(j=0;j<5;j++)
             {
              System.out.print(T[i][j]);
              System.out.print("\t");
             }
             System.out.println("\n");
            }
val=msg.length();
if((val%2)==1)
msg+='x';
int I1=0,I2=0,J1=0,J2=0,c,flag1=0,flag2=0;
m=0;
   //encryption
  do
  {
   for(i=0;i<5;i++)
   {

        for(j=0;j<5;j++)
```

```
      {
           if(T[i][j]==msg.charAt(m))
            {
             I1=i;
             J1=j;
             flag1=1;
            }
           if(T[i][j]==msg.charAt(m+1))
            {
             I2=i;
             J2=j;
             flag2=1;
             }
           if(flag1==1 && flag2==1)
             break;
       }
if(flag1==1 && flag2==1)
      {
           if(I1==I2)
              c=1;
           else if(J1==J2)
              c=2;
           else
              c=3;
             switch(c)
               {
                   case 1:

                          if((J1+1)==5)
                             enmsg+=T[I1][0];
                          else
                             enmsg+=T[I1][J1+1];

                           if((J2+1)==5)
                             enmsg+=T[I2][0];
                           else
                             enmsg+=T[I2][J2+1];
                           break;
                     case 2:
                           if((I1+1)==5)
                               enmsg+=T[0][J1];
                           else
                               enmsg+=T[I1+1][J1];
                           if((I2+1)==5)
                                enmsg+=T[0][J2];
                           else
                                enmsg+=T[I2+1][J2];
                           break;

                  case 3:
```

```
                                    enmsg+=T[I1][J2];
                                    enmsg+=T[I2][J1];
                                    break;
                        }//switch end
          break;
         }//if end
       }//i loop end
 m=m+2;
 flag1=0;
 flag2=0;
 }while(m<msg.length());
System.out.println("\n\nPlayfair Cipher Text:\n\t"+ enmsg);
m=0;
//decryption
flag1=0;
 flag2=0;
do
 {
   for(i=0;i<5;i++)
   {
        for(j=0;j<5;j++)
         {
              if(T[i][j]==enmsg.charAt(m))
               {
                I1=i;
                J1=j;
                flag1=1;
              }
              if(T[i][j]==enmsg.charAt(m+1))
               {
                I2=i;
                J2=j;
                flag2=1;
                }
              if(flag1==1 && flag2==1)
                break;
         }
    if(flag1==1 && flag2==1)
         {
              if(I1==I2)
               c=1;
              else if(J1==J2)
                 c=2;
              else
                 c=3;
            switch(c)
                 {
                     case 1:
                             if(J1==0)
                              plainmsg+=T[I1][4];
```

```
                                    else
                                       plainmsg+=T[I1][J1-1];
                                     if(J2==0)
                                     plainmsg+=T[I2][4];
                                     else
                                    plainmsg+=T[I2][J2-1];
                                     break;
                                   case 2:
                                      if(I1==0)
                                       plainmsg+=T[4][J1];
                                      else
                                          enmsg+=T[I1-1][J1];
                                      if(I2==0)
                                          plainmsg+=T[4][J2];
                                      else
                                           plainmsg+=T[I2-1][J2];
                                      break;
                        case 3:
                                    plainmsg+=T[I1][J2];
                                    plainmsg+=T[I2][J1];
                                      break;
                        }//switch end
      break;
        }//if end
   }//i loop end
 m=m+2;
 flag1=0;
 flag2=0;
 }while(m<enmsg.length());
System.out.println("\n\nPlayfair Plain Text:\n\t"+ plainmsg);
}
}
```

OUTPUT:

D:\Java\jdk1.6.0\bin>javac Playfaircipher1.java

D:\Java\jdk1.6.0\bin>java Playfaircipher1

enter the msg:

cryptography

enter the key:

security

The matrix:

s e c u r

i t y a b

d f g h k

l m n o p

q v w x z

Playfair Cipher Text:

usbnamkcboga

Playfair Plain Text:

cryptography

RESULT:

Thus a java program to perform encryption and decryption using Playfair cipher algorithm was executed successfully.

HILL CIPHER

EX.NO: 1.3

DATE:

AIM:

To write a java program to perform encryption using hill cipher algorithm.

ALGORITHM:

This encryption algorithm takes m successive plaintext letters and substitutes for them m ciphertext letters

$$\mathbf{C} = PK \bmod 26$$

$$(c1\ c2\ c3) = (p1\ p2\ p3)\begin{pmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{pmatrix} \bmod 26$$

$$c1 = (k_{11}p_1 + k_{21}p_2 + k_{31}p_3) \bmod 26$$
$$c2 = (k_{12}p_1 + k_{22}p_2 + k_{32}p_3) \bmod 26$$
$$c3 = (k_{13}p_1 + k_{23}p_2 + k_{33}p_3) \bmod 26$$

$$P = D(K, C) = CK^{-1} \bmod 26 = PKK^{-1} = P$$

PROGRAM:

```
import java.io.*;
import java.lang.*;
public class hillcipher
{
public static void main(String []arg)throws Exception
{
    int k[][]={{17,17,5}, {21,18,21}, {2,2,19}};
    int p[]=new int[300];
    int c[]=new int[300];
    int i=0;
    BufferedReader br=new BufferedReader(new
InputStreamReader(System.in));
     System.out.println("enter plain text");
    String str=br.readLine();
     for( i=0;i<str.length();i++)
    {
        int c1=str.charAt(i);
       p[i]=(c1)-97;
    }
          i=0;int zz=0;
        for( int b=0;b<str.length()/3;b++)
        {
```

```
        for(int j=0;j<3;j++)
        {
        for(int x=0;x<3;x++)
        {
        c[i]+=k[j][x]*p[x+zz];
        }i++;
        }
        zz+=3;
        }
                System.out.println("Encrypted Text : ");
        for(int z=0;z<str.length();z++)
        System.out.print((char)((c[z]%26)+97));
 }
}
```

OUTPUT:

C:\Java\jdk1.6.0\bin>javac hillcipher.java

C:\Java\jdk1.6.0\bin>java hillcipher

enter plain text

paymoremoney

Encrypted Text :

lnshdlewmtrw

RESULT:

Thus a java program to perform encryption using hill cipher algorithm was executed successfully.

EX.NO: 1.4 **VIGENERE CIPHER**

DATE:

AIM:

To write a java program to perform encryption and decryption using vigenere cipher technique.

ALGORITHM:

- Read the plaintext text and keyword
- To encrypt a message, a key is needed that is as long as the message. Usually, the key is a repeating keyword.
- The first letter of the key is added to the first letter of the plaintext, mod 26,the second letters are added, and so on through the first m letters of the plaintext.
- Encryption process
  - $C_i = (p_i + k_{i \bmod m}) \bmod 26$
- Decryption process
  - $p_i = (C_i - k_{i \bmod m}) \bmod 26$

PROGRAM:

```
import java.util.*;
class Vigeneree
{
  public static void main(String[] args)
  {
    Scanner sc=new Scanner(System.in);
    String alpha="abcdefghijklmnopqrstuvwxyz";
    char T[][]=new char[26][26];
    String key=new String();
    String msg=new String();
    System.out.println("\nEnter the key and plaintext");
    key=sc.next();
    msg=sc.next();
    String enmsg="";
    String plmsg="";
    int i,j,count,m=0,h=1;
    for(i=0;i<26;i++)
    {
     count=i;
     for(j=0;j<26;j++)
     {
       T[i][j]=alpha.charAt(count);
       count=(count+1)%26;
     }
    }
          //key generation
```

```
  int len=msg.length();
  int klen=key.length();
  count=0;
  while(klen<len)
  {
   key+=key.charAt(count);
   count++;
   klen++;
  }
  System.out.println("Message   :\t"+msg);
  System.out.println("Key Text  :\t"+key);
          //encryption
  while(m<len)
  {
    i=alpha.indexOf(key.charAt(m));
    j=alpha.indexOf(msg.charAt(m));
    enmsg+=T[i][j];
    m++;
  }
  System.out.println("Cipher Text:\t"+enmsg);
          //decryption
  m=0;
  while(m<len)
  {
   i=0;
   j=alpha.indexOf(key.charAt(m));
   char k=enmsg.charAt(m);
   while(h==1)
   {
    if(T[i][j]==k)
    break;
    else
    i++;
   }
   plmsg+=alpha.charAt(i);
   m++;
  }
   System.out.println("Plain Text:\t"+plmsg);
}
 }
```

OUTPUT:

```
C:\Java\jdk1.6.0\bin>javac Vigenere.java
C:\Java\jdk1.6.0\bin>java Vigenere
Enter the key and plaintext
deceptive
wearediscoveredsaveyourself
Message   :     wearediscoveredsaveyourself
Key Text  :     deceptivedeceptivedeceptive
Cipher Text:    zicvtwqngrzgvtwavzhcqyglmgj
Plain Text:     wearediscoveredsaveyourself
```

RESULT:

Thus a java program to perform encryption and decryption using vigenere cipher technique was executed successfully.

EX.NO:2.1

RAIL FENCE

DATE:

AIM:

To write a java program to perform encryption and decryption using rail fence cipher technique.

ALGORITHM:

- Read the plaintext
- The plaintext is written down as a sequence of diagonals and then read off as a sequence of rows.
- Read the character array, the characters at even positions are stored in an array and characters at odd positions are stored in another array.
- Both the arrays are concatenated and displayed

PROGRAM:

```
import java.util.*;
class Rail
{
  public static void main(String[] args)
  {
  Scanner sc=new Scanner(System.in);
  String msg=new String();
  System.out.println("Enter the message:");
  msg=sc.next();
  int i,ptr;
  char a[]=new char[20];
  char b[]=new char[20];
  String enmsg="";
  String pltxt="";
   i=0;
  for(ptr=0;ptr<msg.length();ptr=ptr+2)
  {
  a[i]=msg.charAt(ptr);
  i++;
  }
    i=0;
  for(ptr=1;ptr<msg.length();ptr=ptr+2)
  {
  b[i]=msg.charAt(ptr);
  i++;
  }
  i=0;
  for(ptr=0;ptr<msg.length()/2;ptr++)
```

```
  {
  enmsg+=a[i];
  i++;
  }
  i=0;
  for(ptr=0;ptr<msg.length()/2;ptr++)
  {
  enmsg+=b[i];
  i++;
  }
  System.out.println("Cipher Text:" +enmsg);
  i=0;
  for(ptr=0;ptr<msg.length()/2;ptr++)
  {
  pltxt+=a[i];
  pltxt+=b[i];
  i++;
  }
  System.out.println("Plain Text:" +pltxt);
  }
}
```

OUTPUT

C:\Java\jdk1.6.0\bin>javac Rail.java

C:\Java\jdk1.6.0\bin>java Rail

Enter the message:

meetmeafterthetogaparty

Cipher Text:mematrhtgpretefeteoaat

Plain Text:meetmeafterthetogapart

RESULT:

Thus a java program to perform encryption and decryption using rail fence cipher technique was executed successfully.

EX.NO:2.2 # ROW COLUMN TRANSPOSITION

DATE:

AIM:

To write a java program to perform encryption and decryption using row column transposition technique.

ALGORITHM:

- Read the plaintext
- The plaintext is written down as a sequence of row by row
- Then read in the sequence column by column.
- The Key is used to read the column order
- The read character array is displayed as encrypted message.
- Decrypt the encrypted message with the same key.
- Display the decrypted message.

PROGRAM:

```
import java.util.Scanner;
public class ColTransCipher {
   private static Scanner in;
   public static void main(String[] args){
      System.out.println("Columnar Transposition Cipher");
      in = new Scanner(System.in);
      System.out.print("1. Encryption\n2. Decryption\nChoose(1,2): ");
      int choice = in.nextInt();
      in.nextLine();
      if (choice == 1){
         System.out.println("Encryption");
         encryption();
      } else if (choice == 2){
         System.out.println("Decryption");
         decryption();
         System.out.println("Invalid Choice");
         System.exit(0);
      }
   }
   private static void encryption(){
      System.out.print("Enter Message: ");
```

```
        String plainText = in.nextLine().toUpperCase().replace(" ", "");
        StringBuilder msg = new StringBuilder(plainText);
        System.out.print("Enter Keyword: ");
        String keyword = in.nextLine().toUpperCase();
        int[] kywrdNumList = keywordNumAssign(keyword);
        for (int i = 0, j = 1; i < keyword.length(); i++, j++) {
            System.out.print(keyword.substring(i, j) + " ");
        }
        System.out.println();

        for (int i: kywrdNumList){
            System.out.print(i + " ");
        }
        System.out.println();
        System.out.println("-------------------------");
        int extraLetters = msg.length() % keyword.length();
        //System.out.println(extraLetters);
        int dummyCharacters = keyword.length() - extraLetters;
//          System.out.println(dummyCharacters);
        if (extraLetters != 0){
            for (int i = 0; i < dummyCharacters; i++) {
                msg.append(".");
            }
        }
        int numOfRows = msg.length() / keyword.length();
        char[][] arr = new char[numOfRows][keyword.length()];
        int z = 0;
        for (int i = 0; i < numOfRows; i++) {
            for (int j = 0; j < keyword.length(); j++) {
                arr[i][j] = msg.charAt(z);
                z++;
        }

        for (int i = 0; i < numOfRows; i++) {
            for (int j = 0; j < keyword.length(); j++) {
                System.out.print(arr[i][j] + " ");
            }
            System.out.println();
        }
```

```
    StringBuilder cipherText = new StringBuilder();
    System.out.println();
    String numLoc = getNumberLocation(keyword, kywrdNumList);
    System.out.println("Location of numbers: " + numLoc);
    System.out.println();
    for (int i = 0, k = 0; i < numOfRows; i++, k++) {
        int d;
        if (k == keyword.length()){
            break;
        } else {
            d = Character.getNumericValue(numLoc.charAt(k));
        }
        for (int j = 0; j < numOfRows; j++) {
            cipherText.append(arr[j][d]);
        }
    }
    System.out.println("Cipher Text: " + cipherText);
}
private static void decryption(){
    System.out.print("Enter Message: ");
    String msg = in.nextLine().toUpperCase().replace(" ", "");
    System.out.print("Enter Keyword: ");
    String keyword = in.nextLine().toUpperCase();
    int numOfRows = msg.length() / keyword.length();
    char[][] arr = new char[numOfRows][keyword.length()];
    int[] kywrdNumList = keywordNumAssign(keyword);
    String numLoc = getNumberLocation(keyword, kywrdNumList);
    for (int i = 0, k = 0; i < msg.length(); i++, k++) {
        int d = 0;
        if (k == keyword.length()){
            k = 0;
        } else {
            d = Character.getNumericValue(numLoc.charAt(k));
        }

        for (int j = 0; j < numOfRows; j++, i++) {
            arr[j][d] = msg.charAt(i);
        } // for loop
        --i;
    }
```

```
        StringBuilder plainText = new StringBuilder();

        for (int i = 0; i < numOfRows; i++) {
            for (int j = 0; j < keyword.length(); j++) {
                plainText.append(arr[i][j]);
            }
        }
        System.out.println("Plain Text: " + plainText);
    }
    private static String getNumberLocation(String keyword, int[]
kywrdNumList) {
        String numLoc = "";
        for (int i = 1; i < keyword.length() + 1; i++) {
            for (int j = 0; j < keyword.length(); j++) {
                if (kywrdNumList[j] == i){
                    numLoc += j;
                }
            }
        }
        return numLoc;
    }
    private static int[] keywordNumAssign(String keyword) {
        String alpha = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
        int[] kywrdNumList = new int[keyword.length()];

        int init = 0;
        for (int i = 0; i < alpha.length(); i ++){
            for (int j = 0; j < keyword.length(); j++) {
                if (alpha.charAt(i) == keyword.charAt(j)){
                    init++;
                    kywrdNumList[j] = init;
                }
            }
        }
        return kywrdNumList;
    }
```

OUTPUT :

C:\Users\placement\Desktop\New folder (2)>java ColTransCipher

Columnar Transposition Cipher

1. Encryption

2. Decryption

Choose(1,2): 1

Encryption

Enter Message: I LIKE POTATOES BECAUSE THEY ARE TASTY

Enter Keyword: POTATO

P O T A T O

4 2 5 1 6 3

I L I K E P

O T A T O E

S B E C A U

S E T H E Y

A R E T A S

T Y . . . .

Location of numbers: 315024

Cipher Text: KTCHT.LTBERYPEUYS.IOSSATIAETE.EOAEA.

C:\Users\placement\Desktop\New folder (2)>java ColTransCipher

Columnar Transposition Cipher

1. Encryption

2. Decryption

Choose(1,2): 2

Decryption

Enter Message: KTCHT.LTBERYPEUYS.IOSSATIAETE.EOAEA.

Enter Keyword: POTATO

Plain Text: ILIKEPOTATOESBECAUSETHEYARETASTY....

RESULT :

Thus a java program to perform encryption and decryption using Row Column Transposition technique has been executed successfully.

EX.NO:3 **DES**

DATE:

AIM:

To write a java program to perform encryption using Data Encryption Standard (DES) .algorithm.

ALGORITHM:

- Initial permutation rearranging the bits to form the "permuted input".
- Followed by 16 iterations of the same function (substitution and permutation).
- The output of the last iteration consists of 64 bits which is a function of the plaintext and key. The left and right halves are swapped to produce the preoutput.
- Finally, the preoutput is passed through a permutation which is simply the inverse of the initial permutation (IP). The output of IP^{-1} is the 64-bit ciphertext.
- Initially the key is passed through a permutation function
- For each of the 16 iterations, a subkey ($\mathbf{K}_i$) is produced by a combination of a left circular shift and a permutation which is the same for each iteration. However, the resulting sub key is different for each iteration because of repeated shifts

PROGRAM:

```
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
import javax.crypto.BadPaddingException;
import javax.crypto.Cipher;
import javax.crypto.IllegalBlockSizeException;
import javax.crypto.KeyGenerator;
import javax.crypto.NoSuchPaddingException;
import javax.crypto.SecretKey;
public class DESS
 {
 private static Cipher encryptCipher;
 private static Cipher decryptCipher;
 public static void main(String[] args)
{
  try {
   KeyGenerator keygenerator = KeyGenerator.getInstance("DES");
   SecretKey secretKey = keygenerator.generateKey();
   encryptCipher = Cipher.getInstance("DES/ECB/PKCS5Padding");
   encryptCipher.init(Cipher.ENCRYPT_MODE, secretKey);
   byte[] encryptedData = encryptData("Classified Information!");
   decryptCipher = Cipher.getInstance("DES/ECB/PKCS5Padding");
   decryptCipher.init(Cipher.DECRYPT_MODE, secretKey);
   decryptData(encryptedData);
```

```
  }
catch (NoSuchAlgorithmException e)
{
   e.printStackTrace();
  }
catch (NoSuchPaddingException e)
{
   e.printStackTrace();
  }
catch (InvalidKeyException e) {
   e.printStackTrace();
  }
catch (IllegalBlockSizeException e) {
   e.printStackTrace();
  }
catch (BadPaddingException e) {
   e.printStackTrace();
  }
 }
 // Encrypt Data
 private static byte[] encryptData(String data)   throws
IllegalBlockSizeException, BadPaddingException {
  System.out.println("Data Before Encryption :" + data);
  byte[] dataToEncrypt = data.getBytes();
  byte[] encryptedData = encryptCipher.doFinal(dataToEncrypt);
  System.out.println("Encryted Data: " + encryptedData);
  return encryptedData;
 }
 // Decrypt Data
   private static void decryptData(byte[] data)   throws
IllegalBlockSizeException, BadPaddingException
{
  byte[] textDecrypted = decryptCipher.doFinal(data);
  System.out.println("Decryted Data: " + new
String(textDecrypted));
 }
}
```

OUTPUT:

```
D:\Java\jdk1.6.0\bin>java DESS
Data Before Encryption :Classified Information!
Encryted Data: [B@4e79f1
Decryted Data: Classified Information!
```

RESULT:

Thus a java program to perform encryption using Data Encryption Standard(DES) algorithm was executed successfully.

EX.NO: 4 # IMPLEMENTATION OF AES IN JAVA

Date:

AIM:

To write a Java program to implement AES Algorithm.

ALGORITHM:

AES steps of encryption:

1. Derive the set of round keys from the cipher key.
2. Initialize the state array with the block data (plaintext).
3. Add the initial round key to the starting state array.
4. Perform nine rounds of state manipulation.
5. Perform the tenth and final round of state manipulation.
6. Copy the final state array out as the encrypted data (ciphertext).

PROGRAM:

```
package com.javapapers.java.security;

import java.util.Base64;

import javax.crypto.Cipher;

import javax.crypto.KeyGenerator;

import javax.crypto.SecretKey;

public class EncryptionDecryptionAES {

    static Cipher cipher;

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

        KeyGenerator keyGenerator = KeyGenerator.getInstance("AES");

        keyGenerator.init(128);
```

```
        SecretKey secretKey = keyGenerator.generateKey();
        cipher = Cipher.getInstance("AES");
        String plainText = "AES Symmetric Encryption Decryption";
        System.out.println("Plain Text Before Encryption: " + plainText);
        String encryptedText = encrypt(plainText, secretKey);
        System.out.println("Encrypted Text After Encryption: " + encryptedText);
        String decryptedText = decrypt(encryptedText, secretKey);
        System.out.println("Decrypted Text After Decryption: " + decryptedText);
}
public static String encrypt(String plainText, SecretKey secretKey)
                throws Exception {
        byte[] plainTextByte = plainText.getBytes();
        cipher.init(Cipher.ENCRYPT_MODE, secretKey);
        byte[] encryptedByte = cipher.doFinal(plainTextByte);
        Base64.Encoder encoder = Base64.getEncoder();
        String encryptedText = encoder.encodeToString(encryptedByte);
        return encryptedText;
}
public static String decrypt(String encryptedText, SecretKey secretKey)
                throws Exception {
        Base64.Decoder decoder = Base64.getDecoder();
        byte[] encryptedTextByte = decoder.decode(encryptedText);
        cipher.init(Cipher.DECRYPT_MODE, secretKey);
        byte[] decryptedByte = cipher.doFinal(encryptedTextByte);
        String decryptedText = new String(decryptedByte);
```

```
        return decryptedText;
    }
}
```

OUTPUT:

```
Plain Text Before Encryption: AES Symmetric Encryption Decryption

Encrypted Text After Encryption:
sY6vkQrWRg0fvRzbqSAYxepeBIXg4AySj7Xh3x4vDv8TBTkNiTfca7wW/dxiMMJl

Decrypted Text After Decryption: AES Symmetric Encryption Decryption
```

RESULT:

Thus a Java program to implement AES Algorithm was executed successfully.

EX.NO: 5

DATE:

RSA ALGORITHM

AIM:

To write a Java program to implement RSA Algorithm.

ALGORITHM:

Generate public and private key

- Select p, q such that p and q both prime, p is not equal to q
- Calculate n = p * q
- Calculate $\Phi(n) = (p - 1)(q - 1)$
- Select integer e such that gcd $(\Phi(n), e) = 1; 1 \leq e \leq \Phi(n)$
- Calculate d such that $d = e\text{-}1 \pmod{\Phi(n)}$
- Public key PU = {e, n}
- Private key PR = {d, n}

Encryption using Public Key

- Choose Plaintext $M < n$
- Ciphertext: $C = M^e \bmod n$

Decryption using private Key

- Ciphertext: C
- Plaintext: $M = C^d \bmod n$

PROGRAM:

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.math.*;
import java.util.Random;
import java.util.Scanner;
public class RSA
 {
     static Scanner sc = new Scanner(System.in);
     public static void main(String[] args)
     {
          System.out.print("Enter a Prime number: ");
          BigInteger p = sc.nextBigInteger(); //  one prime
number
          System.out.print("Enter another prime number: ");
          BigInteger q = sc.nextBigInteger(); // another prime
number
          BigInteger n = p.multiply(q);
          BigInteger n2 =
p.subtract(BigInteger.ONE).multiply(q.subtract(BigInteger.ONE));
          BigInteger e = generateE(n2);
          BigInteger d = e.modInverse(n2); // Here's the
multiplicative inverse
```

```
            System.out.println("Encryption keys are: " + e + ", " +
n);
            System.out.println("Decryption keys are: " + d + ", " +
n);
            System.out.print("Enter message: ");
            BigInteger m=sc.nextBigInteger();
            BigInteger c=encrypt(m,e,n);
            System.out.println("Ciphertext: " +c);
            System.out.println("Plaintext: " + decrypt(c,d,n));
       }
      public static BigInteger generateE(BigInteger fiofn)
      {
      int y, intGCD;
      BigInteger e;
      BigInteger gcd;
      Random x = new Random();
      do
      {
            y = x.nextInt(fiofn.intValue()-1);
            String z = Integer.toString(y);
            e = new BigInteger(z);
            gcd = fiofn.gcd(e);
            intGCD = gcd.intValue();
      }
      while(y <= 2 || intGCD != 1);
      return e;
      }
      public static BigInteger encrypt(BigInteger m,BigInteger
e,BigInteger n)
      {
            BigInteger x;
            x=m.modPow(e,n);
            return x;
      }
      public static BigInteger decrypt(BigInteger c,BigInteger
d,BigInteger n)
      {
            BigInteger y;
            y=c.modPow(d,n);
            return y;
      }
}
```

OUTPUT:

C:\Java\jdk1.6.0\bin>javac RSA.java

C:\Java\jdk1.6.0\bin>java RSA

Enter a Prime number: 7

Enter another prime number: 11

Encryption keys are: 43, 77

Decryption keys are: 7, 77

Enter message: 6

Ciphertext: 62

Plaintext: 6

RESULT:

Thus a Java program to implement RSA Algorithm was executed successfully.

EX.NO:6

DATE:

Digital Signature Standard

AIM:

To write a java program to sign a document by using digital signature algorithm.

ALGORITHM:

- Generate a key pair
- Create a Signature object and initialize it with the private key
- Update and sign the data
- Now that all the data to be signed has been read in, generate a signature for it
- Save the signature in a file
- Save the public key in a file

PROGRAM

```
import java.io.*;
import java.security.*;
class GenSig {
     public static void main(String[] args) {
         /* Generate a DSA signature */
         if (args.length != 1) {
            System.out.println("Usage: GenSig nameOfFileToSign");
            }
        else try{
             /* Generate a key pair */
             KeyPairGenerator keyGen =
KeyPairGenerator.getInstance("DSA", "SUN");
            SecureRandom random =
SecureRandom.getInstance("SHA1PRNG", "SUN");
            keyGen.initialize(1024, random);
            KeyPair pair = keyGen.generateKeyPair();
            PrivateKey priv = pair.getPrivate();
            PublicKey pub = pair.getPublic();
            /* Create a Signature object and initialize it with
the private key */
            Signature dsa = Signature.getInstance("SHA1withDSA",
"SUN");
            dsa.initSign(priv);
            /* Update and sign the data */
             FileInputStream fis = new FileInputStream(args[0]);
            BufferedInputStream bufin = new
BufferedInputStream(fis);
            byte[] buffer = new byte[1024];
```

```
            int len;
            while (bufin.available() != 0) {
                len = bufin.read(buffer);
                dsa.update(buffer, 0, len);
                };
            bufin.close();
            /* Now that all the data to be signed has been read
in, generate a signature for it */
            byte[] realSig = dsa.sign();
            /* Save the signature in a file */
            FileOutputStream sigfos = new
FileOutputStream("sig");
            sigfos.write(realSig);
             sigfos.close();
             /* Save the public key in a file */
            byte[] key = pub.getEncoded();
            FileOutputStream keyfos = new
FileOutputStream("suepk");
            keyfos.write(key);
             keyfos.close();
         } catch (Exception e) {
            System.err.println("Caught exception " +
e.toString());
        }
     };
 }
```

OUTPUT

D:\Java\jdk1.6.0\bin>javac GenSig.java

D:\Java\jdk1.6.0\bin>java GenSig link.txt

RESULT:

Thus a java program to sign a document using digital signature algorithm was executed successfully.

Ex No: 7 **DEMONSTRATE INTRUSION DETECTION SYSTEM (IDS) USING SNORT**

Date:

SNORT can be configured to run in three modes:

1. Sniffer mode 2. Packet Logger mode 3. Network Intrusion Detection System mode

Sniffer mode

snort –v Print out the TCP/IP packets header on the screen

Snort –vd show the TCP/IP ICMP header with application data in transit.

Packet Logger mode

 snort –dev –l c:\log [create this directory in the C drive] and snort will automatically know to go into packet logger mode, it collects every

packet it sees and places it in log directory.

snort –dev –l c:\log –h ipaddress/24 This rule tells snort that you want to print out the data link and TCP/IP headers as well as application data into the log directory.

snort –l c:\log –b This is binary mode logs everything into a single file.

Network Intrusion Detection System mode

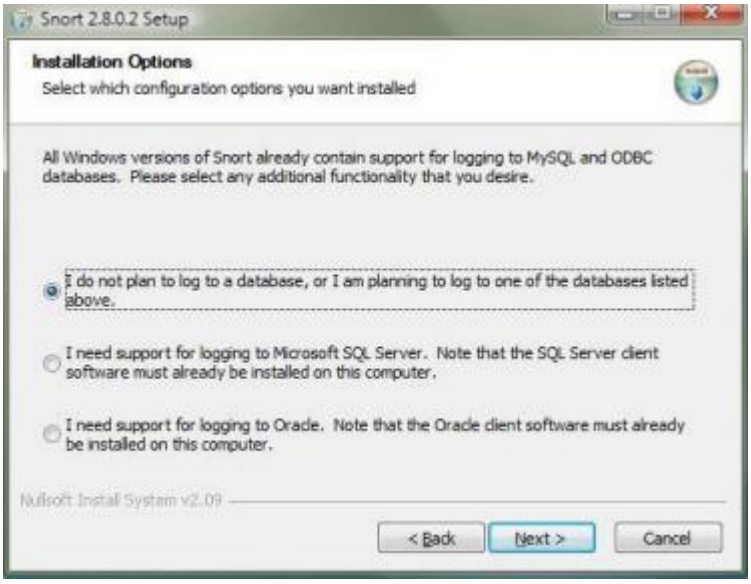
 snort –d c:\log –h ipaddress/24 –c snort.conf This is a configuration file applies rule to each packet to decide it an action based upon the rule type in the file.

Snort –d –h ipaddress/24 –l c:\log –c snort.conf

This will cnfigure snort to run in its most basic NIDS form, logging packets that trigger rules specifies in the snort.conf

Download SNORT from snort.org

Install snort with or without database support.



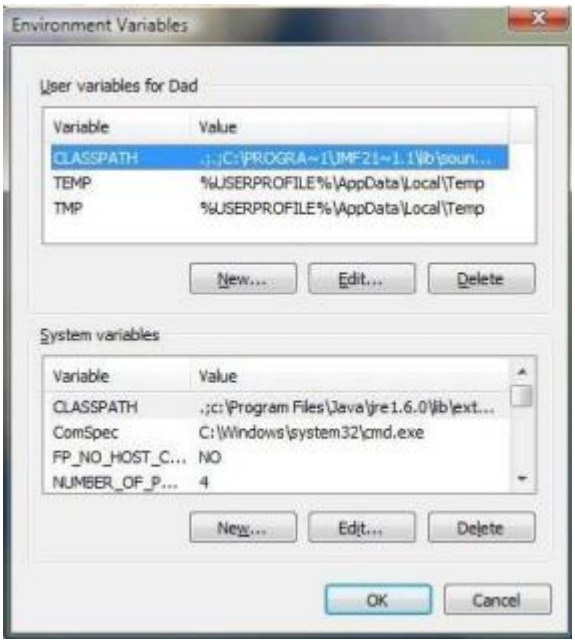
Select all the components and Click Next.

Install and Close.

Skip the WinPcap driver installation

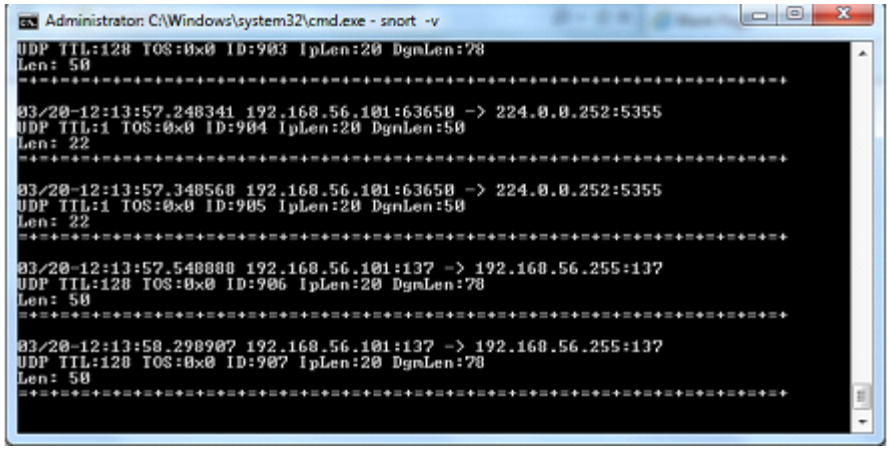
Add the path variable in windows environment variable by selecting new classpath.

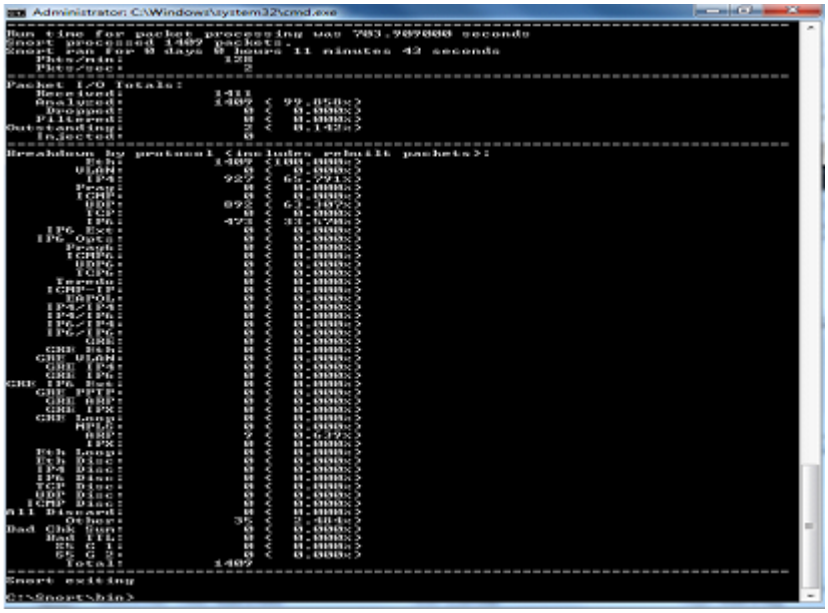
Create a path variable and point it at snort.exe variable name□path and variable value□c:\snort\bin.



Click OK button and then close all dialog boxes.

Open command prompt and type the following command





Ex.No.8
Date:

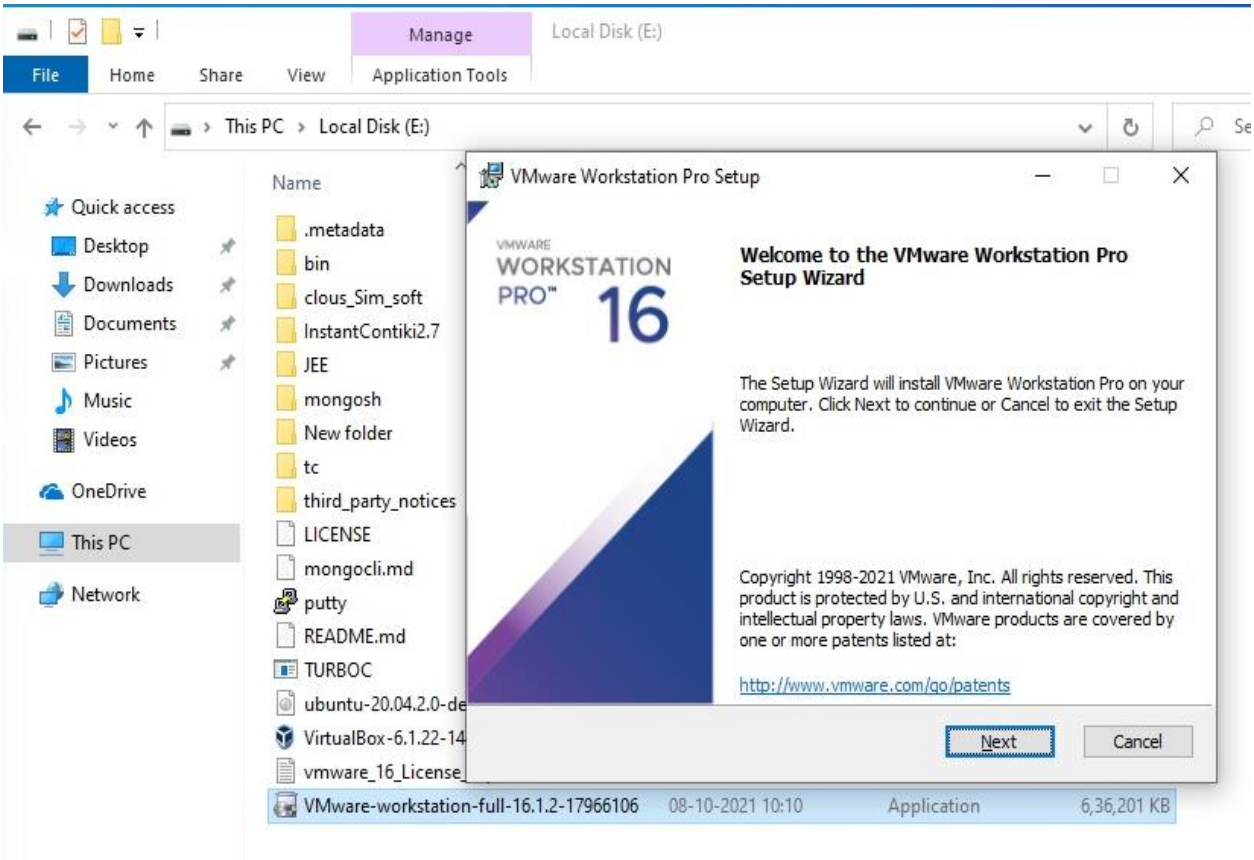
Install Virtualbox/VMware Workstation with different flavours of linux or windows OS on top of windows7 or 8 and execute Simple Programs

AIM:

To installVirtualbox/VMware workstation with different flavours of linux or windows OS on top of windows 7 or 8 and execute Simple C programs.

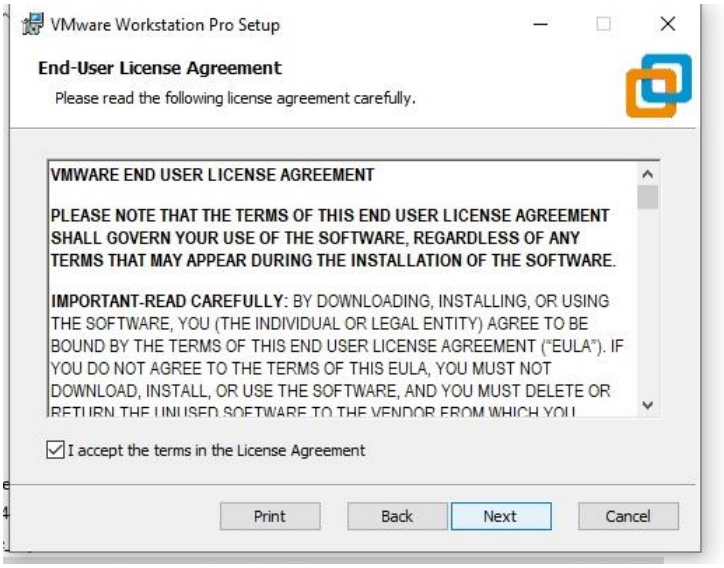
Steps to install VMware on windows 10/8/7

1. **Download** VMware-workstation-full-16.1.2 software from www.vmware.com
2. **Double-click** on downloaded **VMware-workstation-full-16.1.2-17966106.exe** file to bring up the welcome screen.
3. Click on **Next** button to start VMware installation Setup Wizard



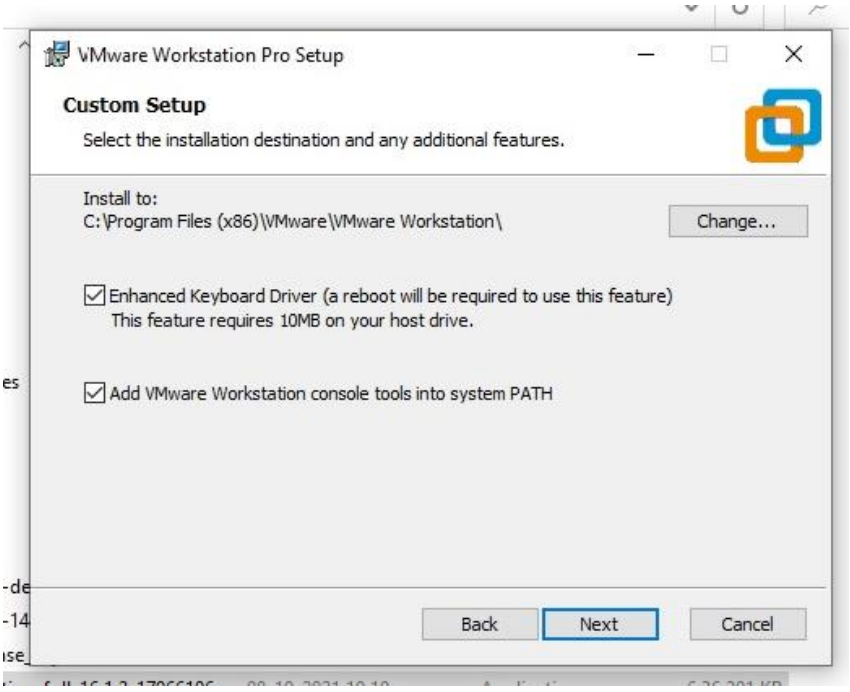
4. At this stage, you will see a check box to accept the license agreement.
   - **I Accept the terms in the License Agreement.**

   Click on the **check box** and Click **NEXT** button.



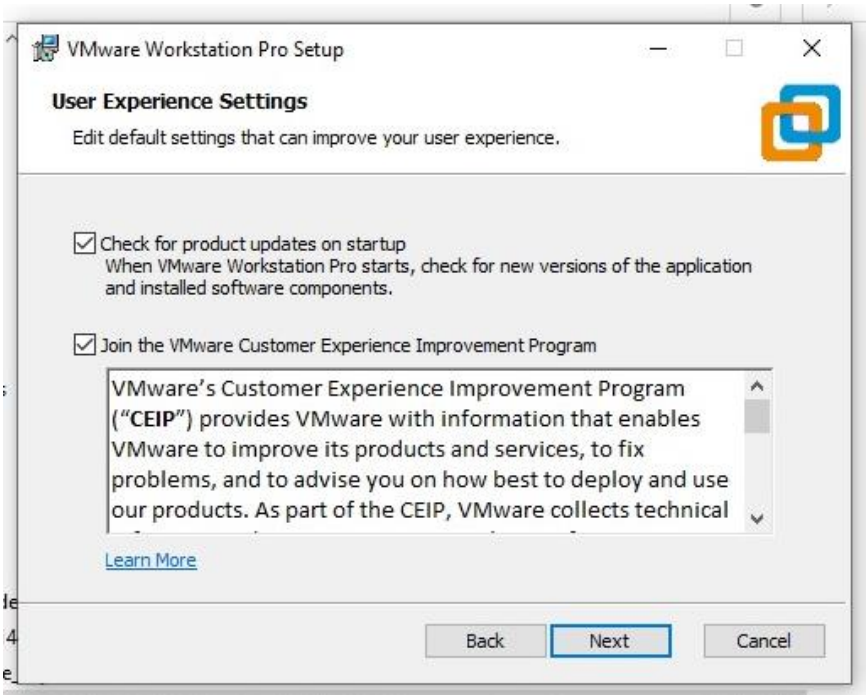
5. By default the VMware will install its core files in the C: drive. Incase you have low space on the C: drive , then just click on the **Change** button and select the location where you want to install it.However , if you are not acquainted with this option then simply leave it as default and then there are two check boxes ,
   - **Enhanced Keyboard Driver**
   - **Add VMware Workstation console tools into system PATH.**

   Click on the **check boxes** and click on **NEXT** button.



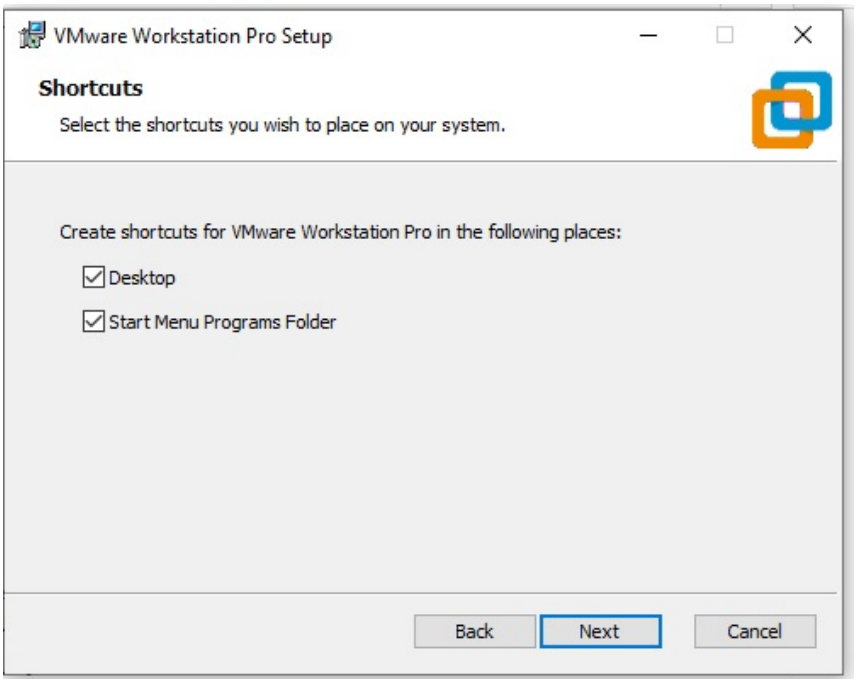
6. In the User Experience Settings there are 2 check boxes
    - **Check for product update on startup**
    - **Join the vmware Customer Experience Improvement Program**

    Click on the **check boxes** and click on **NEXT** button.

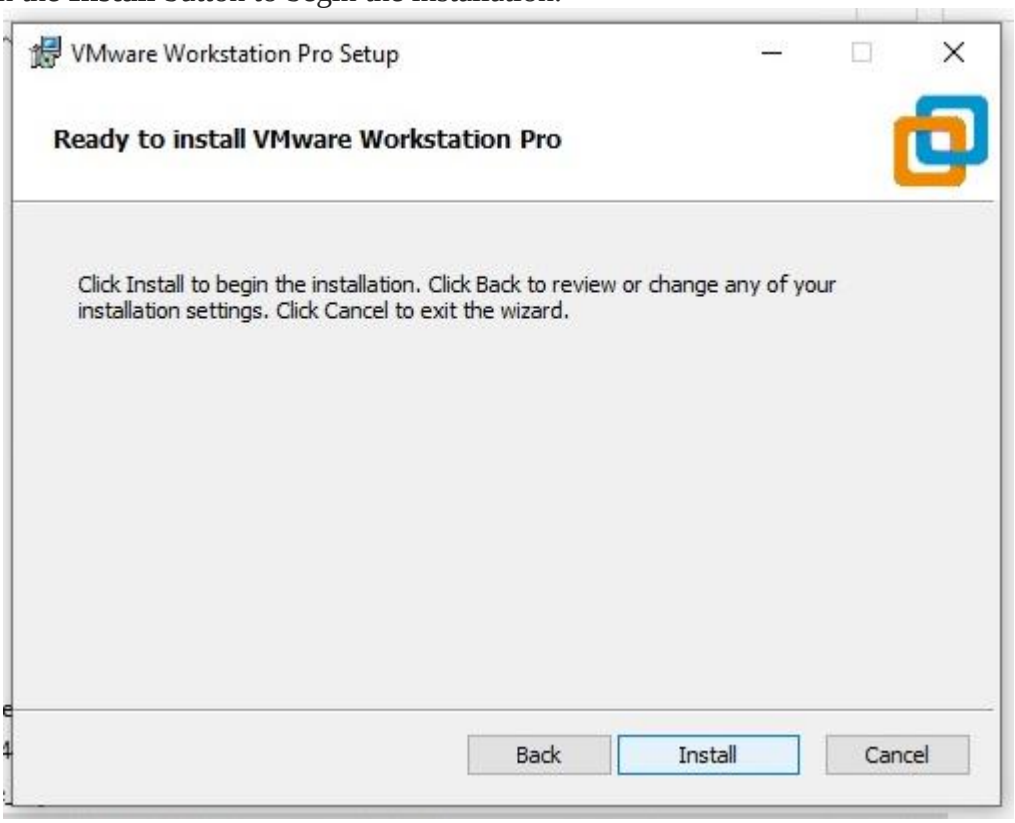


7. Next Shortcuts to be created ,
    - **Desktop :** To create shortcuts for vmware Workstation on desktop
    - **Start Menu Program Folder :** Will get added to the start menu.

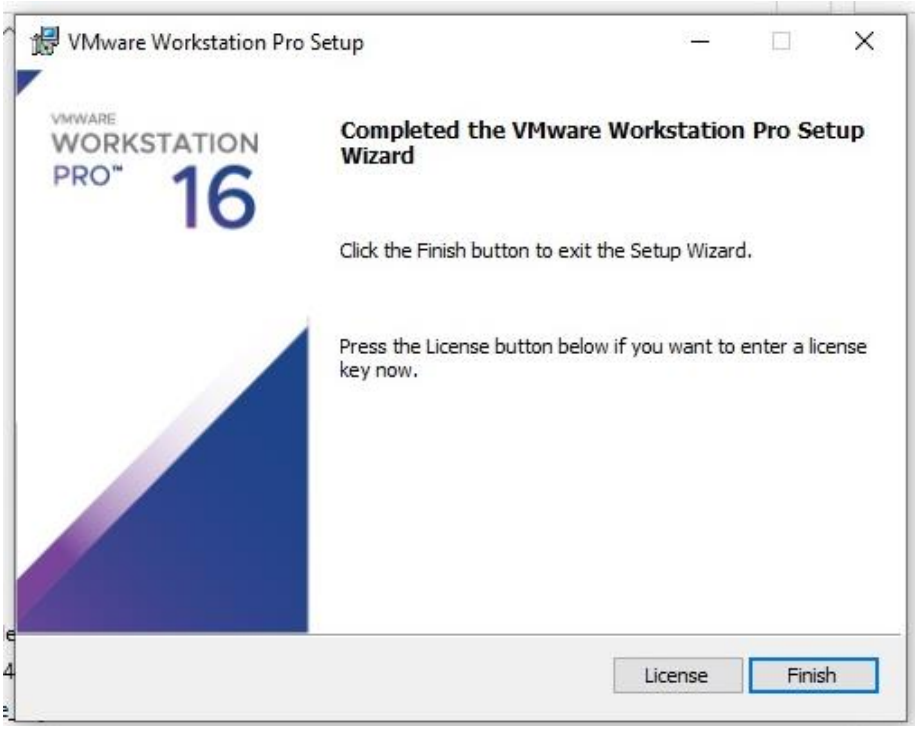
    Click on the check boxes and click on NEXT button.



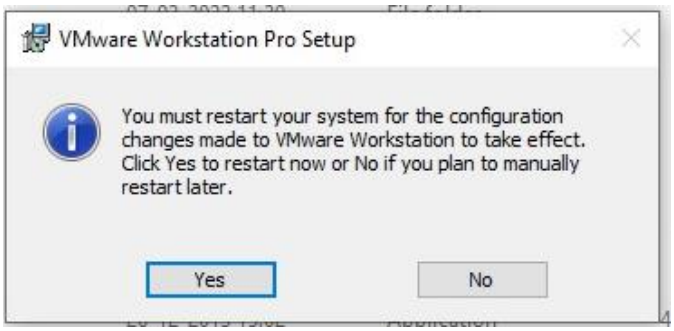
8. Clik on the **Install** button to begin the installation.



9. After installing, the installation wizard will show you a **Finish** button, click on that and it will ask to restart the system once the installation is over.



10. Click on **YES** button to restart the system.

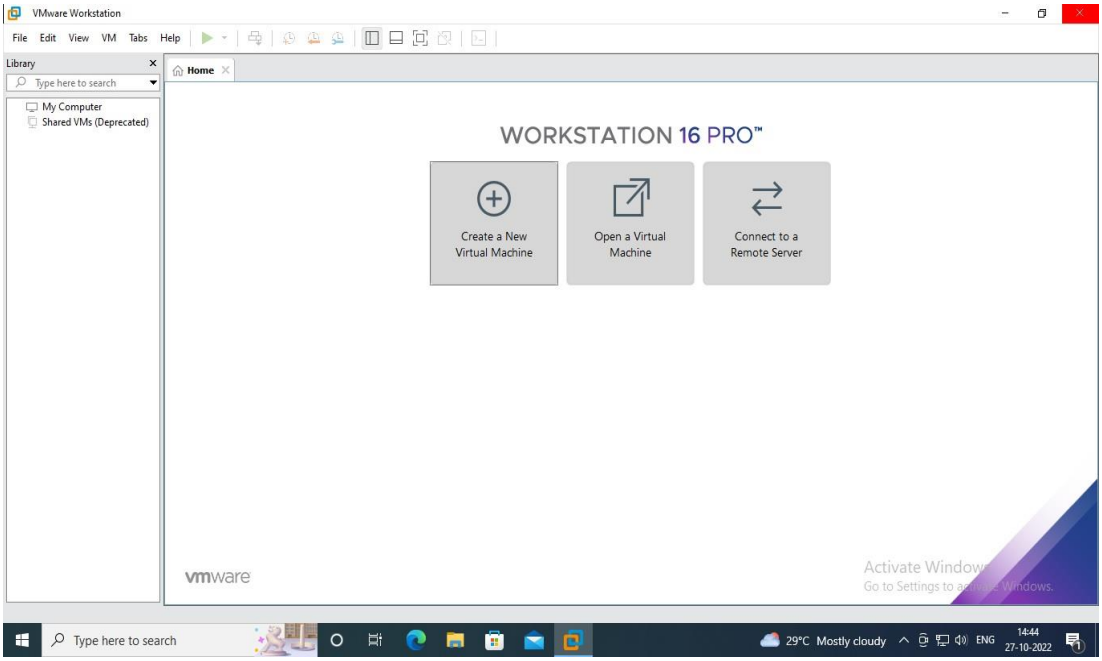


11. Once the system is restarted , double click on VMware workstation to open. Enter the license key mentioned below and click on **Continue** button.
    - **ZF3R0-FHED2-M80TY-8QYGC-NPKYF**

12. Click on Finish button and it will start the VMware on your Windows10/7/8 machines.

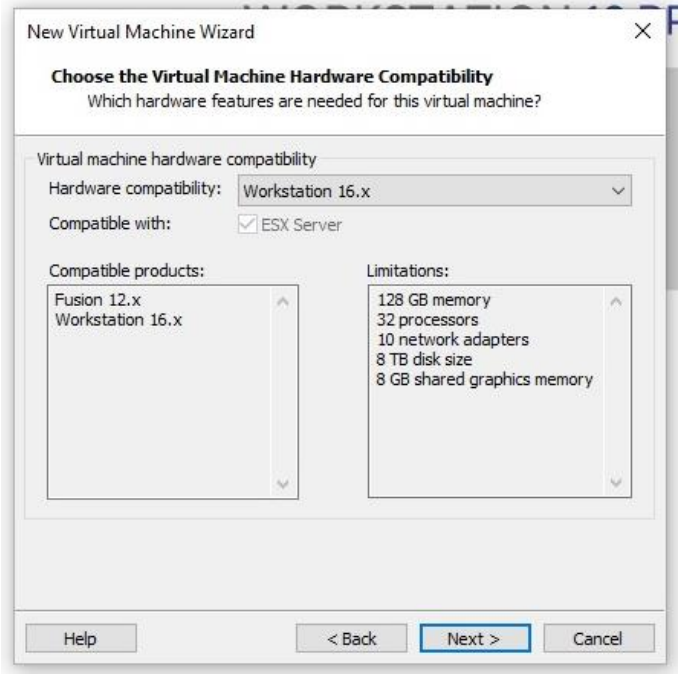
Install Ubuntu on VirtualBox:

1. Visit the page http://www.ubuntu.com/download/ubuntu/download
2. Run **VMware** by double-clicking the icon
3. Click "**Create a New Virtual Machine**" button at the center.

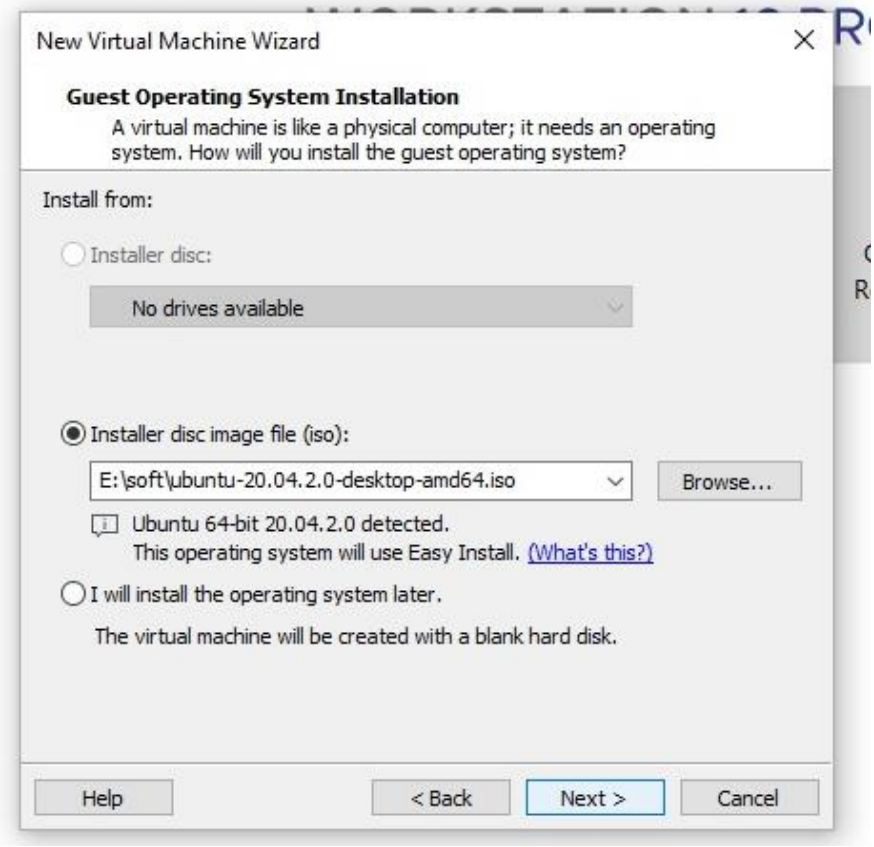


4. Select **Custom (advanced)** radio button and click on **NEXT** button.

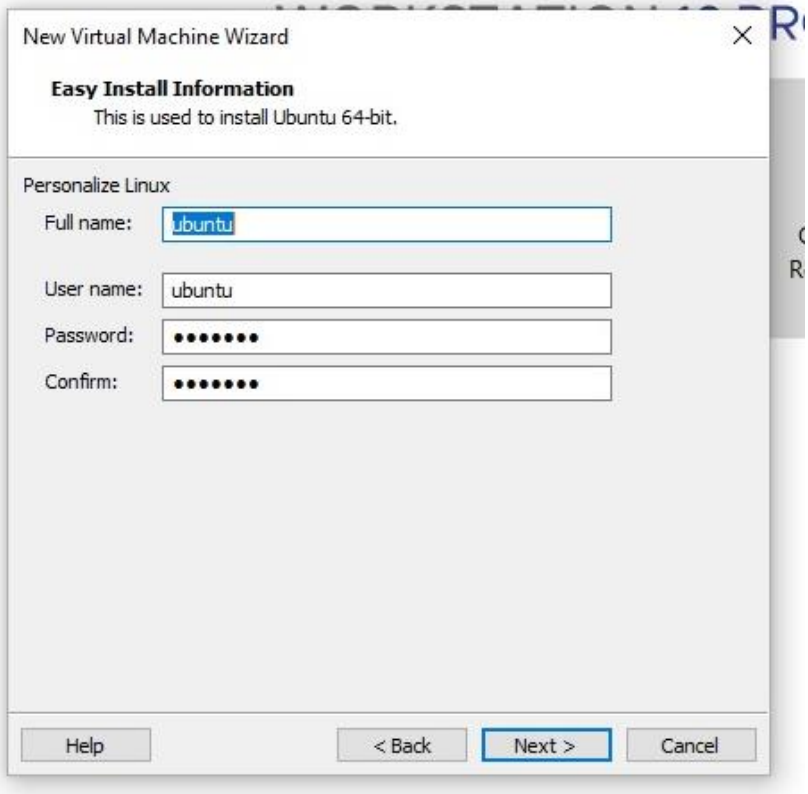
5. Then Click on **NEXT** Button



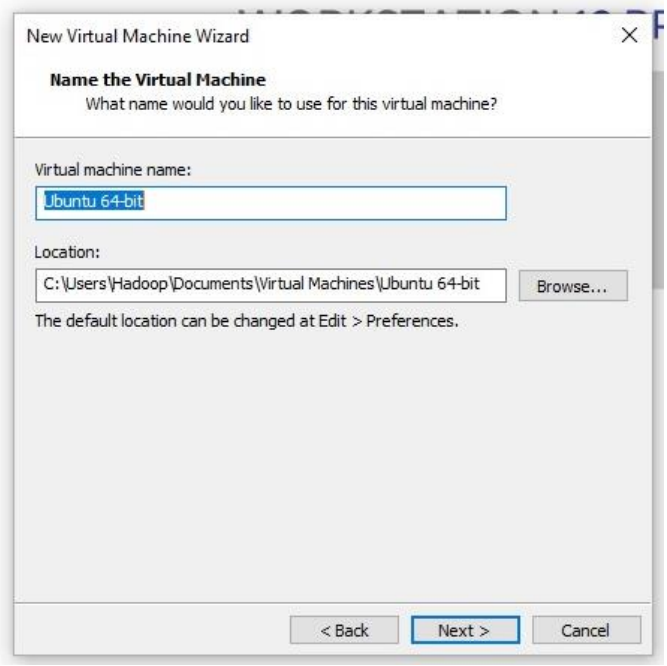
6. Click on the **Installer disk image file (iso) :** radio button and click on **Browse** button to give the **ubuntu-20.04.2.0-desktop-amd64.iso** path to install the OS. Click on **Next** button to proceed.



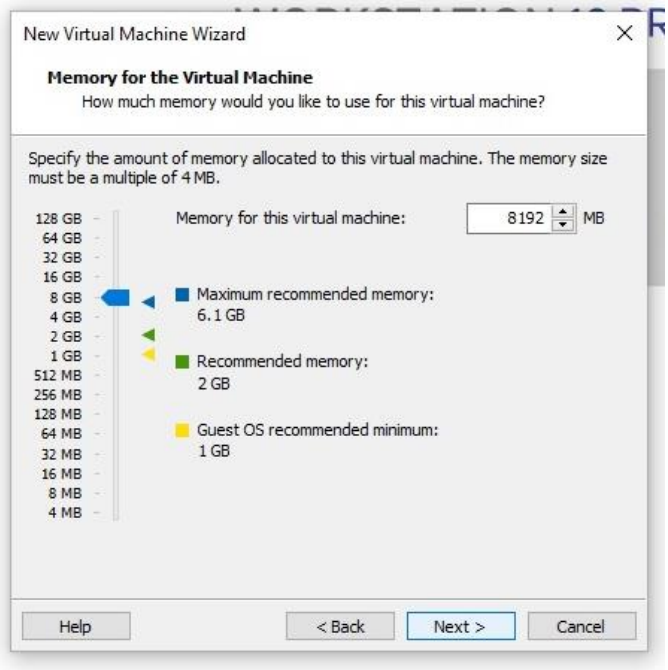
7. Enter the **Full name** as **ubuntu** , **User name** as **ubuntu** and enter the **password** as **cse123**. Then click on **NEXT** Button.



8. By default the virtual machine name and path will be displayed. In case you have low space on the C: drive , then just click on the Browse button and select the location where you want to install it. However , if you are not acquainted with this option then leave it as default and click on **NEXT** Button.

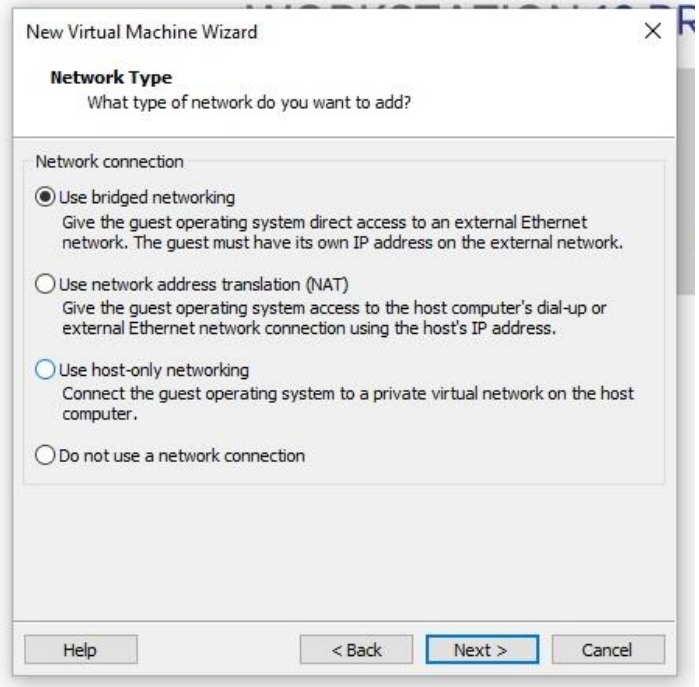


9. Choose the amount of memory as **8GB** and then click on **NEXT** button.

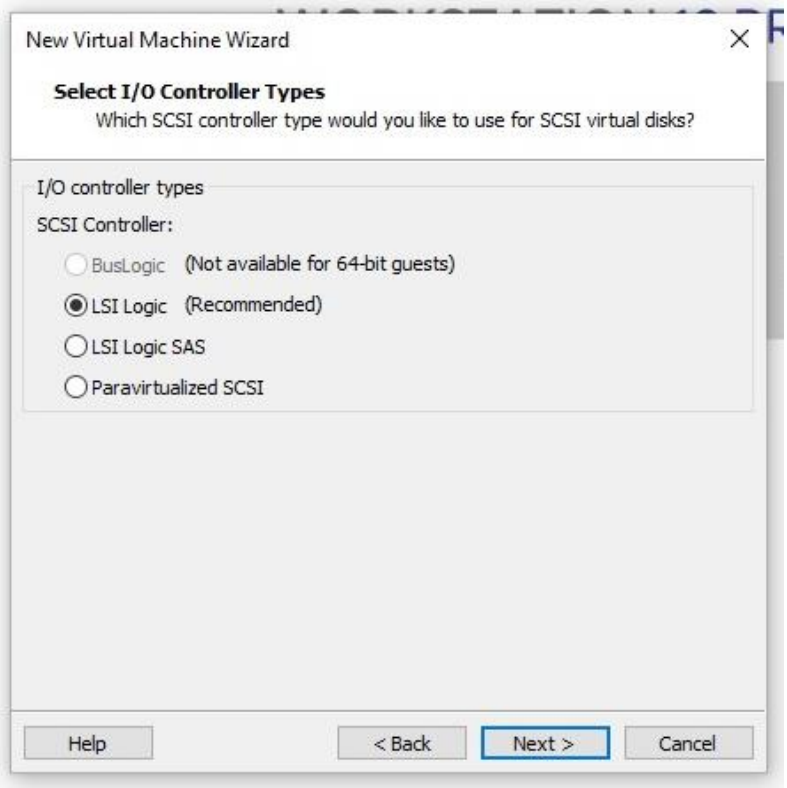


10. In the Network Type , Select the radio button mentioned below,
    - Use bridged networking – give the guest operating system direct access to an external Ethernet network. The guest must have its own IP address on the external network.

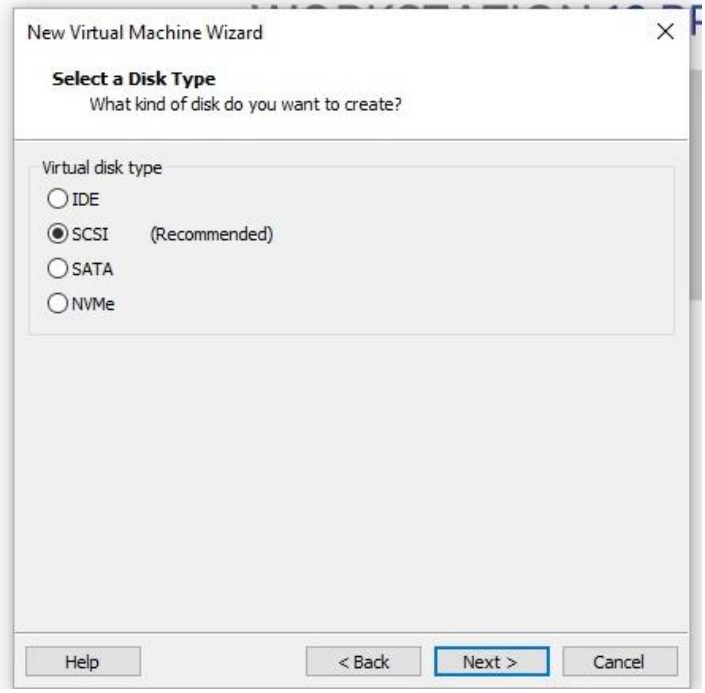
    Click on NEXT button to proceed.



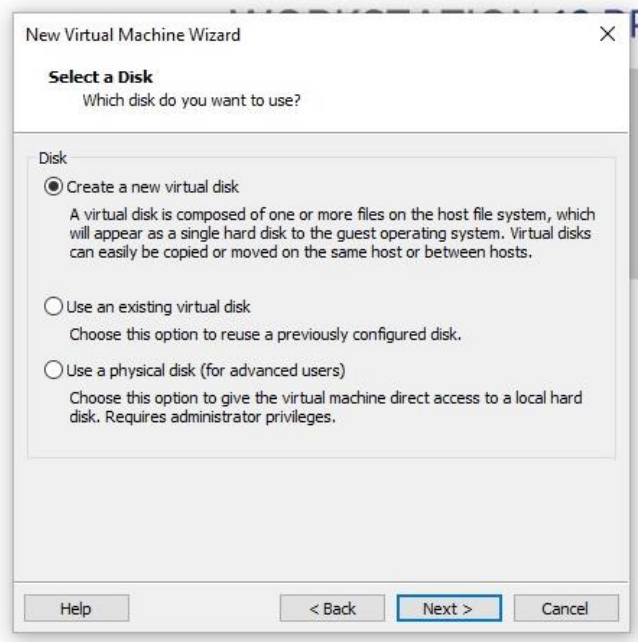
11. Select the I/O controller types as **LSI Logic** from the SCSI controllers mentioned and click on **Next** button.



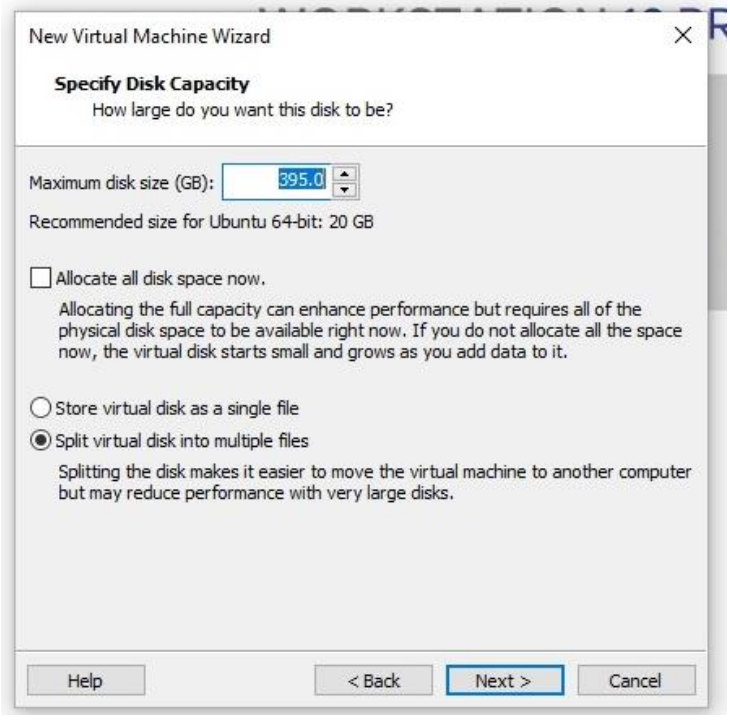
12. Under Virtual disk type select **SCSI** radio button and click on **Next** button.



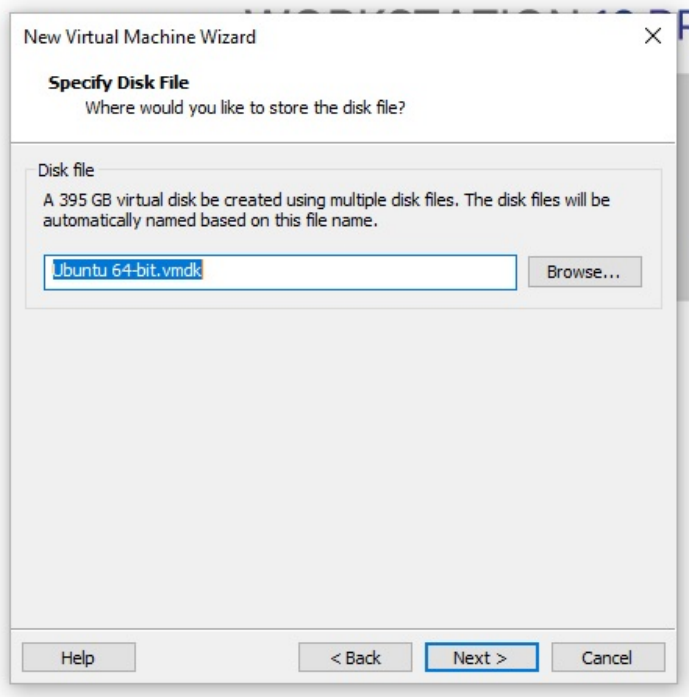
13. Under Disk type select **Create a new virtual disk** radio button and click on **Next** button.



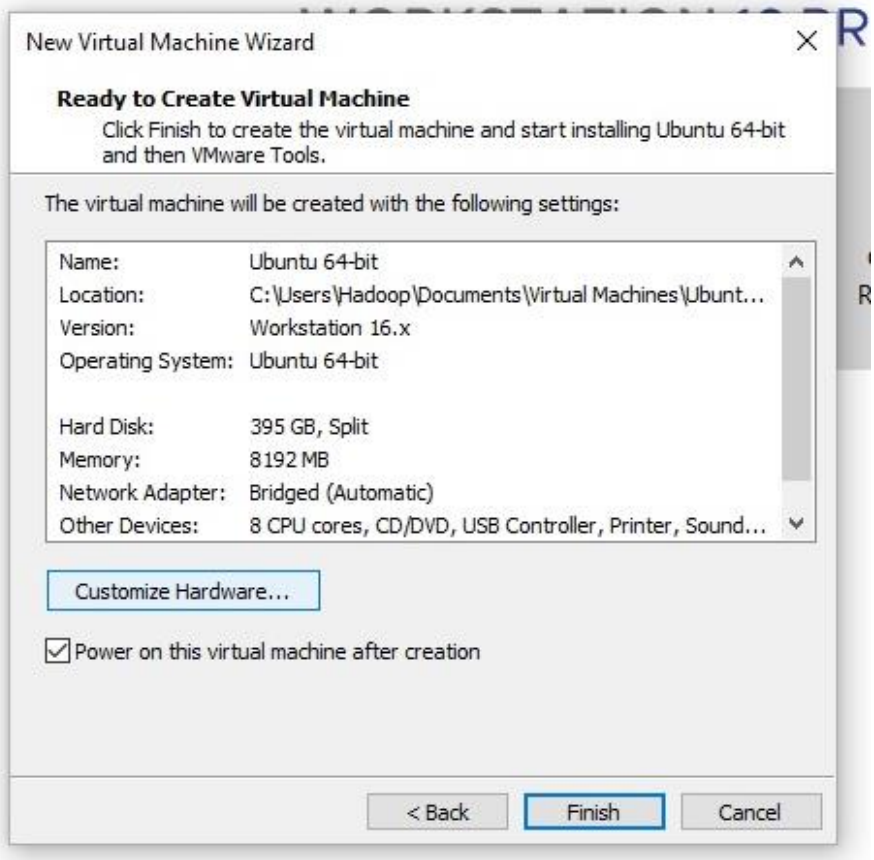
14. Give the Maximum disk size as **110 GB** and select the **Split virtual disk into multiple files** radio button.Click on **next** button.



15. The disk files will be automatically named based on the OS file name , click on Next button.

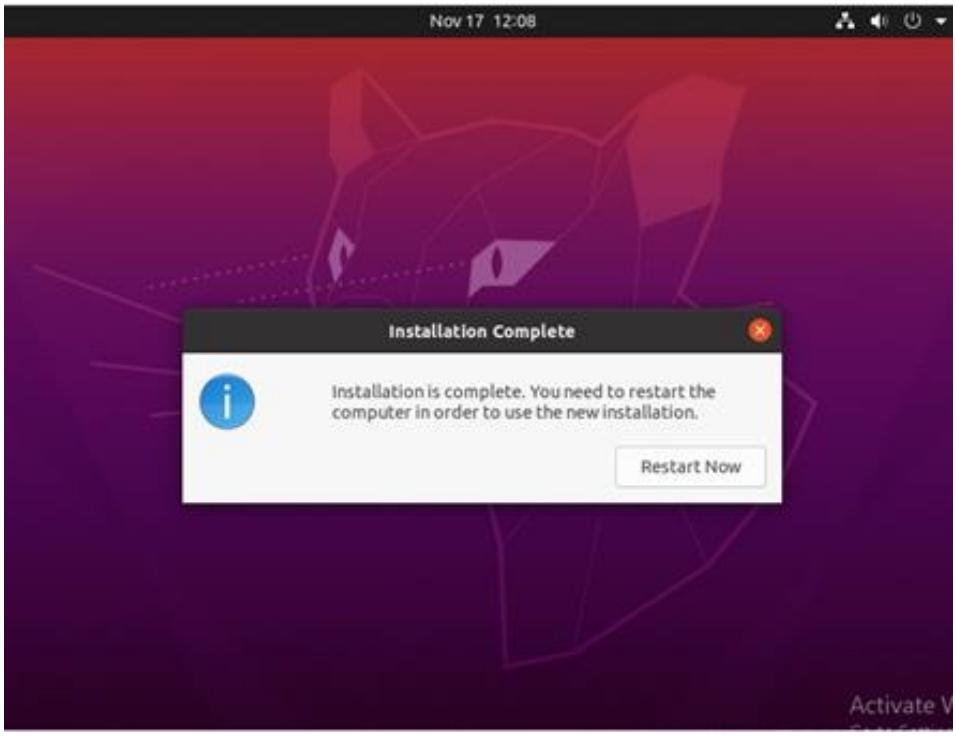


16. Click **Power on this virtual machine after creation** check box and **Finish** button.



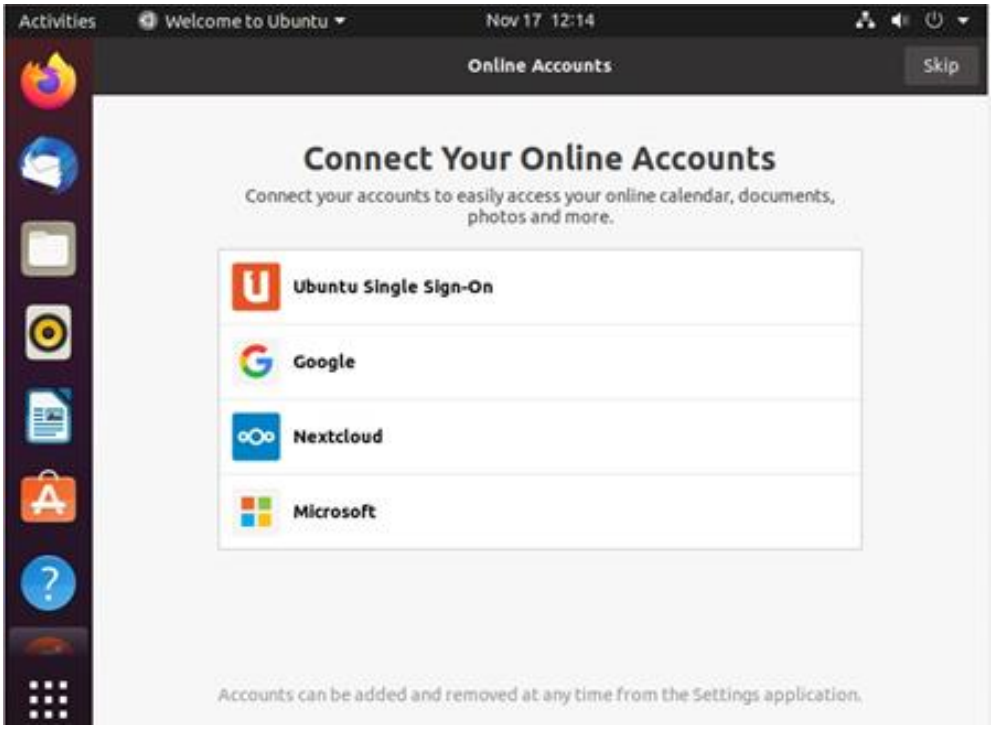
17. After which Ubuntu is being installed by taking the files from iso file and installing one by one.

18. Once installation is completed , click on **Restart Now** Button.

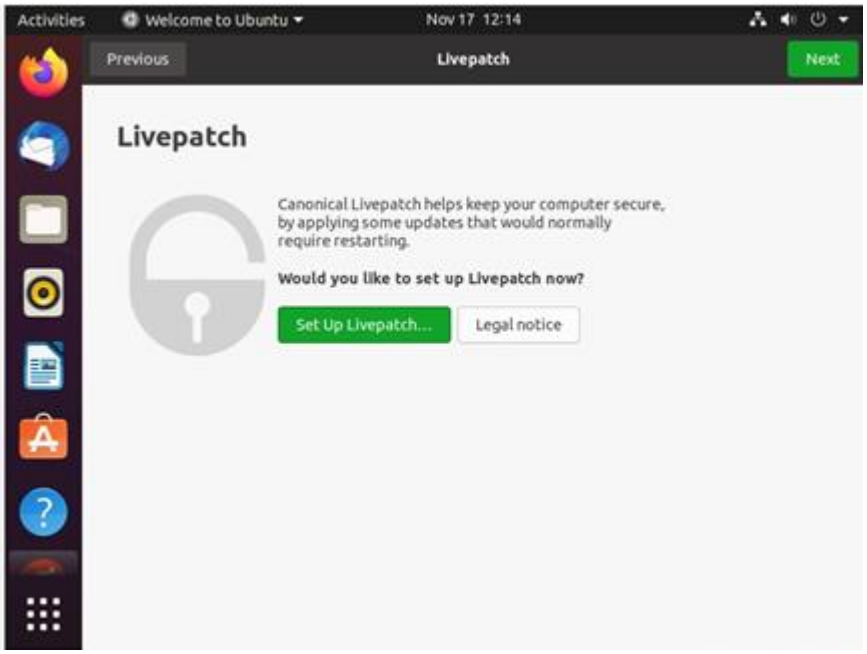


19. Press **Enter key** on keyboard , click on user and provide the **password** as **cse123** and login.

20. Click on **Skip** Button on top right corner.

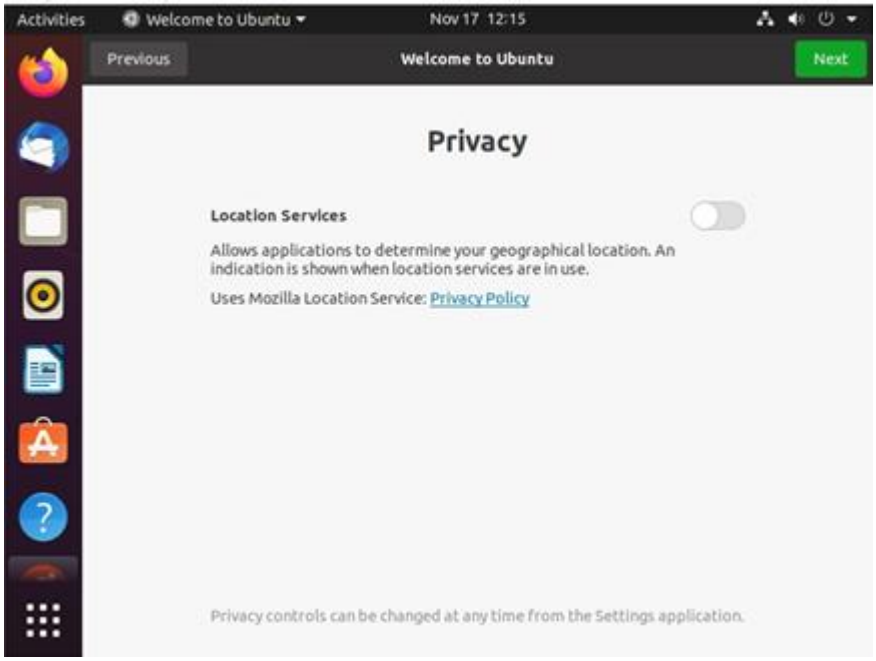


21. Click on **Next** Button on top right corner.

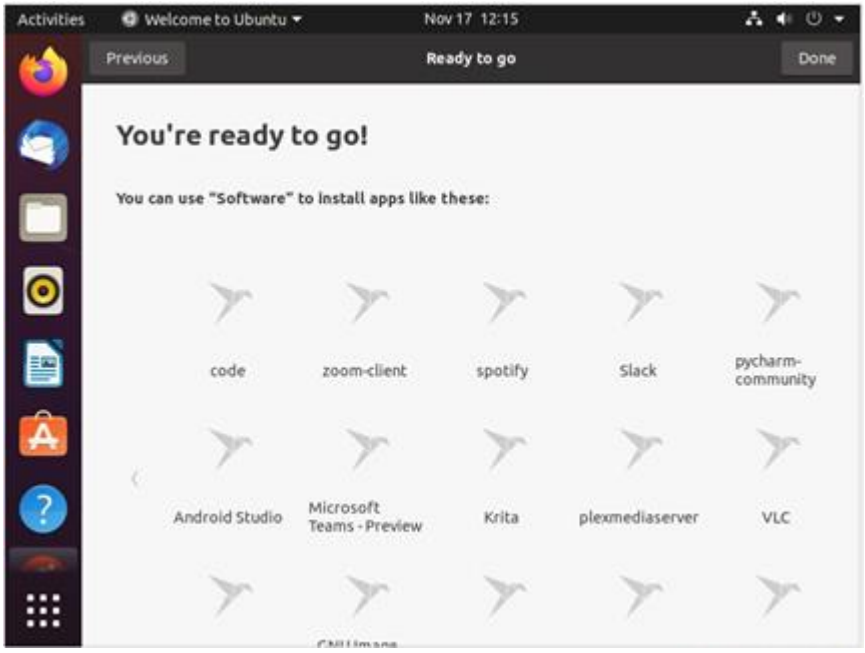


22. Click on **Next** Button on top right corner.

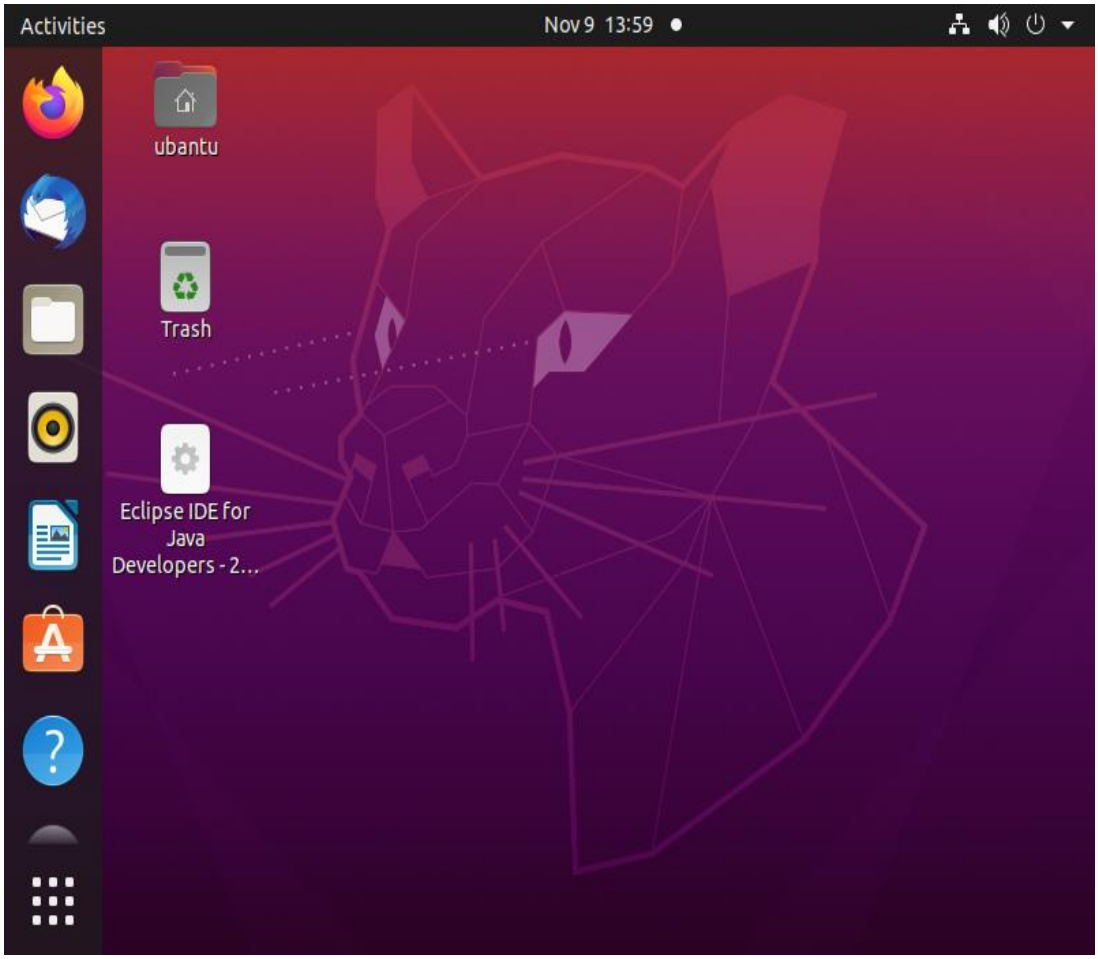
23. Click on **Next** Button on top right corner.



24. Click on **Done** Button on top right corner.



25. Ubuntu Desktop after installation has completed successfully.



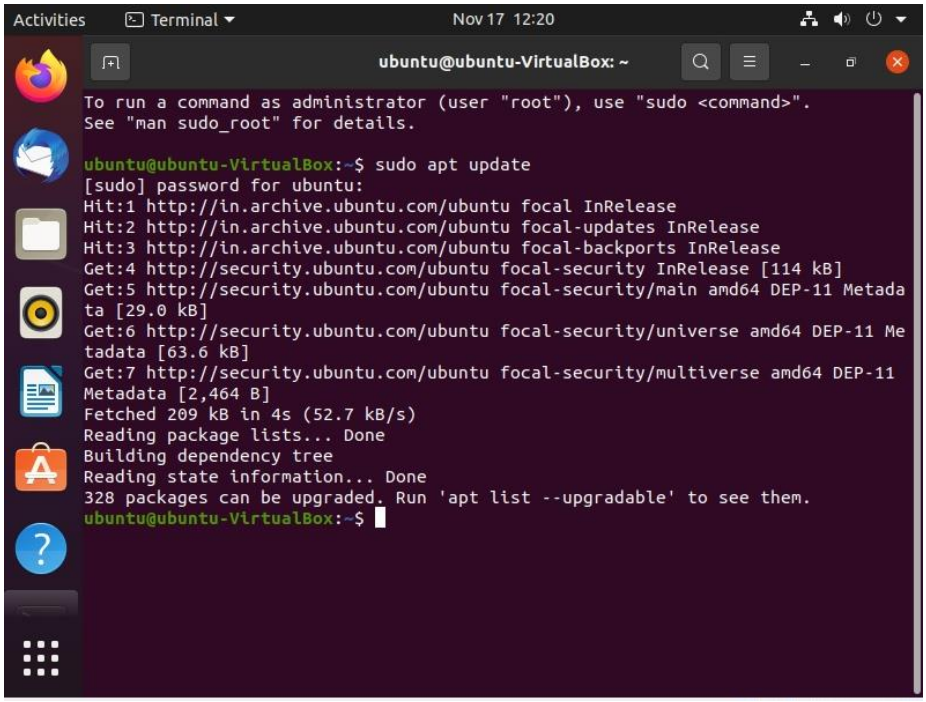
Install a C compiler in the virtual machine created using VMware and execute Simple Programs

PROCEDURE:

Step1: Open Terminal (Applications-Accessories-Terminal)

Step2: Update the guest system by typing this commands to update repositories and upgrade the emulated system

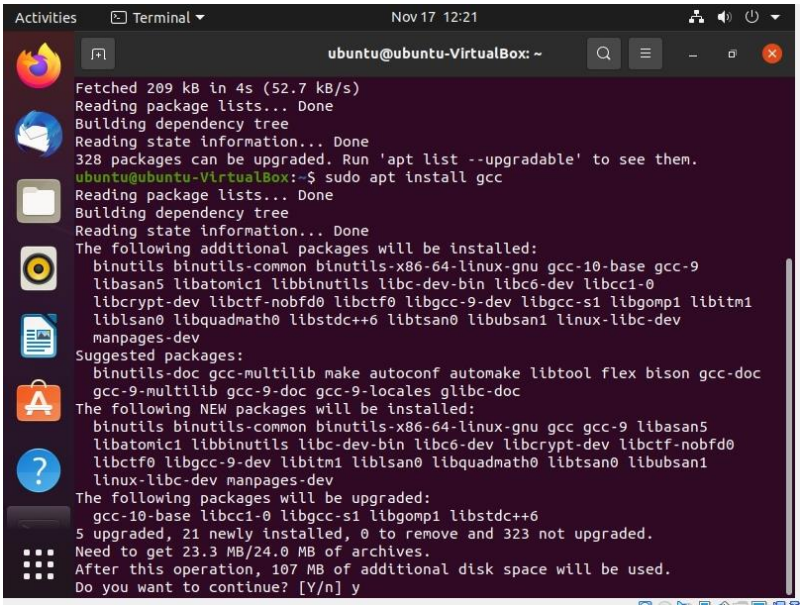
 # sudo apt update



 # sudo apt upgrade –y

Step3: Install c compiler.

 # sudo apt install gcc



Enter **Y** to denote yes to continue the installation and press **Enter Key**.

Step3: Open text file to type the program.

 # gedit hello.c

Step4: Type the code on the gedit.

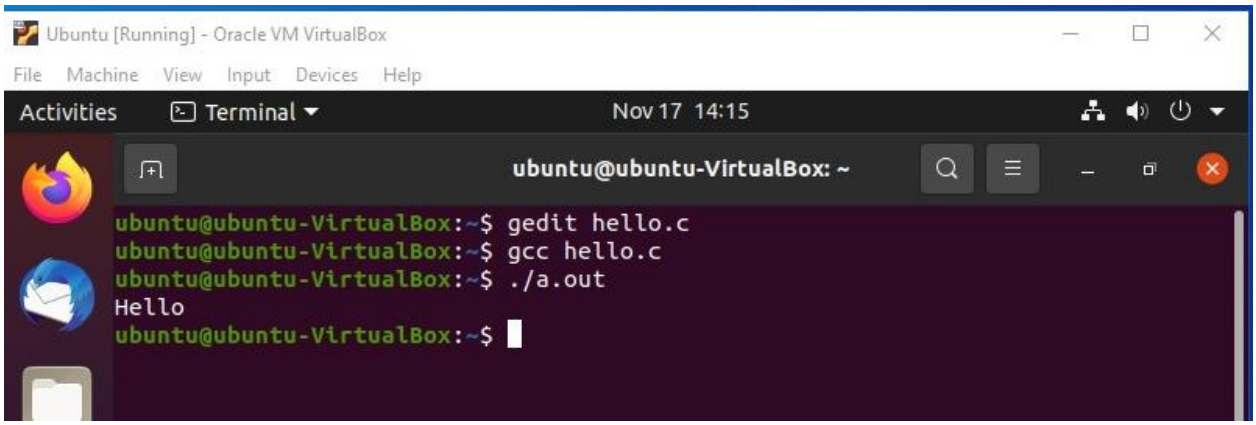
```
#include<stdio.h>
int main()
{
        Printf("Hello \n");
}
```

Step5: Save the file.
Step6: Compile the program.

 # gcc hello.c

Step7: Run the program.

 # ./a.out



RESULT:

Thus the installation of Vmware with linux-20.04 OS on top of windows7,8,10 and simple C Program has been executed successfully.

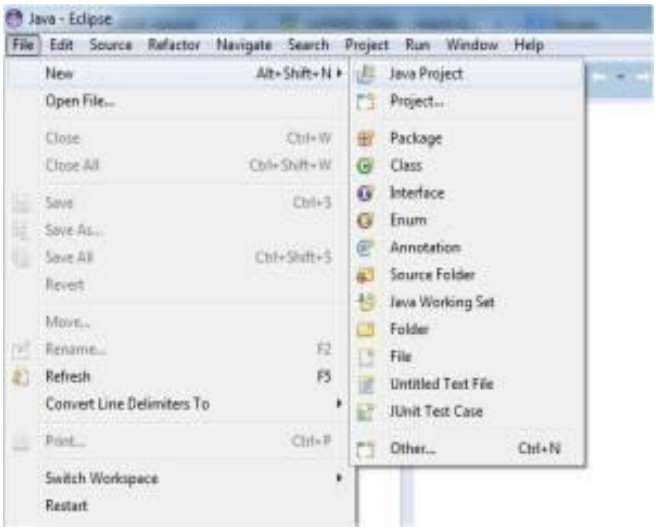
Ex.No.9 **Simulate a cloud scenario using CloudSim and run a scheduling algorithm that**
Date: **is not present in CloudSim.**

OBJECTIVE:

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

PROCEDURE:

Step1: Download **Cloudsim3.0.3 zip** file and extract it.
Step2: Download **Commons math 3.3.6.1 zip** file and extract it.
Step3: Copy **Commons-math-3.3.6.1 jar** file from **Common math3.3.6.1** folder and paste to **/Cloudsim3.0.3/ Cloudsim3.0.3/Jars/**
Step4: Open up Eclipse and Click on **File→Java Project**



Step2: .Enter project name. (I have named it as CloudIntro)

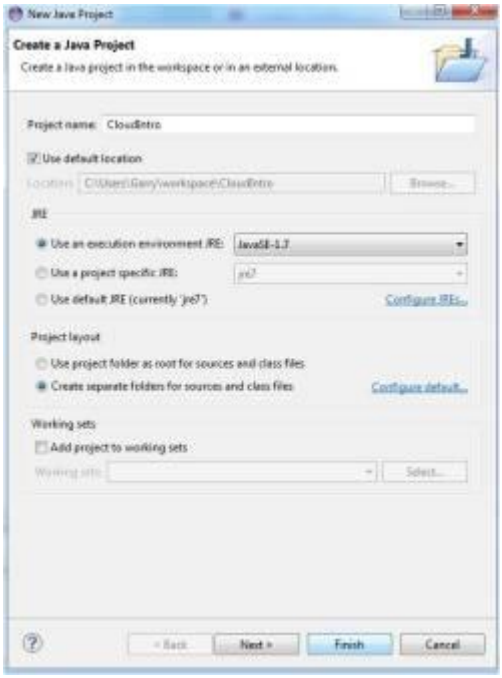
1.In the next line you will see the path where your project will be created. Uncheck **Use default location**

2. Click on Browse and set path as /Downloads/CloudSim-3.0.3/ CloudSim-3.0.3/

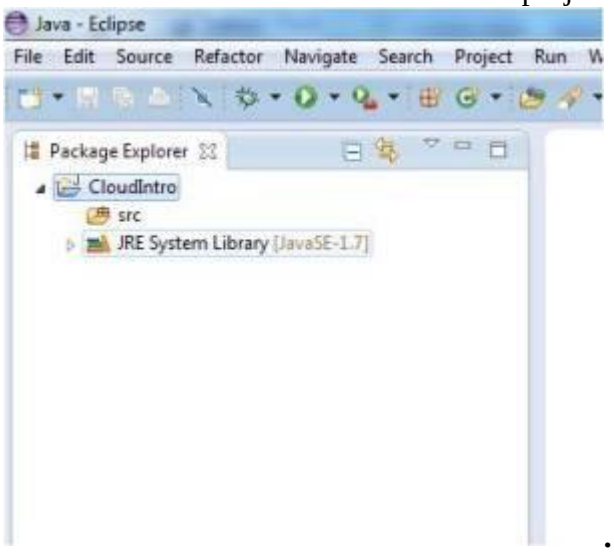
3.Next You need to select the JRE environment as **java1.8**.

4. Click on **Next** Button , all the jar files in CloudSim-3.0.3 will be displayed.

5.Finally Click **Finish**



6.An empty project named CloudIntro will be created in the project List.



7.Finally the cloud sim is installed into your Eclipse environment.

8.Open examples java file and try executing the example program by clicking on to RUN Button on the Menu bar.

9.Output will be displayed on the console window below the program.

Program:

First come first serve scheduling:

```
public void bindCloudletsToVmsSimple(){
 int cloudletNum=cloudletList.size();
 int vmNum=vmList.size();
 int idx=0;
 for(int i=0;i<cloudletNum;i++){
 cloudletList.get(i).setVmId(vmList.get(idx).getId());
 idx=(idx+1)%vmNum;
 }
 }
```

Our own Shortest First:

```
 public void bindCloudletToVmsScheduling(){
  int cloudletNum=cloudletList.size();
  int vmNum=vmList.size();
  double[] vmLoad=new double[vmNum];
  int idx=0;

  cloudletList.get(0).setVmId(vmList.get(0).getId());
  vmLoad[0]+=cloudletList.get(0).getCloudletLength()/vmList.get(0).getMips();
  for(int j=1;j<cloudletNum;j++){
   for(int i=0;i<vmNum;i++){
  if(vmLoad[i]<vmLoad[idx]){
  idx=i;
  }
}

  cloudletList.get(j).setVmId(vmList.get(idx).getId());
  vmLoad[idx]+=cloudletList.get(j).getCloudletLength()/vmList.get(idx).getMips();
  }
  }
```

Both methods are to be put in DatacenterBroker.java in the CloudSim package

In the example code, use broker.bindCloudletToVmsScheduling();

Output:

========== OUTPUT ==========

Cloudlet ID	STATUS	Data center ID	VM ID	Time	Start Time	Finish Time
0	SUCCESS	2	0	60	0.1	60.1
2	SUCCESS	2	0	70	60.1	130.1
1	SUCCESS	2	1	140	0.1	140.1

ExtendedExample finished!

RESULT:

Thus the cloud scenario using CloudSim to run a scheduling algorithm that is not present in CloudSim has been executed successfully.

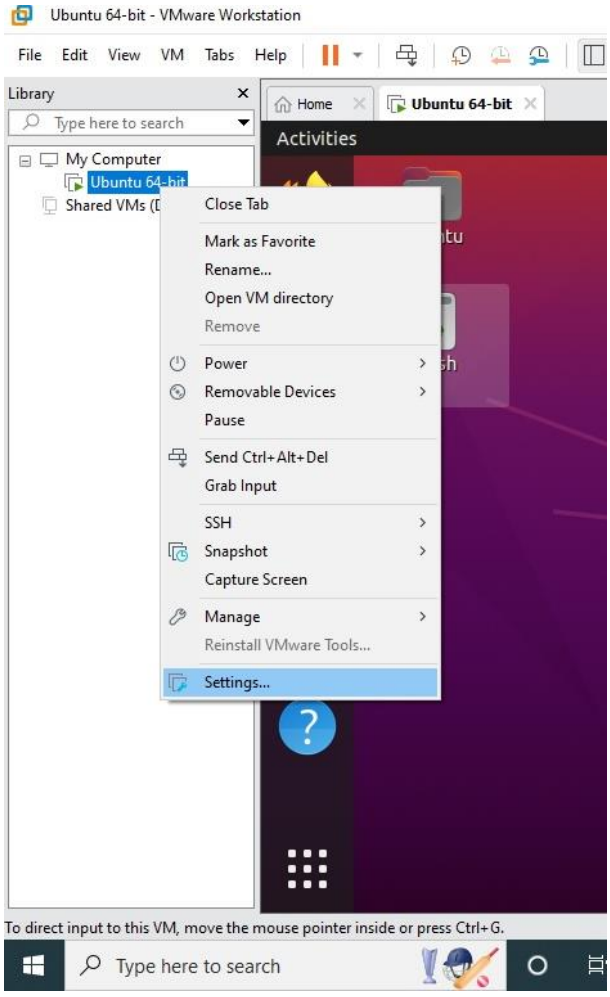
Ex.No.10 **Find a procedure to transfer the files from one virtual machine to another virtual machine.**
Date:

AIM:

To transfer the files from one virtual machine to another virtual machine,

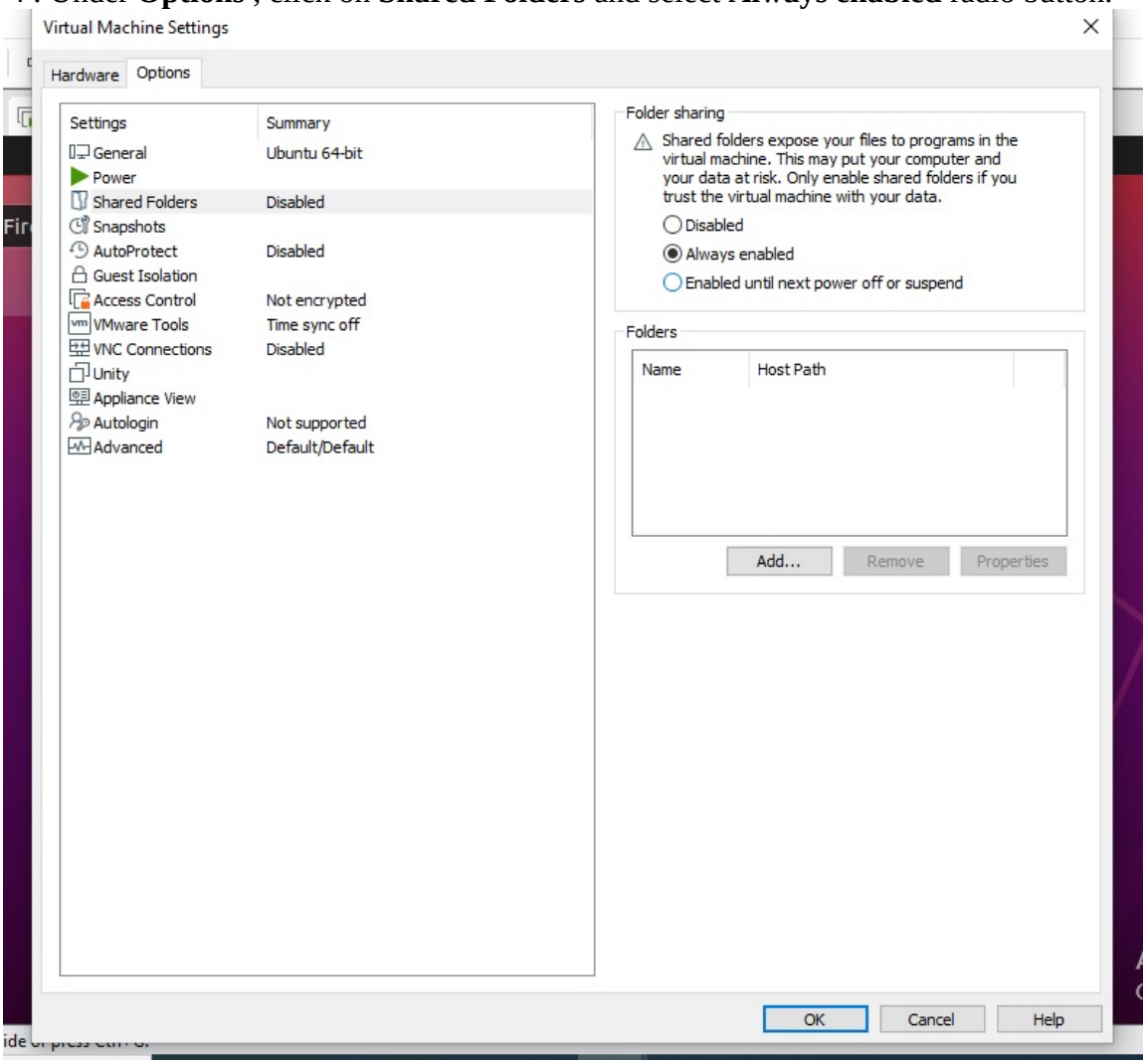
PROCEDURE:

Step 1: Open VMware and login to Ubuntu.
Step 2: Right click on Ubuntu 64-bit from the left side of VMware Workstation , and select Settings.



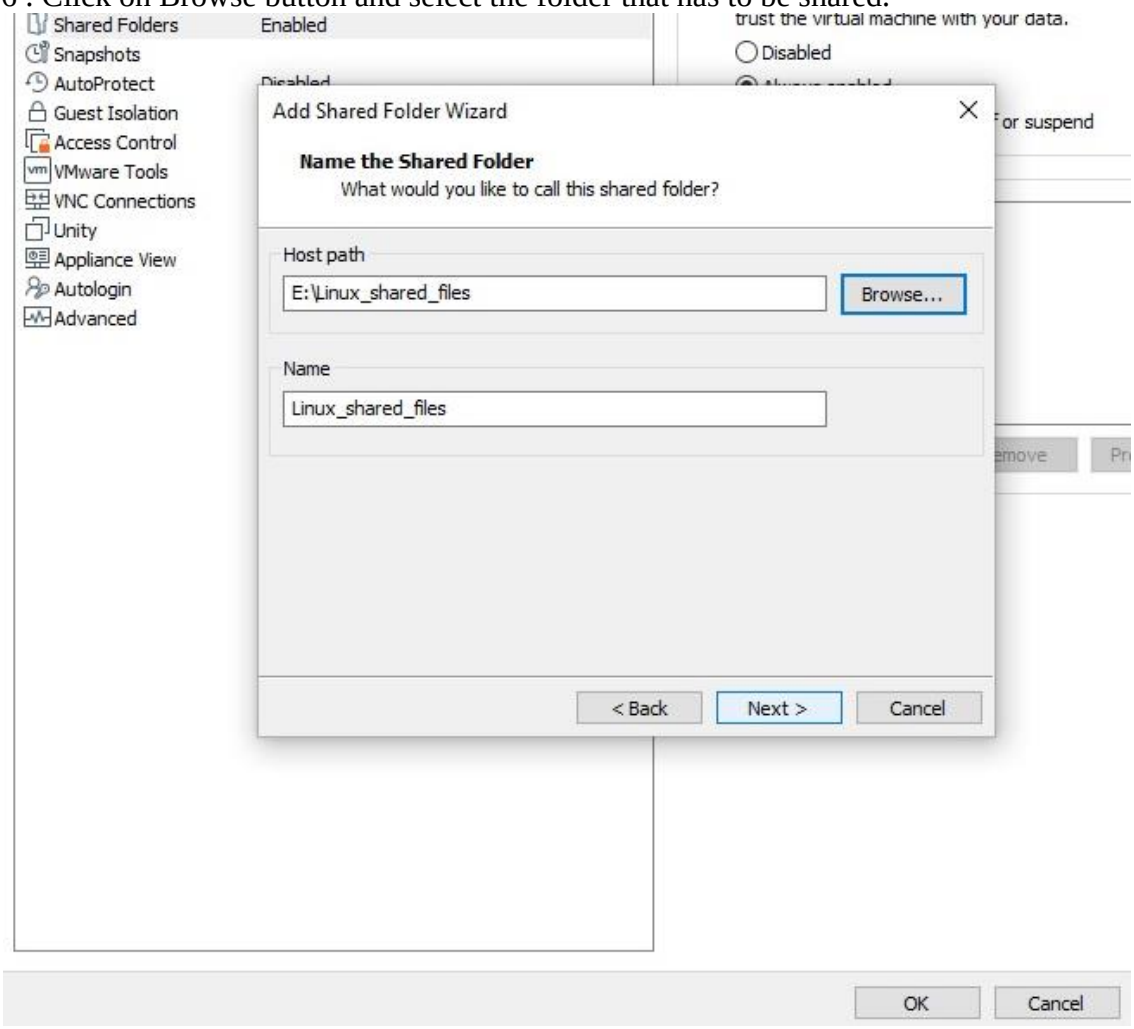
Step 3 : An **Virtual Machine settings** window will open , Click on the **Options** tab.

Step 4 : Under **Options** , click on **Shared Folders** and select **Always enabled** radio button.

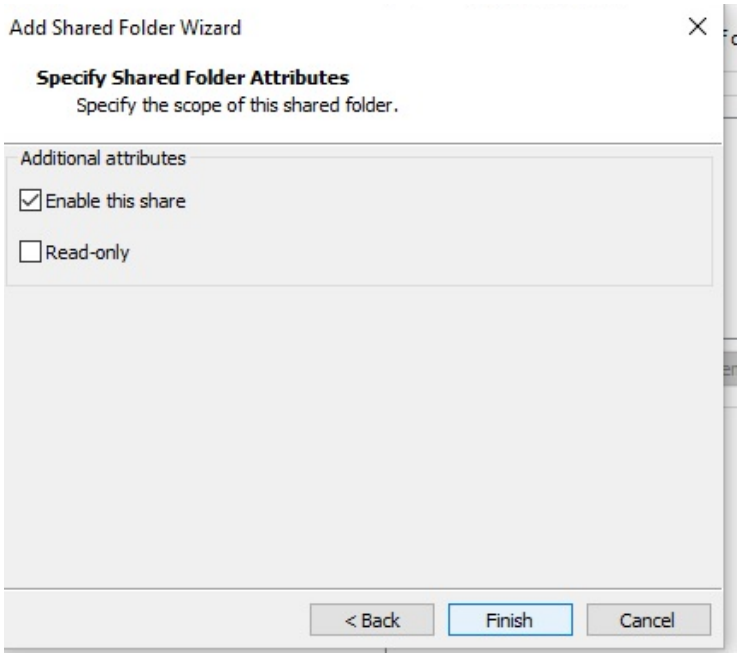


Step 5 : Click on **Add** button and **Next** button in a **Add shared Folder wizard**.

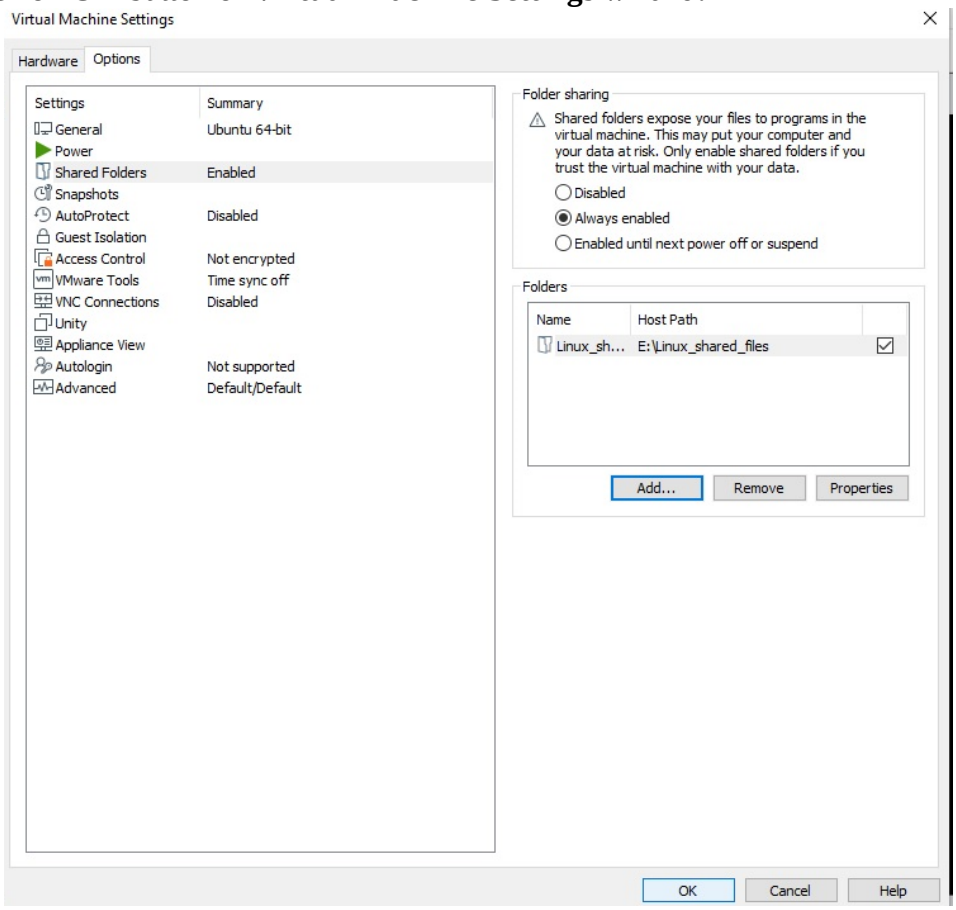
Step 6 : Click on Browse button and select the folder that has to be shared.



Step 7 : Click on **Next** button in the above **Add Shared Folder wizard**. In **Specify Shared Folder Attributes** click on **Enable this share** check box and click on **Finish** button.



Step 8 : Click on **OK** button of **Virtual Machine Settings wizard.**

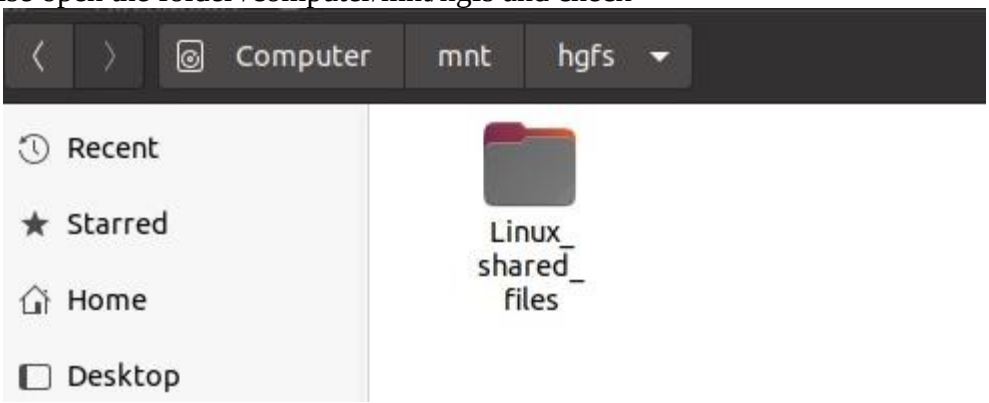


Step 9 : Next Login to Ubuntu user and open the terminal to check the shared files.

Step 10 : On typing the below command , the shared files name will be displayed.

ubuntu@ubuntu:~$ vmware-hgfsclient
Linux_shared_files

Step11 : Else open the folder /computer/mnt/hgfs and check



Step12 : Create **new folder** named **hadoop** to the **Linux_shared_files** folder which is being shared from windows10.

```
ubuntu@ubuntu:~$ cd /mnt/hgfs/Linux_shared_files/
ubuntu@ubuntu:/mnt/hgfs/Linux_shared_files$ mkdir hadoop
```

Step 13: To verify it type **ls** to display the folder contents.

```
ubuntu@ubuntu:/mnt/hgfs/Linux_shared_files$ ls
h1  hadoop  Hadoop_1
```

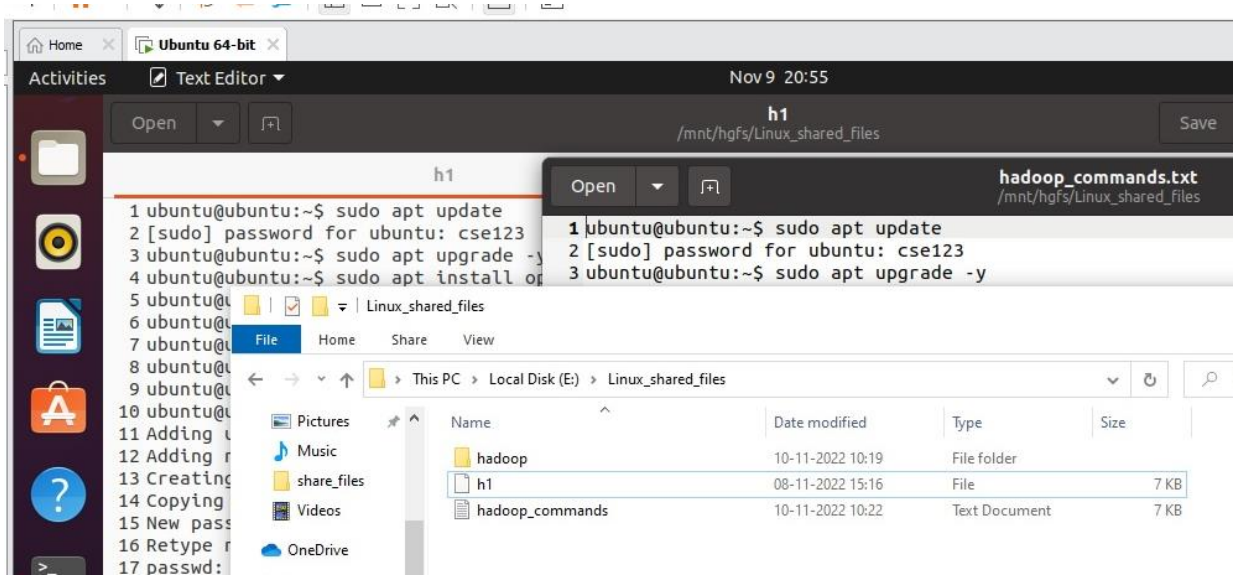
Step 14 : Now move the **Hadoop-1** file to the **hadoop folder** created under **Linux_shared_files** and verify it using **ls** command.

```
ubuntu@ubuntu:/mnt/hgfs/Linux_shared_files$ mv Hadoop_1
/mnt/hgfs/Linux_shared_files/hadoop/
ubuntu@ubuntu:/mnt/hgfs/Linux_shared_files$ ls
h1  hadoop
```

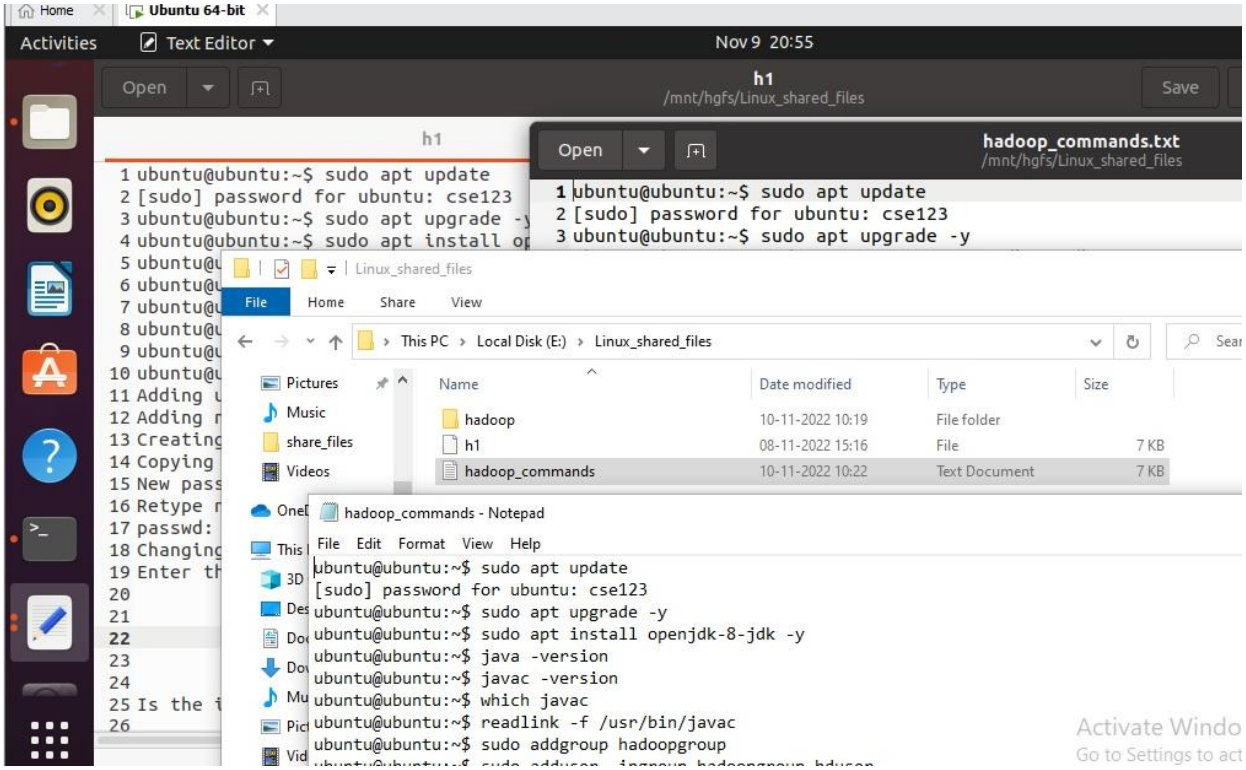
Step 15 : Now copy the contents of h1 file to new file named Hadoop_commands.

```
ubuntu@ubuntu:/mnt/hgfs/Linux_shared_files$ cp h1 hadoop_commands.txt
ubuntu@ubuntu:/mnt/hgfs/Linux_shared_files$ ls
h1  hadoop  hadoop_commands.txt
```

Can be verified in windows as well as shown in the image below,



The **file contents** of **h1** being **copied** to **hadoop_commands** can be verified in both **Ubuntu** and **windows** as shown below.



RESULT:

Thus the implementation of file transfer from one virtual machine to another virtual machine has been executed successfully.

Ex.No.11
Date:

Install Hadoop single node cluster

1. INSTALL , CONFIGURE , RUN HADOOP AND HDFS

OBJECTIVE:

To install, configure and run Hadoop and HDFS.

PROCEDURE:

1) Update the guest system

Open up a terminal and fire this commands to update repositories and upgrade the emulated system.

```
ubuntu@ubuntu:~$ sudo apt update

ubuntu@ubuntu:~$ sudo apt upgrade -y
```

2) Installing Java

Hadoop is a framework written in Java for running applications on large clusters of commodity hardware. Hadoop needs Java 8 or above to work.

```
ubuntu@ubuntu:~$ sudo apt install openjdk-8-jdk

You can verify Java version by typing:

ubuntu@ubuntu;~$ java -version
java version "1.8.0_60"
Java(TM) SE Runtime Environment (build 1.8.0_60-b27)
Java HotSpot(TM) 64-Bit Server VM (build 25.60-b23, mixed mode)
ubuntu@ubuntu;~$ javac –version
javac 1.8.0_342
```

3) To Set Environments

we need to create the JAVA_HOME environmental variable, to give hadoop the capability to find java executables.

```
ubuntu@ubuntu:~$ which javac
/usr/bin/javac
ubuntu@ubuntu:~$ readlink -f /usr/bin/javac
/usr/lib/jvm/java-8-openjdk-amd64/bin/javac
```

4) configure SSH Keys

Step 1 : We want to run our setup on a different general purpose user, so we will create a hduser user and a hadoopgroup group

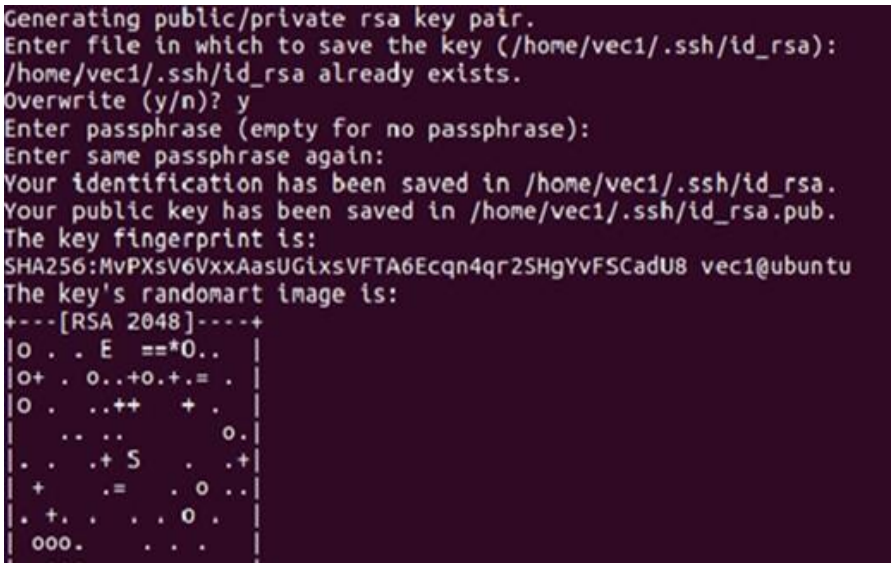
```
ubuntu@ubuntu:~$ sudo addgroup hadoopgroup
Adding group `hadoopgroup' (GID 1001) ...
Done.
ubuntu@ubuntu:~$ sudo adduser -ingroup hadoopgroup hduser
Adding user `hduser' ...
Adding new user `hduser' (1001) with group `hadoopgroup' ...
Creating home directory `/home/hduser' ...
Copying files from `/etc/skel' ...
New password: cse123
Retype new password: cse123
passwd: password updated successfully
Changing the user information for hduser
Enter the new value, or press ENTER for the default
       Full Name []:
       Room Number []:
       Work Phone []:
       Home Phone []:
       Other []:
Is the information correct? [Y/n] y
```

Step 2: We nees ssh access to our machine , so let's install and start an OpenSSh server

```
ubuntu@ubuntu:~$ sudo apt install openssh-server openssh-client –y
ubuntu@ubuntu:~$ su - hduser
Password: cse123
```

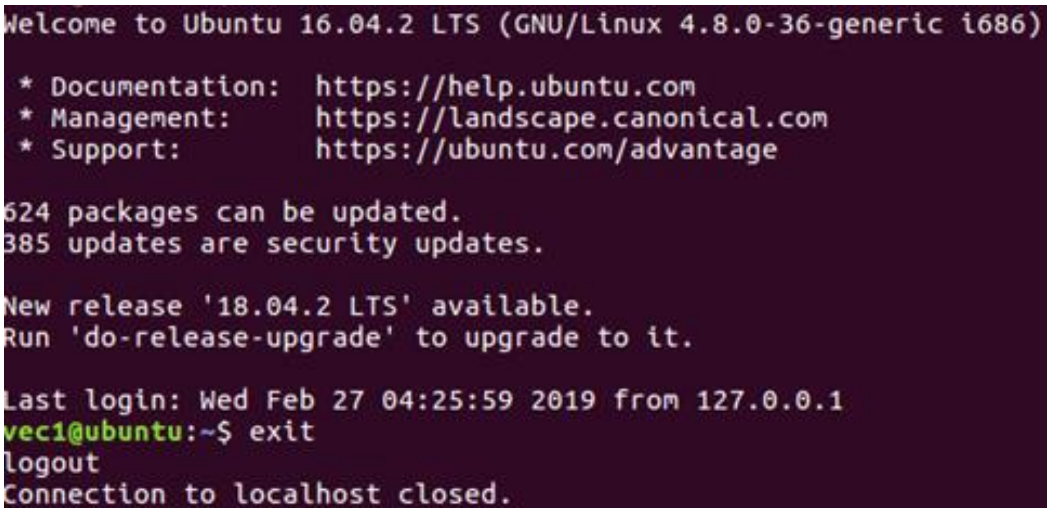
Now we need to setup passwordless ssh, by means of crypto keys. Then we create the key usi ng RSA encryption and finally authorize the key for the current user.

```
hduser@ubuntu:~$ ssh-keygen -t rsa -P '' -f ~/.ssh/id_rsa
```



hduser@ubuntu:~$ cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
hduser@ubuntu:~$ cd .ssh/
hduser@ubuntu:~/.ssh$ ls
authorized_keys id_rsa id_rsa.pub
hduser@ubuntu:~/.ssh$ chmod 0600 ~/.ssh/authorized_keys
hduser@ubuntu:~/.ssh$ ls -l
total 12
-rw------- 1 hduser hadoopgroup 567 Nov 8 01:03 authorized_keys
-rw------- 1 hduser hadoopgroup 2602 Nov 8 01:03 id_rsa
-rw-r--r-- 1 hduser hadoopgroup 567 Nov 8 01:03 id_rsa.pub
hduser@ubuntu:~/.ssh$ ssh-copy-id -i ~/.ssh/id_rsa.pub localhost
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/hduser/.ssh/id_rsa.pub"
The authenticity of host 'localhost (127.0.0.1)' can't be established.
ECDSA key fingerprint is SHA256:eM6XbH3tlMzAyEWtYhVnB4JntnoKwyyKmbhu8Z/yQeY.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed

/usr/bin/ssh-copy-id: WARNING: All keys were skipped because they already exist on the remote system.
(if you think this is a mistake, you may want to use -f option)
hduser@ubuntu:~$ ssh localhost



If no password were asked on ssh login, you successfully configured passwordless ssh.

3) Hadoop installation

We are ready to install Hadoop. Unfortunately, it does not come prepackaged, but we have to extract and move it to `/usr/local`.
Step 1: Download the tar.gz file of latest version Hadoop (hadoop-2.7.x) from the official site

hduser@ubuntu:~$wget https://dlcdn.apache.org/hadoop/common/hadoop-3.3.1/hadoop-3.3.1.tar.gz

--2022-11-08 01:11:10-- https://dlcdn.apache.org/hadoop/common/hadoop-3.3.1/hadoop-3.3.1.tar.gz
Resolving dlcdn.apache.org (dlcdn.apache.org)... 151.101.2.132, 2a04:4e42::644
Connecting to dlcdn.apache.org (dlcdn.apache.org)|151.101.2.132|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 605187279 (577M) [application/x-gzip]
Saving to: 'hadoop-3.3.1.tar.gz'

hadoop-3.3.1.tar.gz 100%[==================>] 577.15M 7.34MB/s in 2m 14s

2022-11-08 01:13:24 (4.31 MB/s) - 'hadoop-3.3.1.tar.gz' saved [605187279/605187279]

```
hduser@ubuntu:~$ ls
Desktop    Downloads         hadoop-3.3.1.tar.gz   Pictures
Templates
Documents  examples.desktop  Music                      Public
Videos
```

Step 2: Extract (untar) the downloaded file

hduser@ubuntu:~$ tar xzf hadoop-3.3.1.tar.gz

Step 3: Move the file to /usr/local , since hduser do not have admin rights go to admin user.

hduser@ubuntu:~$ su - ubuntu
Password:
ubuntu@ubuntu:~$ sudo mv /home/hduser/hadoop-3.3.1 /usr/local/

Step 4: Create a symbolic Link

ubuntu@ubuntu:~$ sudo ln -sf /usr/local/hadoop-3.3.1/ /usr/local/hadoop

Step 5: Change the file owner as hduser

ubuntu@ubuntu:~$ sudo chown -R hduser:hadoopgroup /usr/local/hadoop

 # sudo chown –R hduser:hadoopgroup /usr/local/hadoop-2.7.7/

Now we need to configure some environmental variables, with the `hadoopuser` account. Switch to that account and edit `~/.bashrc`:

Step 6: login to hduser using the password
ubuntu@ubuntu:~$ su - hduser
Password: cse123

Step 7: Open .bashrc file

```
hduser@ubuntu:~$ nano .bashrc
```

Append at the end:

```
# Hadoop config
export HADOOP_PREFIX=/usr/local/hadoop
export HADOOP_HOME=/usr/local/hadoop
export HADOOP_MAPRED_HOME=${HADOOP_HOME}
export HADOOP_COMMON_HOME=${HADOOP_HOME}
export HADOOP_HDFS_HOME=${HADOOP_HOME}
export YARN_HOME=${HADOOP_HOME}
export HADOOP_CONF_DIR=${HADOOP_HOME}/etc/hadoop
# Native path
export HADOOP_COMMON_LIB_NATIVE_DIR=${HADOOP_PREFIX}/lib/native
export HADOOP_OPTS="-Djava.library.path=$HADOOP_PREFIX/lib/native"
# Java path
export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
# OS path
export PATH=$PATH:$HADOOP_HOME/bin:$JAVA_PATH/bin:$HADOOP_HOME/sbin
```

Next, source `~/.bashrc` to apply changes.

```
hduser@ubuntu:~$ source ~/.bashrc
```

Now we need to edit `/usr/local/hadoop/etc/hadoop/hadoop-env.sh`:

```
hduser@ubuntu:~$ source ~/.bashrc
hduser@ubuntu:~$ cd /usr/local/hadoop/etc/hadoop/
hduser@ubuntu:/usr/local/hadoop/etc/hadoop$ ls
capacity-scheduler.xml            kms-log4j.properties
configuration.xsl                 kms-site.xml
container-executor.cfg            log4j.properties
core-site.xml                     mapred-env.cmd
hadoop-env.cmd                    mapred-env.sh
hadoop-env.sh                     mapred-queues.xml.template
hadoop-metrics2.properties        mapred-site.xml
hadoop-policy.xml                 shellprofile.d
hadoop-user-functions.sh.example  ssl-client.xml.example
hdfs-rbf-site.xml                 ssl-server.xml.example
hdfs-site.xml                     user_ec_policies.xml.template
httpfs-env.sh                     workers
httpfs-log4j.properties           yarn-env.cmd
httpfs-site.xml                   yarn-env.sh
kms-acls.xml                      yarnservice-log4j.properties
kms-env.sh                        yarn-site.xml
```

hduser@ubuntu:/usr/local/hadoop/etc/hadoop$ nano hadoop-env.sh

And add this at the end:

export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64/

4) Configure Hadoop

Hadoop configuration is quite hard, because it has a lot of config files. We need to navigate to /usr/local/hadoop/etc/hadoop and edit these files:

- core-site.xml
- hdfs-site.xml
- mapred-site.xml
- yarn-site.xml

They all are XML files with a top-level **<configuration>** node. For clarity we report the configuration node only.

core-site.xml

hduser@ubuntu:/usr/local/hadoop/etc/hadoop$ nano core-site.xml

#Add below lines in this file(between "<configuration>" and "<"/configuration>")

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

hdfs-site.xml

hduser@ubuntu:/usr/local/hadoop/etc/hadoop$ nano hdfs-site.xml

#Add below lines in this file(between "<configuration>" and "<"/configuration>")

```
<configuration>
<property>
 <name>dfs.replication</name>
 <value>1</value>
</property>
<property>
  <name>dfs.name.dir</name>
    <value>file:/usr/local/hadoop/hadoopdata/hdfs/namenode</value>
</property>
<property>
  <name>dfs.data.dir</name>
    <value>file:/usr/local/hadoop/hadoopdata/hdfs/datanode</value>
</property>
</configuration>
```

mapred-site.xml

```
hduser@ubuntu:/usr/local/hadoop/etc/hadoop$ nano mapred-site.xml
```

#Add below lines in this file(between "<configuration>" and "<"/configuration>")

```
<configuration>
<property>
  <name>mapreduce.framework.name</name>
   <value>yarn</value>
 </property>
</configuration>
```

yarn-site.xml

```
hduser@ubuntu:/usr/local/hadoop/etc/hadoop$ nano yarn-site.xml
```

#Add below lines in this file(between "<configuration>" and "<"/configuration>")

```
<configuration>
<property>
  <name>yarn.nodemanager.aux-services</name>
  <value>mapreduce_shuffle</value>
</property>
<property>
  <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
  <value>org.apache.hadoop.mapred.ShuffleHandler</value>
</property>
<property>
  <name>yarn.resourcemanager.hostname</name>
  <value>127.0.0.1</value>
</property>
<property>
  <name>yarn.acl.enable</name>
  <value>0</value>
</property>
<property>
  <name>yarn.nodemanager.env-whitelist</name>

<value>JAVA_HOME,HADOOP_COMMON_HOME,HADOOP_HDFS_HOME,HADOOP_CONF_DIR,CLASSPATH_PERPEND_DISTCACHE,HADOOP_YARN_HOME,HADOOP_MAPRED_HOME</value>
</property>
</configuration>
```

Create a datanode and namenode directory as mentioned below.

```
hduser@ubuntu:~$ cd /usr/local/hadoop
hduser@ubuntu:/usr/local/hadoop$ mkdir hadoopdata
hduser@ubuntu:/usr/local/hadoop$ cd hadoopdata/
hduser@ubuntu:/usr/local/hadoop/hadoopdata$ mkdir hdfs
hduser@ubuntu:/usr/local/hadoop/hadoopdata$ cd hdfs/
hduser@ubuntu:/usr/local/hadoop/hadoopdata/hdfs$ mkdir namenode
hduser@ubuntu:/usr/local/hadoop/hadoopdata/hdfs$ mkdir datanode
```

5) Format namenode :

Next, we need to format the namenode filesystem with the following command:

hduser@ubuntu:/usr/local/hadoop$ cd sbin/
hduser@ubuntu:/usr/local/hadoop/sbin$ hdfs namenode -format
WARNING: HADOOP_PREFIX has been replaced by HADOOP_HOME. Using value of HADOOP_PREFIX.
WARNING: /usr/local/hadoop/logs does not exist. Creating.
2022-11-08 01:42:03,594 INFO namenode.NameNode: STARTUP_MSG:
/**
STARTUP_MSG: Starting NameNode
STARTUP_MSG: host = ubuntu/127.0.1.1
STARTUP_MSG: args = [-format]
STARTUP_MSG: version = 3.3.1
duser@ubuntu:~$ cd /usr/local/Hadoop
…….
…….
…….
…….
2022-11-08 01:42:04,967 INFO namenode.FSImage: Allocated new BlockPoolId: BP-295377839-127.0.1.1-1667900524961
2022-11-08 01:42:04,986 INFO common.Storage: Storage directory /usr/local/hadoop/hadoopdata/hdfs/namenode has been successfully formatted.
…….
…….
…….
2022-11-08 01:42:05,172 INFO namenode.FSImage: FSImageSaver clean checkpoint: txid=0 when meet shutdown.
2022-11-08 01:42:05,172 INFO namenode.NameNode: SHUTDOWN_MSG:
/**
SHUTDOWN_MSG: Shutting down NameNode at ubuntu/127.0.1.1
**/

Search the output: if you can read a string like this:

INFO common.Storage: Storage directory /usr/local/hadoop/hadoopdata/hdfs/namenode has been successfully formatted.

It's done.

6) Start and Stop Services :
Now , the last thing to do is starting Hadoop services:

```
hduser@ubuntu:~$ start-dfs.sh
WARNING: HADOOP_PREFIX has been replaced by HADOOP_HOME. Using value of
HADOOP_PREFIX.
Starting namenodes on [localhost]
WARNING: HADOOP_PREFIX has been replaced by HADOOP_HOME. Using value of
HADOOP_PREFIX.
Starting datanodes
WARNING: HADOOP_PREFIX has been replaced by HADOOP_HOME. Using value of
HADOOP_PREFIX.
Starting secondary namenodes [ubuntu]
WARNING: HADOOP_PREFIX has been replaced by HADOOP_HOME. Using value of
HADOOP_PREFIX.
ubuntu: Warning: Permanently added 'ubuntu' (ECDSA) to the list of
known hosts.
WARNING: HADOOP_PREFIX has been replaced by HADOOP_HOME. Using value of
HADOOP_PREFIX.
hduser@ubuntu:~$ jps
57824 SecondaryNameNode
57464 NameNode
57610 DataNode
57982 Jps
```

To check the Hadoop demons status use the command **jps.**
Jps sands for java virtual Machine Process Status tool.

```
hduser@ubuntu:~$ jps
57824 SecondaryNameNode
57464 NameNode
57610 DataNode
57982 Jps
```

```
hduser@ubuntu:~$ start-yarn.sh
WARNING: HADOOP_PREFIX has been replaced by HADOOP_HOME. Using value of
HADOOP_PREFIX.
Starting resourcemanager
WARNING: HADOOP_PREFIX has been replaced by HADOOP_HOME. Using value of
HADOOP_PREFIX.
Starting nodemanagers
WARNING: HADOOP_PREFIX has been replaced by HADOOP_HOME. Using value of
HADOOP_PREFIX.
```

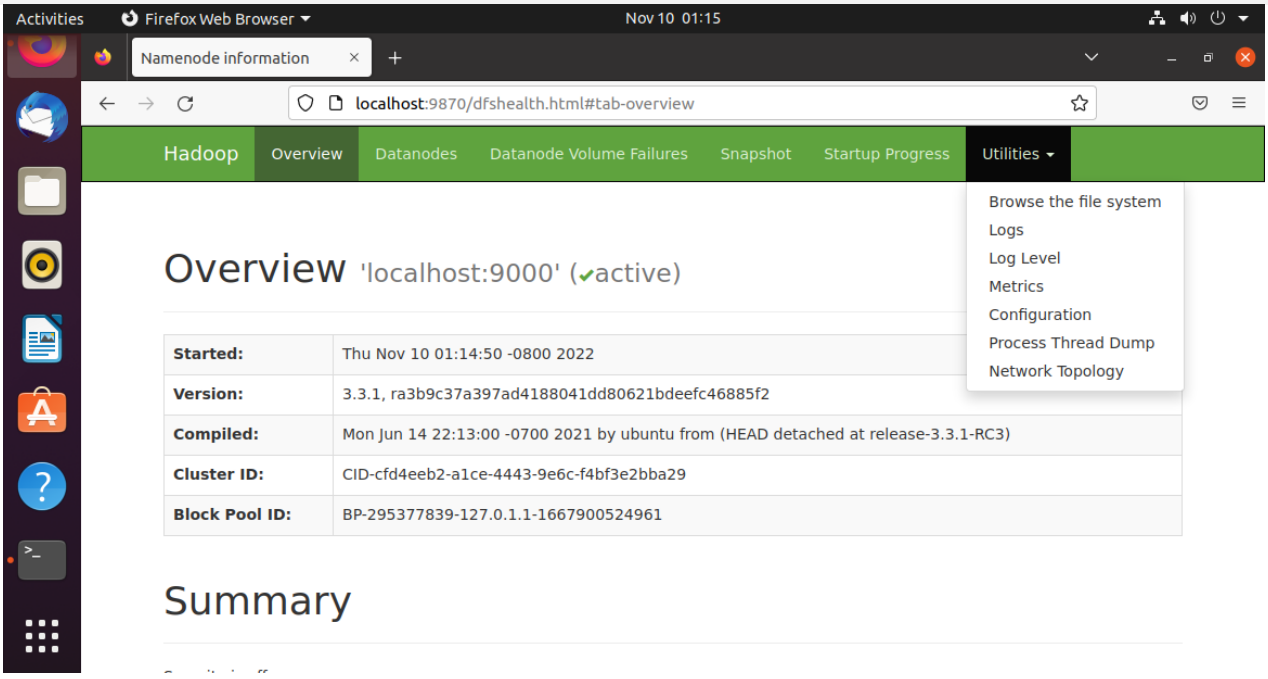
To check the status of the services use the jps command
jps sands for java virtual Machine Process Status tool.

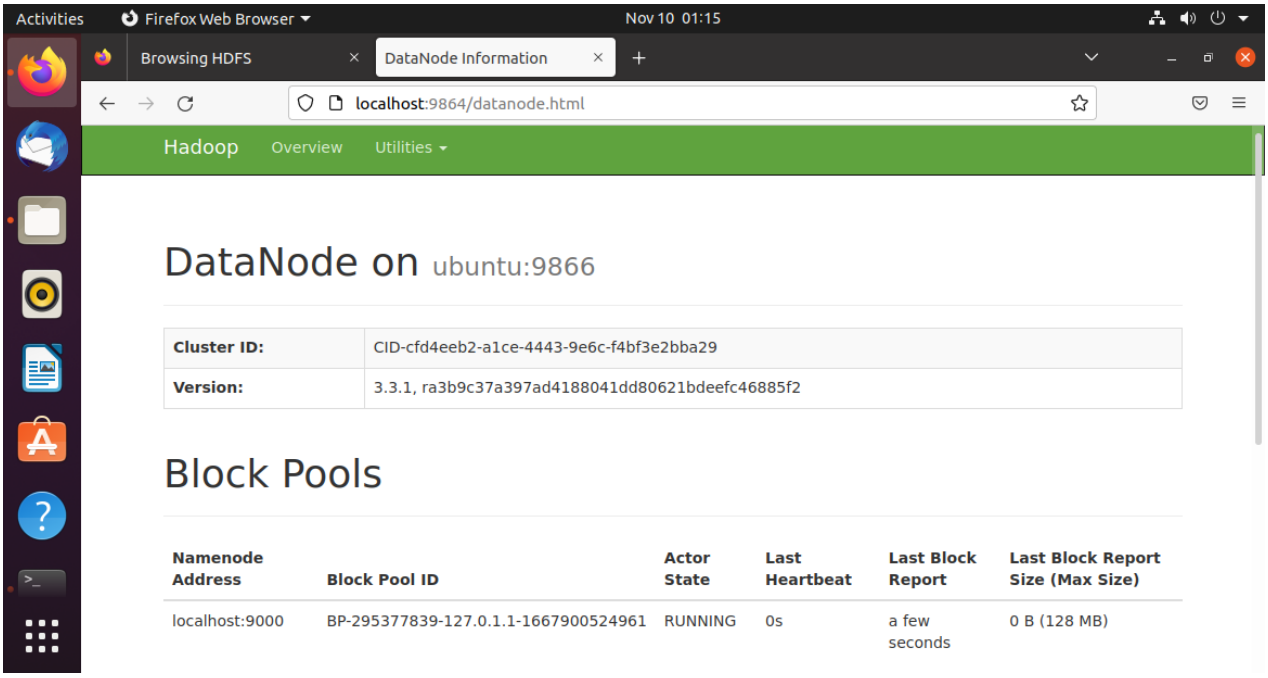
```
hduser@ubuntu:~$ jps
57824 SecondaryNameNode
58082 ResourceManager
58374 Jps
57464 NameNode
57610 DataNode
58221 NodeManager
(OR)
hduser@ubuntu:~$ start-all.sh
```

Verify Hadoop Installation:

```
# hadoop version
```

To view the files in HDFS browse the URL **localhost:9870 and localhost:9864**





To stop services, these are the commands:

```
hduser@ubuntu:~$ stop-dfs.sh
hduser@ubuntu:~$ stop-yarn.sh
(or)
hduser@ubuntu:~$ stop-all.sh
```

RESULT:

Thus the installation and configuration of Hadoop and HDFS is successfully executed.

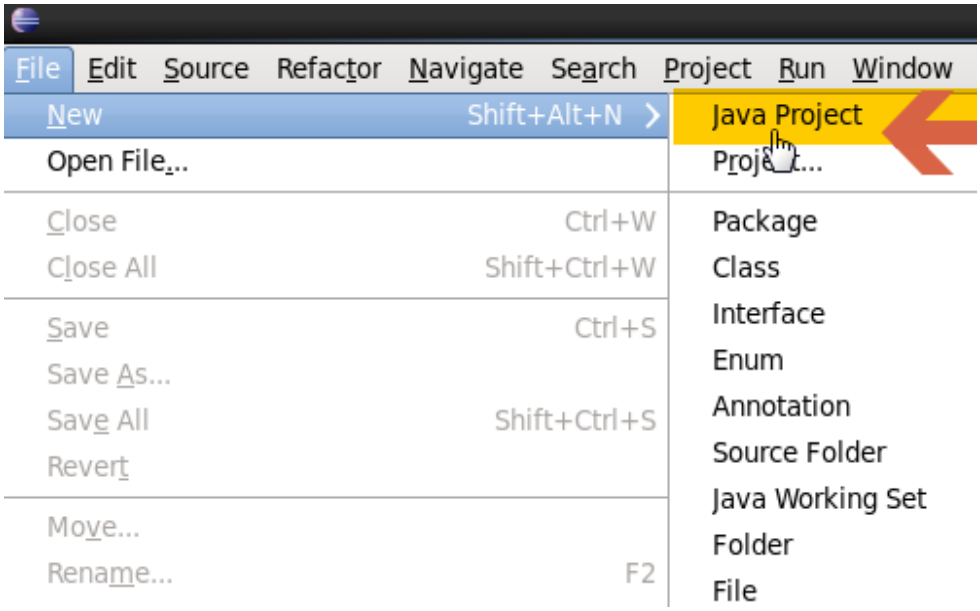
Ex No : 12 **IMPLEMENT WORD COUNT USING MAPREDUCE**
DATE :

OBJECTIVE:
Word count program to demonstrate the use of Map and Reduce tasks

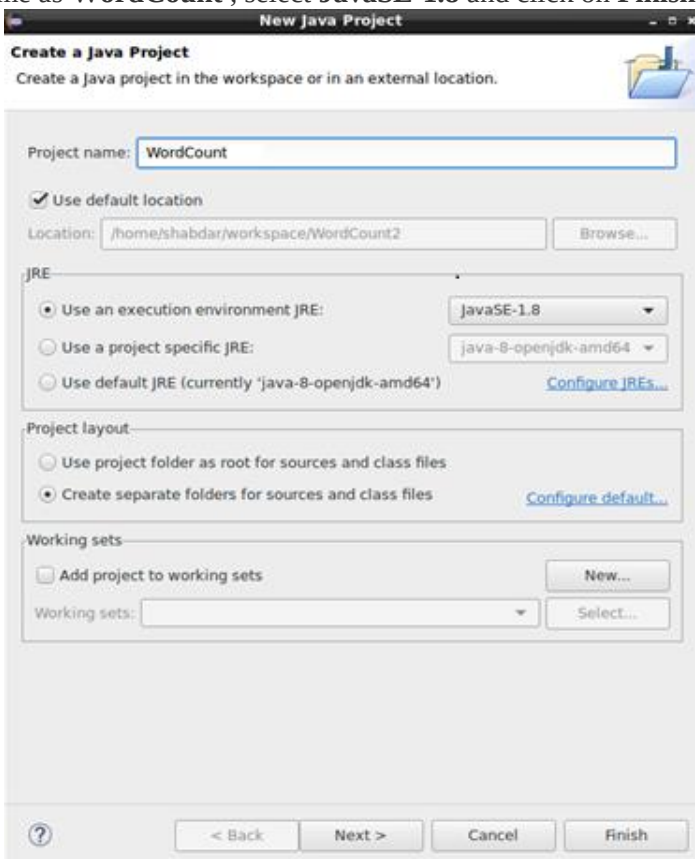
PROCEDURE:
STEPS:
1. Analyze the input file content
2. Develop the code
    a. Writing a map function
    b. Writing a reduce function
    c. Writing the Driver class
3. Compiling the source
4. Building the JAR file
5. Starting the DFS
6. Creating Input path in HDFS and moving the data into Input path
7. Executing the program

Steps

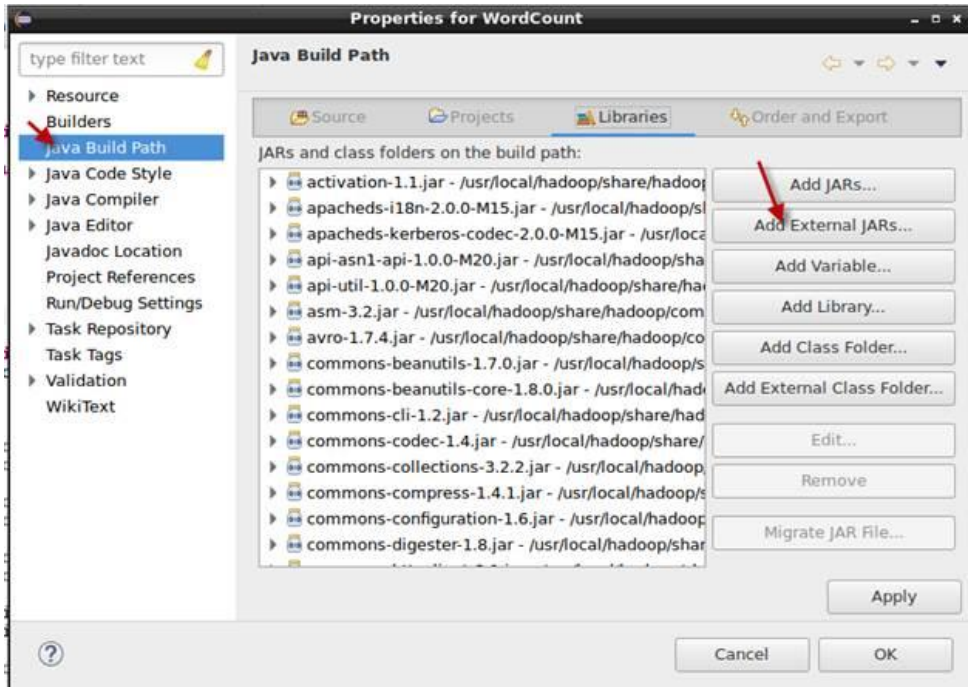
1. Open Eclipse and create new Java Project
2. File -> New -> Java Project



3. Enter the project name as **WordCount** , select **JavaSE-1.8** and click on **Finish** Button.



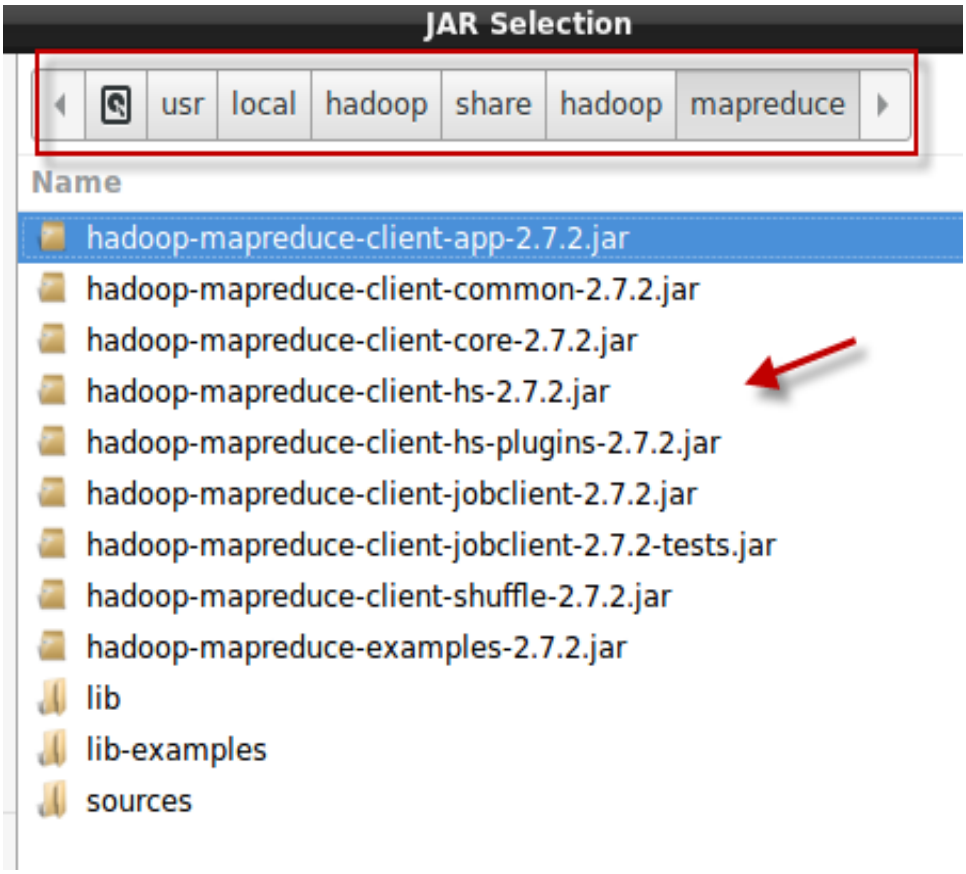
4.Right click on **WordCount** Project under **Package Explorer** and go to **properties** in Eclipse IDE.



