JAVA.IO.BUFFEREDREADER CLASS

The **Java.io.BufferedReader** class reads text from a character-input stream, buffering characters so as to provide for the efficient reading of characters, arrays, and lines.Following are the important points about BufferedReader:

- The buffer size may be specified, or the default size may be used.
- Each read request made of a Reader causes a corresponding read request to be made of the underlying character or byte stream.

Class declaration for Java.io.BufferedReader **class:**

```
public class BufferedReader extends Reader
```

Java BufferedReader class constructors

Constructor	Description
BufferedReader(Reader rd)	It is used to create a buffered character input stream that uses the default size for an input buffer.
BufferedReader(Reader rd, int size)	It is used to create a buffered character input stream that uses the specified size for an input buffer.

Java BufferedReader class methods

Method	Description
int read()	It is used for reading a single character.
int read(char[] cbuf, int off, int len)	It is used for reading characters into a portion of an array.
boolean markSupported()	It is used to test the input stream support for the mark and reset method.
String readLine()	It is used for reading a line of text.
boolean ready()	It is used to test whether the input stream is ready to be read.
long skip(long n)	It is used for skipping the characters.

void reset()	It repositions the stream at a position the mark method was last called on this input stream.
void mark(int readAheadLimit)	It is used for marking the present position in a stream.
void close()	It closes the input stream and releases any of the system resources associated with the stream.

Reading data from console by InputStreamReader and BufferedReader

In this example, we are connecting the BufferedReader stream with the InputStreamReader stream for reading the line by line data from the keyboard.

```
package com.javatpoint;
import java.io.*;
public class BufferedReaderExample{
public static void main(String args[])throws Exception{
   InputStreamReader r=new InputStreamReader(System.in);
   BufferedReader br=new BufferedReader(r);
   System.out.println("Enter your name");
   String name=br.readLine();
   System.out.println("Welcome "+name);
}
}
```

Output:

```
Enter your name
Nakul Jain
Welcome Nakul Jain
```

Example of reading data from console until user writes *stop*

In this example, we are reading and printing the data until the user prints stop.

```
package com.javatpoint;
import java.io.*;
public class BufferedReaderExample{
public static void main(String args[])throws Exception{
    InputStreamReader r=new InputStreamReader(System.in);
    BufferedReader br=new BufferedReader(r);
    String name="";
    while(!name.equals("stop")){
     System.out.println("Enter data: ");
     name=br.readLine();
     System.out.println("data is: "+name);
    }
   br.close();
   r.close();
   }
   }
```

Output:

```
Enter data: Nakul
data is: Nakul
Enter data: 12
data is: 12
Enter data: stop
data is: stop
```

Difference between Scanner and BufferReader Class in Java

java.util.Scanner class is a simple text scanner which can parse primitive types and strings. It internally uses regular expressions to read different types.

Java.io.BufferedReader class reads text from a character-input stream, buffering characters so as to provide for the efficient reading of sequence of characters

Following are differences between above two.

Issue with Scanner when nextLine() is used after nextXXX()

Try to guess the output of following code:

```
// Code using Scanner Class
import java.util.Scanner;
class Differ
{
    public static void main(String args[])
    {
        Scanner scn = new Scanner(System.in);
        System.out.println("Enter an integer");
        int a = scn.nextInt();
        System.out.println("Enter a String");
        String b = scn.nextLine();
        System.out.printf("You have entered:- "
                + a + " " + "and name as " + b);
    }
}
```

Input:

```
50
Geek
```

Output:

```
Enter an integer
Enter a String
You have entered:- 50 and name as
```

NOTE: In Scanner class if we call nextLine() method after any one of the seven nextXXX() method then the nextLine() doesn't not read values from console and cursor will not come into console it will skip that step. The nextXXX() methods are nextInt(), nextFloat(), nextByte(), nextShort(), nextDouble(), nextLong(), next().

The same using Buffer class and same Input

```
// Code using Buffer Class
import java.io.*;
class Differ
{
   public static void main(String args[])
                throws IOException
   {
      BufferedReader br = new BufferedReader(new
      InputStreamReader(System.in));
      System.out.println("Enter an integer");
      int a = Integer.parseInt(br.readLine());
      System.out.println("Enter a String");
      String b = br.readLine();
      System.out.printf("You have entered:- " + a +
                     " and name as " + b);
   }
}
```

Input:

```
50
Geek
```

Output:

```
Enter an integer
Enter a String
you have entered:- 50 and name as Geek
```

In BufferReader class there is no such type of problem. This problem occurs only for Scanner class, due to nextXXX() methods ignore newline character and nextLine() only reads newline character. If we use one more call of nextLine() method between nextXXX() and nextLine(), then this problem will not occur because nextLine() will consume the newline character. ***This problem is same as scanf() followed by gets() in C/C++.***

Other differences:

- BufferedReader is synchronous while Scanner is not. BufferedReader should be used if we are working with multiple threads.
- BufferedReader has significantly larger buffer memory than Scanner.
- The Scanner has a little buffer (1KB char buffer) as opposed to the BufferedReader (8KB byte buffer), but it's more than enough.
- BufferedReader is a bit faster as compared to scanner because scanner does parsing of input data and BufferedReader simply reads sequence of characters.

Java I/O

Java Input/Output

FileOutputStream

FileInputStream

BufferedOutputStream

BufferedInputStream

SequenceInputStream

ByteArrayOutputStream

ByteArrayInputStream

DataOutputStream

DataInputStream

Java FilterOutputStream

Java FilterInputStream

Console

FilePermission

FileWriter

FileReader

BufferedWriter

BufferedReader

CharArrayReader

CharArrayWriter

PrintStream

PrintWriter

PushbackInputStream

PushbackReader

StringWriter

StringReader