========================================================================

File handling:

=======================================================================

Task 1:

Run the below code and see the file with the given name created or not..

Run it again with “I like India” instead of “I love India”.. And see the file …

Import [java.io](http://java.io).\*;

Import java.io.IOException;

public class WriteByte

{

public static void main(String args[])

{

File f1=new File(“FileName01.txt”); \\ to create new file

FileOutputStream outfile = null;

byte Text[] = {'I',’ ‘,’'L','O','V','E',’ ‘,'I','N','D','I’,’A'};

try

{

outfile = new FileOutputStream(f1);

outfile.write(Text);

}

catch(IOException e)

{

System.out.println(e);

System.exit(-1);

}

System.out.println("Write Byte");

System.out.println("Thank You...!!!");

}

}

Task 2:

Try this code to see the output …

**Write a program which reads byte from file.**

import java.io.\*;

public class ReadingByte

{

public static void main(String args[])

{

FileInputStream infile = null;

int b;

try

{

infile = new FileInputStream("FileName01.txt");

while((b = infile.read()) != -1)

{

System.out.println((char)b);

}

infile.close();

}

catch(IOException e)

{

System.out.println("Sorry..!! File Not Found...!!!");

}

}

}

OutPUT: File not found

Task 3:

Taking input from the user and writing on the file…

Create  a file and see the output…

import java.io.\*;

import java.util.\*;

public class WriteByte\_1

{

public static void main(String args[]) {

FileOutputStream outfile = null;

//String s=args[0]; // to input string from command line Scanner sc=new Scanner(System.in);

String s=sc.nextLine();

byte b1[] = s.getBytes();

try

{

outfile = new FileOutputStream("FileName02.txt");

outfile.write(b1);

}

catch(IOException e)

{

System.out.println(e);

System.exit(-1);

}

System.out.println("Write Byte");

System.out.println("Thank You...!!!");

}

}

**Reading/writing characters**

=========================================

**FileReader** and **FileWriter**

**Task 4:**

**Write a program which creates file and writes character into that file.**

import java.io.\*;

class CharacterWrite {

public static void main(String[] args) {

File f1 = new File("FileName03.txt");

FileWriter fw = null;

try {

fw = new FileWriter(f1);

fw.write("ahmedabad \n");

fw.write("baroda \n");

fw.close();

} catch (FileNotFoundException e) {

System.out.println("Sorry..!! File Not Found...!!!");

} catch (IOException e) {

System.out.println(e.getMessage());

}

System.out.println("Write operation done!!");

}

}

OUTPUT: Sorry..!! File Not Found...!!!

Write operation done!!

Task 5:

**Write a program which reads character from file.**

import java.io.\*;

class ReadChar {

public static void main(String[] args) {

FileReader fr = null;

try {

fr = new FileReader("FileName03.txt");

int ch;

while ((ch = fr.read()) != -1) {

System.out.print((char) ch);

}

System.out.println("Reading complete");

fr.close();

} catch (FileNotFoundException e) {

System.out.println("Sorry..!! File Not Found...!!!");

} catch (IOException e) {

System.out.println(e.getMessage());

}

}

}

OUTPUT: Sorry..!! File Not Found...!!!

Note:

Task 4 and task 5 talks about File Reader and Writer classes which takes characters and display characters directly.. Unlike FileOUtputstream and File Output stream (which takes bytes).

FileWriter fw = new FileWriter("example.txt");

fw.write("Hello, world!");

fw.close();

FileInputStream fis = new FileInputStream("image.jpg");

FileOutputStream fos = new FileOutputStream("copy.jpg");

int b;

while ((b = fis.read()) != -1) {

fos.write(b);

}

fis.close();

fos.close();

Task 6:

**Write a program to read one byte at a time from a file and copy it into another  file immediately**.

import java.io.\*;

class CopyByte {

public static void main(String[] args) {

try {

FileInputStream infile = new FileInputStream("NewFile01.txt");

FileOutputStream outfile = new FileOutputStream("NewFile05.txt");

int byteread;

while ((byteread = infile.read()) != -1) {

outfile.write(byteread);

}

System.out.println("Byte Copied From NewFile01.txt to NewFile05.txt");

infile.close();

outfile.close();