5.Under **Libraries** click **Add External JARs**

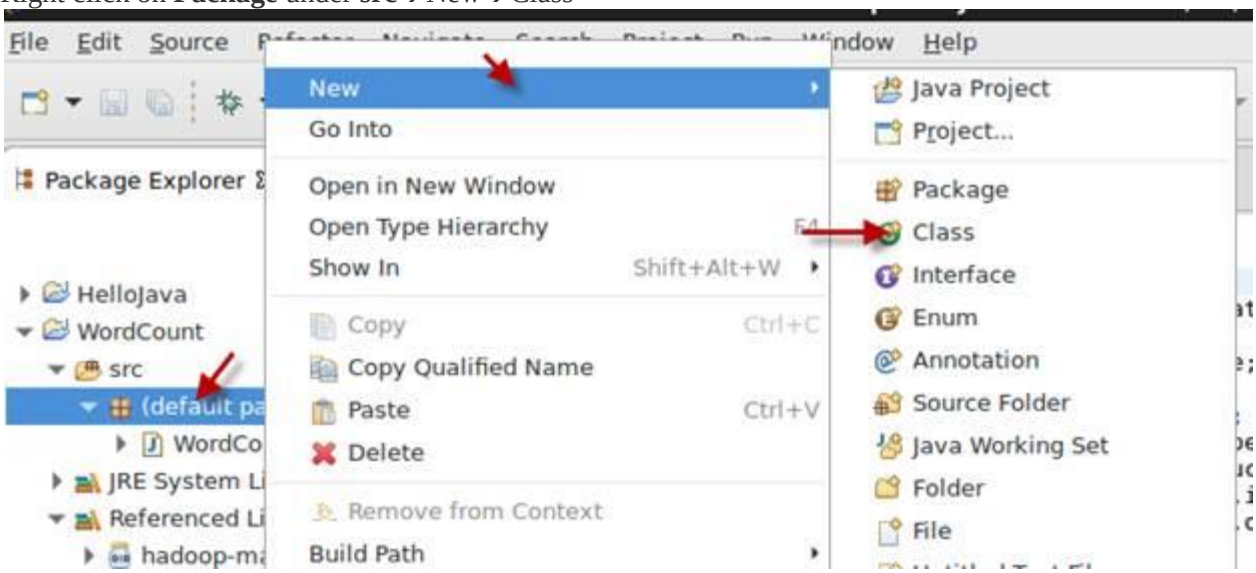
From the hadoop installed directory **/usr/local/hadoop** add the following jar files.

1. **hadoop-common-2.7.7.jar** from the path **/usr/local/hadoop/share/hadoop/common**
2. **common-cli-1.2.jar** from the path **/usr/local/hadoop/share/hadoop/common/lib**
3. **hadoop-mapreduce-client-core-2.7.7.jar** from the path **/usr/local/hadoop/share/hadoop/mapreduce**



1. Right click on **src →New→Package**
2. Enter the package name as **WordCount** and click on **Finish** Button.

3. Right click on **Package** under **src**→New→Class



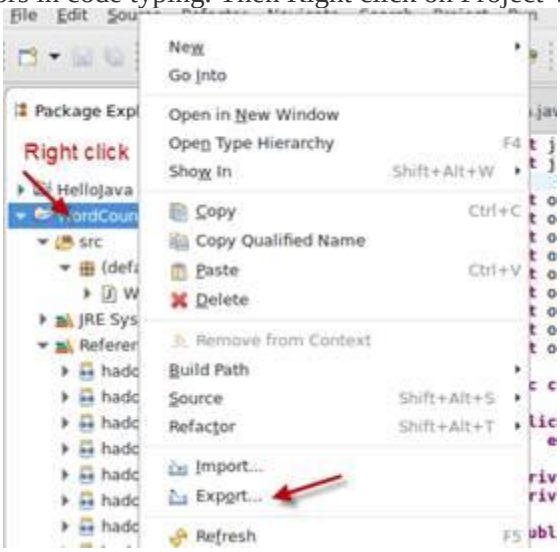
4. Add new Class name as **WordCount.java.**
5. Type in the java code.

Sample Program:

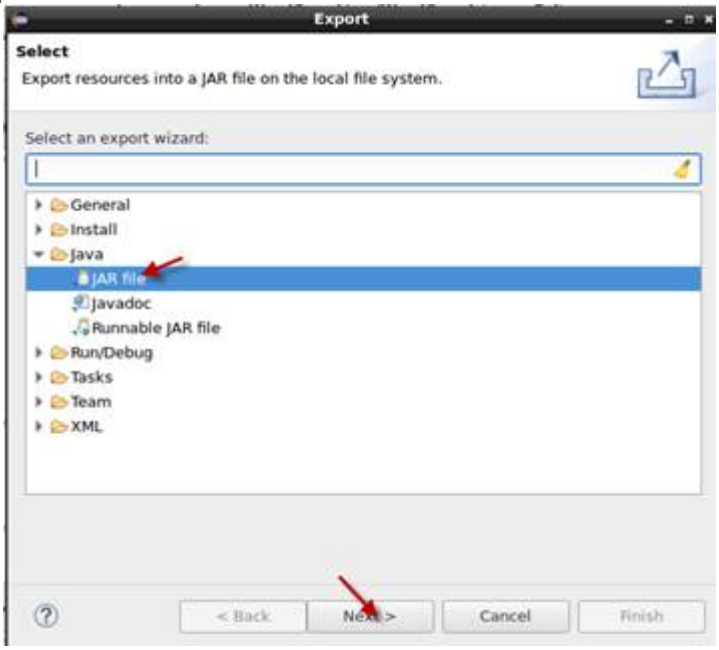
```
package WordCount;

import java.io.IOException;
import java.util.StringTokenizer;
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
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;
public class WordCount {

public static class TokenizerMapper
       extends Mapper<Object, Text, Text, IntWritable>{
    private final static IntWritable one = new IntWritable(1);
    private Text word = new Text();
    public void map(Object key, Text value, Context context
                    ) throws IOException, InterruptedException {
      StringTokenizer itr = new StringTokenizer(value.toString());
      while (itr.hasMoreTokens()) {
        word.set(itr.nextToken());
        context.write(word, one);
      }
    }
  }
```

```
  public static class IntSumReducer
       extends Reducer<Text,IntWritable,Text,IntWritable> {
    private IntWritable result = new IntWritable();

    public void reduce(Text key, Iterable<IntWritable> values,
                       Context context
                       ) throws IOException, InterruptedException {
      int sum = 0;
      for (IntWritable val : values) {
        sum += val.get();
      }
      result.set(sum);
      context.write(key, result);
    }
  }
  public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();
    Job job = Job.getInstance(conf, "word count");
    job.setJarByClass(WordCount.class);
    job.setMapperClass(TokenizerMapper.class);
    job.setCombinerClass(IntSumReducer.class);
    job.setReducerClass(IntSumReducer.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);
    FileInputFormat.addInputPath(job, new Path(args[0]));
    FileOutputFormat.setOutputPath(job, new Path(args[1]));
    System.exit(job.waitForCompletion(true) ? 0 : 1);
  }
}
```

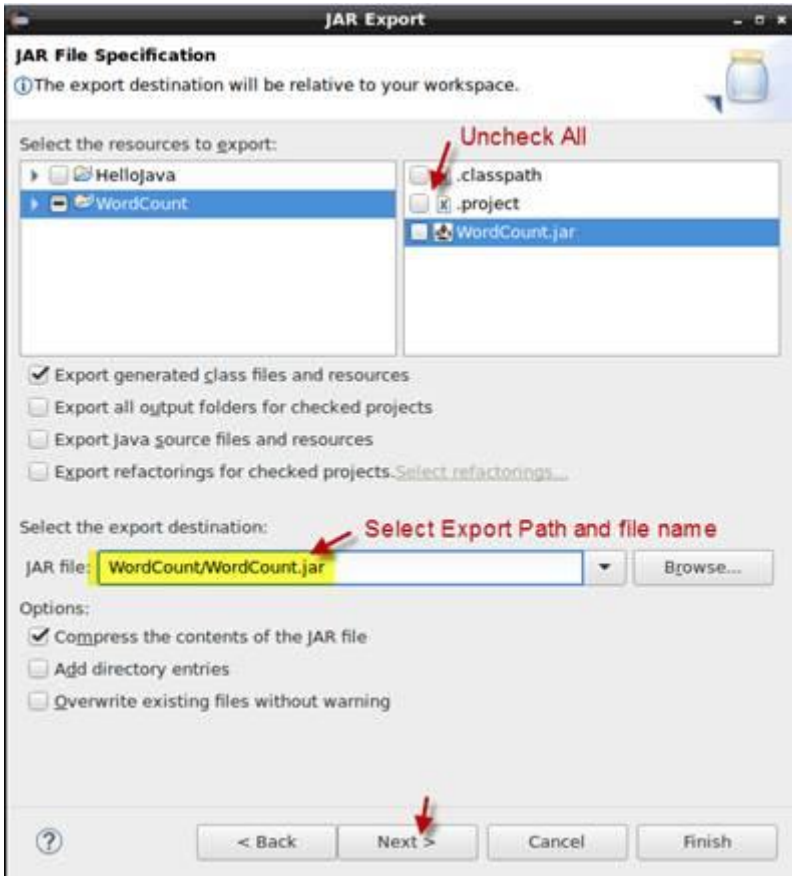
11.Make sure there are no errors in code typing. Then Right click on Project WordCount →**Export.**



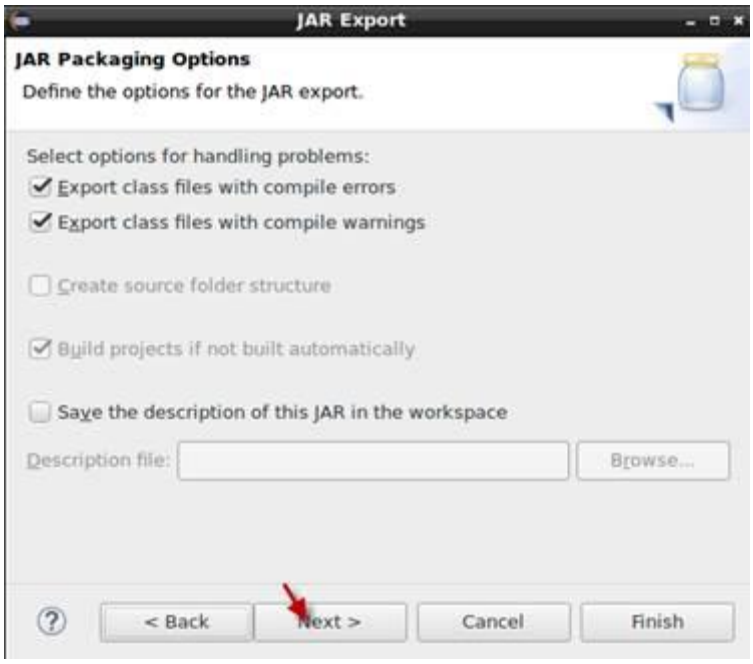
12. Select **JAR file** option under **Java**.



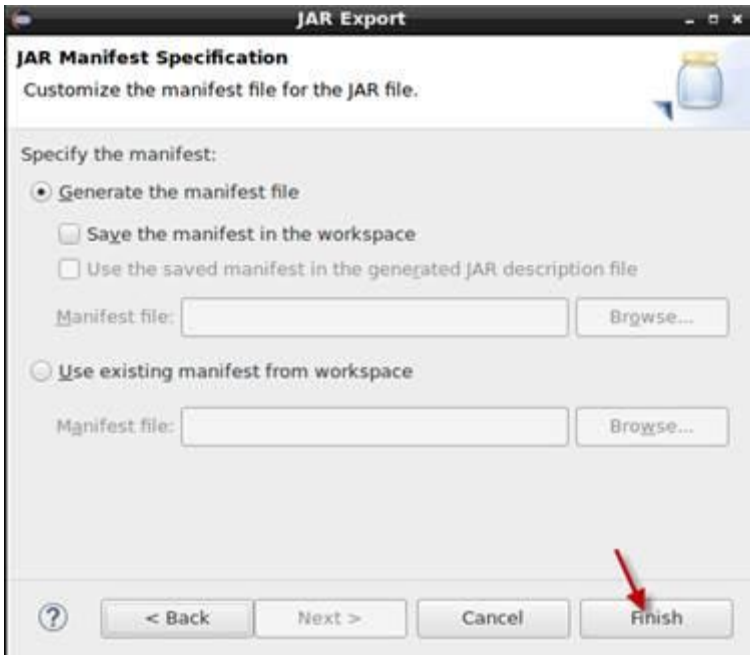
13.Click Next. Uncheck all other resources. Then provide path for exporting .jar file. It could be any path but remember where you exported it. You need this path for running jar file later.



14.Keep options selected as below.



15.Click Next and Finish.



Input Text File:

Input.txt :

```
hi how are you
welcome to velammal engineering college
We are from computer science department
Today we are gonna learn about mapreduce concept
Then we will apply the mapreduce programming knowlege to process the
text file
The output printed should be the count of occurrences of each word in
the text file
here ends our today program so bye.
```

Compiling the source:

Step 1: Starting all the HDFS services (if not running already)

 # start-dfs.sh

```
vecuser@ubuntu:~$ start-dfs.sh
Java HotSpot(TM) Client VM warning: You have loaded library
/usr/local/hadoop-2.7.7/lib/native/libhadoop.so.1.0.0 which might have
disabled stack guard. The VM will try to fix the stack guard now.
It's highly recommended that you fix the library with 'execstack -c
<libfile>', or link it with '-z noexecstack'.
19/03/20 04:29:12 WARN util.NativeCodeLoader: Unable to load native-
hadoop library for your platform... using builtin-java classes where
applicable
Starting namenodes on [localhost]
vecuser@localhost's password:
localhost: starting namenode, logging to /usr/local/hadoop-
2.7.7/logs/hadoop-vecuser-namenode-ubuntu.out
localhost: Java HotSpot(TM) Client VM warning: You have loaded library
/usr/local/hadoop-2.7.7/lib/native/libhadoop.so.1.0.0 which might have
disabled stack guard. The VM will try to fix the stack guard now.
localhost: It's highly recommended that you fix the library with
'execstack -c <libfile>', or link it with '-z noexecstack'.
vecuser@localhost's password:
localhost: starting datanode, logging to /usr/local/hadoop-
2.7.7/logs/hadoop-vecuser-datanode-ubuntu.out
Starting secondary namenodes [0.0.0.0]
vecuser@0.0.0.0's password:
0.0.0.0: starting secondarynamenode, logging to /usr/local/hadoop-
2.7.7/logs/hadoop-vecuser-secondarynamenode-ubuntu.out
0.0.0.0: Java HotSpot(TM) Client VM warning: You have loaded library
/usr/local/hadoop-2.7.7/lib/native/libhadoop.so.1.0.0 which might have
disabled stack guard. The VM will try to fix the stack guard now.
```

```
0.0.0.0: It's highly recommended that you fix the library with
'execstack -c <libfile>', or link it with '-z noexecstack'.
Java HotSpot(TM) Client VM warning: You have loaded library
/usr/local/hadoop-2.7.7/lib/native/libhadoop.so.1.0.0 which might have
disabled stack guard. The VM will try to fix the stack guard now.
It's highly recommended that you fix the library with 'execstack -c
<libfile>', or link it with '-z noexecstack'.
19/03/20 04:29:36 WARN util.NativeCodeLoader: Unable to load native-
hadoop library for your platform... using builtin-java classes where
applicable
```

 # start-yarn.sh

```
vecuser@ubuntu:~$ start-yarn.sh
starting yarn daemons
starting resourcemanager, logging to /usr/local/hadoop/logs/yarn-
vecuser-resourcemanager-ubuntu.out
vecuser@localhost's password:
localhost: starting nodemanager, logging to /usr/local/hadoop-
2.7.7/logs/yarn-vecuser-nodemanager-ubuntu.out
```

 # jps

```
vecuser@ubuntu:~$ jps
4289
11300 DataNode
11638 ResourceManager
12138 Jps
11147 NameNode
11485 SecondaryNameNode
11949 NodeManager
```

Step 2: Creating Input path in HDFS and moving the data into Input path

 # hadoop fs –mkdir /WordCount

```
vecuser@ubuntu:~$ hadoop fs -mkdir /WordCount
Java HotSpot(TM) Client VM warning: You have loaded library
/usr/local/hadoop-2.7.7/lib/native/libhadoop.so.1.0.0 which might have
disabled stack guard. The VM will try to fix the stack guard now.
It's highly recommended that you fix the library with 'execstack -c
<libfile>', or link it with '-z noexecstack'.
19/03/20 05:21:01 WARN util.NativeCodeLoader: Unable to load native-
hadoop library for your platform... using builtin-java classes where
applicable
```

 # hadoop fs –mkdir /WordCount/Input

```
vecuser@ubuntu:~$ hadoop fs -mkdir /WordCount/Input
Java HotSpot(TM) Client VM warning: You have loaded library
/usr/local/hadoop-2.7.7/lib/native/libhadoop.so.1.0.0 which might have
disabled stack guard. The VM will try to fix the stack guard now.
```

```
It's highly recommended that you fix the library with 'execstack -c
<libfile>', or link it with '-z noexecstack'.
19/03/20 05:21:08 WARN util.NativeCodeLoader: Unable to load native-
hadoop library for your platform... using builtin-java classes where
applicable
```

 # hadoop fs –put ‘home/hduser/Desktop/WordCount/Input/Input.txt’ /WordCount/Input

Executing the program

```
vecuser@ubuntu:~$ hadoop fs -put
'/home/vecuser/Desktop/WordCount/Input/Input.txt' /WordCount/Input
Java HotSpot(TM) Client VM warning: You have loaded library
/usr/local/hadoop-2.7.7/lib/native/libhadoop.so.1.0.0 which might have
disabled stack guard. The VM will try to fix the stack guard now.
It's highly recommended that you fix the library with 'execstack -c
<libfile>', or link it with '-z noexecstack'.
19/03/20 05:22:43 WARN util.NativeCodeLoader: Unable to load native-
hadoop library for your platform... using builtin-java classes where
applicable
```

Step 3 : Executing the program

 # hadoop jar ‘home/hduser/Desktop/WordCount/First.jar’ WordCount /WordCount/Input/Input.txt /WordCount/Output

```
vecuser@ubuntu:~$ hadoop jar
'/home/vecuser/Desktop/WordCount/first.jar' WordCount.WordCount
/WordCount/Input/Input.txt /WordCount/Output
Java HotSpot(TM) Client VM warning: You have loaded library
/usr/local/hadoop-2.7.7/lib/native/libhadoop.so.1.0.0 which might have
disabled stack guard. The VM will try to fix the stack guard now.
It's highly recommended that you fix the library with 'execstack -c
<libfile>', or link it with '-z noexecstack'.
19/03/20 08:24:24 WARN util.NativeCodeLoader: Unable to load native-
hadoop library for your platform... using builtin-java classes where
applicable
19/03/20 08:24:25 INFO client.RMProxy: Connecting to ResourceManager at
/0.0.0.0:8032
19/03/20 08:24:25 WARN mapreduce.JobResourceUploader: Hadoop command-
line option parsing not performed. Implement the Tool interface and
execute your application with ToolRunner to remedy this.
19/03/20 08:24:26 INFO input.FileInputFormat: Total input paths to
process : 1
19/03/20 08:24:26 INFO mapreduce.JobSubmitter: number of splits:1
19/03/20 08:24:26 INFO mapreduce.JobSubmitter: Submitting tokens for
job: job_1553095119373_0001
19/03/20 08:24:27 INFO impl.YarnClientImpl: Submitted application
application_1553095119373_0001
```

```
19/03/20 08:24:27 INFO mapreduce.Job: The url to track the job:
http://ubuntu:8088/proxy/application_1553095119373_0001/
19/03/20 08:24:27 INFO mapreduce.Job: Running job:
job_1553095119373_0001
19/03/20 08:24:34 INFO mapreduce.Job: Job job_1553095119373_0001
running in uber mode : false
19/03/20 08:24:34 INFO mapreduce.Job:  map 0% reduce 0%
19/03/20 08:24:39 INFO mapreduce.Job:  map 100% reduce 0%
19/03/20 08:24:46 INFO mapreduce.Job:  map 100% reduce 100%
19/03/20 08:24:47 INFO mapreduce.Job: Job job_1553095119373_0001
completed successfully
19/03/20 08:24:47 INFO mapreduce.Job: Counters: 49
       File System Counters
              FILE: Number of bytes read=588
              FILE: Number of bytes written=246271
              FILE: Number of read operations=0
              FILE: Number of large read operations=0
              FILE: Number of write operations=0
              HDFS: Number of bytes read=456
              HDFS: Number of bytes written=390
              HDFS: Number of read operations=6
              HDFS: Number of large read operations=0
              HDFS: Number of write operations=2
       Job Counters
              Launched map tasks=1
              Launched reduce tasks=1
              Data-local map tasks=1
              Total time spent by all maps in occupied slots (ms)=2832
              Total time spent by all reduces in occupied slots
(ms)=3738
              Total time spent by all map tasks (ms)=2832
              Total time spent by all reduce tasks (ms)=3738
              Total vcore-milliseconds taken by all map tasks=2832
              Total vcore-milliseconds taken by all reduce tasks=3738
              Total megabyte-milliseconds taken by all map
tasks=2899968
              Total megabyte-milliseconds taken by all reduce
tasks=3827712
       Map-Reduce Framework
              Map input records=8
              Map output records=59
              Map output bytes=579
              Map output materialized bytes=588
              Input split bytes=112
              Combine input records=59
              Combine output records=48
              Reduce input groups=48
              Reduce shuffle bytes=588
```

```
        Reduce input records=48
        Reduce output records=48
        Spilled Records=96
        Shuffled Maps =1
        Failed Shuffles=0
        Merged Map outputs=1
        GC time elapsed (ms)=146
        CPU time spent (ms)=910
        Physical memory (bytes) snapshot=250523648
        Virtual memory (bytes) snapshot=652423168
        Total committed heap usage (bytes)=137498624
    Shuffle Errors
        BAD_ID=0
        CONNECTION=0
        IO_ERROR=0
        WRONG_LENGTH=0
        WRONG_MAP=0
        WRONG_REDUCE=0
    File Input Format Counters
        Bytes Read=344
    File Output Format Counters
        Bytes Written=390
```

Please note that this program will read all text files from **input** folder on HDFS and count total number of words in each file. Output is stored under **output** folder on HDFS.

Once run is successful, you should see below 2 files in **output** folder.

hadoop fs -ls output

-rw-r--r-- 1 hduser supergroup 0 2021-10-19 13:54 output/_SUCCESS

-rw-r--r-- 1 hduser supergroup 73 2021-10-19 13:54 output/part-r-00000

_SUCCESS file is an empty file indicating RUN was successful. To see output of run, view part-r-00000 file.

OUTPUT :

Step 5 : To View the Output file

 # hadoop dfs –cat /WordCount/Output/part-r-00000

```
vecuser@ubuntu:~$ hadoop fs -cat /WordCount/Output/part-r-00000
Java HotSpot(TM) Client VM warning: You have loaded library
/usr/local/hadoop-2.7.7/lib/native/libhadoop.so.1.0.0 which might have
disabled stack guard. The VM will try to fix the stack guard now.
It's highly recommended that you fix the library with 'execstack -c
<libfile>', or link it with '-z noexecstack'.
19/10/21 14:25:40 WARN util.NativeCodeLoader: Unable to load native-
hadoop library for your platform... using builtin-java classes where
applicable
```

```
The    1
Then   1
Today  1
We     1
about  1
apply  1
are    3
be     1
bye.   1
college        1
computer       1
concept        1
count  1
department     1
each   1
ends   1
engineering    1
file   2
from   1
gonna  1
here   1
hi     1
how    1
in     1
knowlege       1
learn  1
mapreduce      2
occurrences    1
of     2
our    1
output 1
printed        1
process        1
program        1
programming    1
science        1
should 1
so     1
text   2
the    4
to     2
today  1
velammal       1
we     2
welcome        1
will   1
word   1
you    1
```

RESULT:

Thus the Word count program to use Map and reduce tasks is demonstrated successfully.