} catch (FileNotFoundException e) {

System.out.println("Sorry..!! File Not Found...!!!");

} catch (IOException e) {

System.out.println(e.getMessage());

}

}

}

Task 7: Merging two files to 3rd file..

import java.io.\*;

class FileMergeDemo {

public static void main(String args[]) {

try {

// Create FileInputStream for both input files

FileInputStream file1 = new FileInputStream("NewFile01.txt");

FileInputStream file2 = new FileInputStream("NewFile02.txt");

// Use SequenceInputStream to combine the input streams

SequenceInputStream file3 = new SequenceInputStream(file1, file2);

// Create a BufferedInputStream for efficient reading

BufferedInputStream br1 = new BufferedInputStream(file3);

// Create FileOutputStream to write the merged content to a third file

FileOutputStream fileOut = new FileOutputStream("MergedFile.txt");

BufferedOutputStream br2 = new BufferedOutputStream(fileOut);

int ch;

// Read and write byte by byte from the merged input stream

while ((ch = br1.read()) != -1) {

br2.write(ch); // Write the byte to the output file

}

// Close all streams

br1.close();

br2.close();

file1.close();

file2.close();

System.out.println("Files merged successfully into MergedFile.txt");

} catch (IOException e) {

System.out.println("Sorry..!! File Not Found...!!!");

}

}

}

Output: Sorry..!! File Not Found...!!!

Timport java.io.\*;

class FileRenameDemo {

public static void main(String args[]) {

// Check if the correct number of arguments are passed

if (args.length != 2) {

System.out.println("Usage: java FileRenameDemo <old\_filename> <new\_filename>");

return;

}

// Create File objects for the old and new file names

File oldFile = new File(args[0]);

File newFile = new File(args[1]);ask 8:

**Write an application to rename a file. Use the renameTo() method of File to  accomplish**

/\*this task. The first command line argument is the old filename and the second is  the newfilename.

Lambda Expressions:

Task 9

import java.lang.FunctionalInterface;

\*/

import java.lang.FunctionalInterface;

@FunctionalInterface

interface MyInterface {

double getPiValue();

}

public class Main {

// Static method to return Pi value

public static double getPi() {

return 3.141592653589793;

}

public static void main(String[] args) {

// Lambda expression

MyInterface ref = () -> 3.1415;

System.out.println("Value of Pi from Lambda = " + ref.getPiValue());

// Method reference

MyInterface ref2 = Main::getPi;

System.out.println("Value of Pi from Method Reference = " + ref2.getPiValue());

}

}

OUTPUT: Value of Pi from Lambda = 3.1415

Value of Pi from Method Reference = 3.141592653589793

public class ReverseString {

public static void main(String[] args) {

// Input string

String original = "Hello, World!";

// Create a StringBuilder object with the original string

StringBuilder sb = new StringBuilder(original);

// Reverse the string using the reverse() method

sb.reverse();

// Output the reversed string

System.out.println("Reversed string: " + sb.toString());

}

}

Output: Reversed string: !dlroW ,olleH

Task 11:

@FunctionalInterface

@FunctionalInterface

interface MyInterface {

// abstract method

String reverse(String n);

}

public class Main {

public static void main(String[] args) {

// declare a reference to MyInterface

// assign a lambda expression to the reference

MyInterface ref = (str) -> {

String result = "";

// Loop through the string in reverse order

for (int i = str.length() - 1; i >= 0; i--)

result += str.charAt(i);

return result;

};

// call the method of the interface

System.out.println("Lambda reversed = " + ref.reverse("Lambda"));

}

}

Output: Lambda reversed = adambaL

Task 12:

Wap to create an arraylist with 5 friends names..

import java.util.ArrayList;

public class FriendsList {

public static void main(String[] args) {

// Create an ArrayList to store friends' names

ArrayList<String> friends = new ArrayList<>();

// Add 5 friends' names to the ArrayList

friends.add("Alice");

friends.add("Bob");

friends.add("Charlie");

friends.add("David");

friends.add("Eve");

// Print the list of friends

System.out.println("Friends List: ");

for (String friend : friends) {

System.out.println(friend);

}

}

}

Outpit: Friends List:

Alice

Bob

Charlie

David

Eve

Task 13: