**CSE3011 NETWORK PROGRAMMING**

**LAB EXPERIMENT 5**

NAME – B PRATYUSH

REGISTRATION NUMBER – 19BCN7114

LAB SLOT – L11+L12

FACULTY – PROF. MUNEESWARI

**Experiment Description: Encoding using URLEncoder class and UTF-8 report**

**CODE**

**URLEncodeTest.java**

import java.io.\*;

import java.net.\*;

public class URLEncodeTest{

public static void main(String[] args){

try{

//Example of a string with spaces

System.out.println(URLEncoder.encode("www waaa dexx xxxxywww","UTF-8"));

//Example of a string with . and @ characters

System.out.println(URLEncoder.encode("pratyush.19bcn@vitap.ac.in","UTF-8"));

//Example of a string with Back Slashes

System.out.println(URLEncoder.encode("D:/3rd YEAR FALL SEM/LAB/Network Programming Lab Experiments/Lab5","UTF-8"));

//Example of a string with Equality sign

System.out.println(URLEncoder.encode("a=b=c=d=e","UTF-8"));

//Example of string with tilde, asterix and at Sign with side slash

System.out.println(URLEncoder.encode("vit.123@~np\*\*lab5/url","UTF-8"));

//Example of string with period signs

System.out.println(URLEncoder.encode("192.168.1.1","UTF-8"));

//Example of String with ampersands

System.out.println(URLEncoder.encode("This&string&has&ampersands","UTF-8"));

//Example of String with parenthesis

System.out.println(URLEncoder.encode("This(string)has(parentheses)","UTF-8"));

//Example of String with Non-Ascii Characters

System.out.println(URLEncoder.encode("Thiséstringéhasénon-ASCII characters", "UTF-8"));

//Example of String with plus sign

System.out.println(URLEncoder.encode("a+b+c+D+e", "UTF-8"));

//Example of String with quote marks

System.out.println(URLEncoder.encode("Hello\"Welcome\"to\"NP\"Lab","UTF-8"));

//Example of String with colons

System.out.println(URLEncoder.encode("aed0:12e5:ab12:45fa","UTF-8"));

//Example of String with percentage sign

System.out.println(URLEncoder.encode("a&b&c&d","UTF-8"));

}

catch (UnsupportedEncodingException ex) {

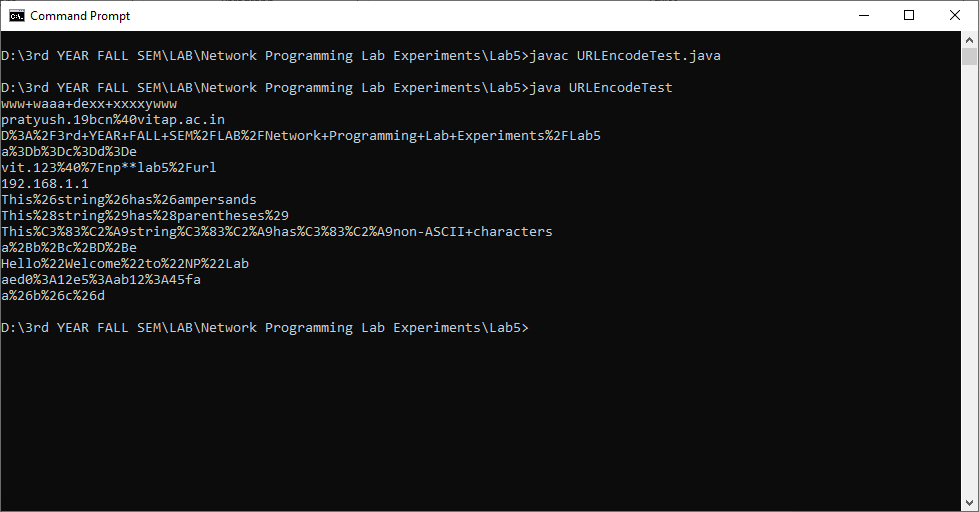
throw new RuntimeException("Broken VM does not support UTF-8");

}

}

}

**Output:**

****

**UTF-8 Encoding in Java**

UTF-8 is the default encoding mechanism used by Java. It is a variable- width character encoding scheme specially used for electronic communication. Since it is defined by the Unicode standard, the name utf-8 is derived from Unicode or Universal Coded Character.

It is capable of encoding all 1,112,064 valid character code points in Unicode using one to four one-[byte](https://en.wikipedia.org/wiki/Byte) (8-bit) code units.

Code points with lower numerical values, which tend to occur more frequently, are encoded using fewer bytes. It was designed for [backward compatibility](https://en.wikipedia.org/wiki/Backward_compatibility) with [ASCII](https://en.wikipedia.org/wiki/ASCII): the first 128 characters of Unicode, which correspond one-to-one with ASCII, are encoded using a single byte with the same binary value as ASCII, so that valid ASCII text is valid UTF-8-encoded Unicode as well. Since ASCII bytes do not occur when encoding non-ASCII code points into UTF-8, UTF-8 is safe to use within most programming and document languages that interpret certain ASCII characters in a special way, such as / ([slash](https://en.wikipedia.org/wiki/Slash_(punctuation))) in filenames, \ ([backslash](https://en.wikipedia.org/wiki/Backslash)) in [escape sequences](https://en.wikipedia.org/wiki/String_literal#Escape_sequences), and % in [printf](https://en.wikipedia.org/wiki/Printf" \o "Printf).

UTF-8 was designed as a superior alternative to [UTF-1](https://en.wikipedia.org/wiki/UTF-1), a proposed variable-width encoding with partial ASCII compatibility which lacked some features including [self-synchronization](https://en.wikipedia.org/wiki/Self-synchronizing_code) and fully ASCII-compatible handling of characters such as slashes. [Ken Thompson](https://en.wikipedia.org/wiki/Ken_Thompson) and [Rob Pike](https://en.wikipedia.org/wiki/Rob_Pike) produced the first implementation for the [Plan 9](https://en.wikipedia.org/wiki/Plan_9_from_Bell_Labs) operating system in September 1992. This led to its adoption by [X/Open](https://en.wikipedia.org/wiki/X/Open) as its specification for *FSS-UTF*, which would first be officially presented at [USENIX](https://en.wikipedia.org/wiki/USENIX) in January 1993 and subsequently adopted by the [Internet Engineering Task Force](https://en.wikipedia.org/wiki/Internet_Engineering_Task_Force) (IETF) in RFC 2277 (BCP 18) for future Internet standards work, replacing Single Byte Character Sets such as [Latin-1](https://en.wikipedia.org/wiki/Latin-1) in older RFCs.

UTF-8 is by far the most common encoding for the [World Wide Web](https://en.wikipedia.org/wiki/World_Wide_Web), accounting for over 97% of all web pages, and up to 100% for some languages, as of 2021.

**UTF-8** is a [variable-width](https://en.wikipedia.org/wiki/Variable-width_encoding) [character encoding](https://en.wikipedia.org/wiki/Character_encoding) used for electronic communication. Defined by the Unicode Standard, the name is derived from Unicode (or Universal Coded Character Set) Transformation Format – 8-bit.

UTF-8 is capable of encoding all 1,112,064valid character [code points](https://en.wikipedia.org/wiki/Code_point) in [Unicode](https://en.wikipedia.org/wiki/Unicode) using one to four one-[byte](https://en.wikipedia.org/wiki/Byte) (8-bit) code units. Code points with lower numerical values, which tend to occur more frequently, are encoded using fewer bytes. It was designed for [backward compatibility](https://en.wikipedia.org/wiki/Backward_compatibility) with [ASCII](https://en.wikipedia.org/wiki/ASCII): the first 128 characters of Unicode, which correspond one-to-one with ASCII, are encoded using a single byte with the same binary value as ASCII, so that valid ASCII text is valid UTF-8-encoded Unicode as well. Since ASCII bytes do not occur when encoding non-ASCII code points into UTF-8, UTF-8 is safe to use within most programming and document languages that interpret certain ASCII characters in a special way, such as / ([slash](https://en.wikipedia.org/wiki/Slash_(punctuation))) in filenames, \ ([backslash](https://en.wikipedia.org/wiki/Backslash)) in [escape sequences](https://en.wikipedia.org/wiki/String_literal#Escape_sequences), and % in [printf](https://en.wikipedia.org/wiki/Printf" \o "Printf).

UTF-8 was designed as a superior alternative to [UTF-1](https://en.wikipedia.org/wiki/UTF-1), a proposed variable-width encoding with partial ASCII compatibility which lacked some features including [self-synchronization](https://en.wikipedia.org/wiki/Self-synchronizing_code) and fully ASCII-compatible handling of characters such as slashes. [Ken Thompson](https://en.wikipedia.org/wiki/Ken_Thompson) and [Rob Pike](https://en.wikipedia.org/wiki/Rob_Pike) produced the first implementation for the [Plan 9](https://en.wikipedia.org/wiki/Plan_9_from_Bell_Labs) operating system in September 1992. This led to its adoption by [X/Open](https://en.wikipedia.org/wiki/X/Open) as its specification for *FSS-UTF*, which would first be officially presented at [USENIX](https://en.wikipedia.org/wiki/USENIX) in January 1993 and subsequently adopted by the [Internet Engineering Task Force](https://en.wikipedia.org/wiki/Internet_Engineering_Task_Force) (IETF) in RFC 2277 (BCP 18) for future Internet standards work, replacing Single Byte Character Sets such as [Latin-1](https://en.wikipedia.org/wiki/Latin-1) in older RFCs.

UTF-8 is by far the most common encoding for the [World Wide Web](https://en.wikipedia.org/wiki/World_Wide_Web), accounting for over 97% of all web pages, and up to 100% for some languages, as of 2021.

UTF-8 is defined to encode code points in one to four bytes, depending on the number of significant bits in the numerical value of the code point. The following table shows the structure of the encoding. The x characters are replaced by the bits of the code point.

Utility class for HTML form encoding. This class contains static methods for converting a String to the application/x-www-form-urlencoded MIME format. For more information about HTML form encoding, consult the HTML [specification](http://www.w3.org/TR/html4/).

When encoding a String, the following rules apply:

* The alphanumeric characters "a" through "z", "A" through "Z" and "0" through "9" remain the same.
* The special characters ".", "-", "\*", and "\_" remain the same.
* The space character " " is converted into a plus sign "+".
* All other characters are unsafe and are first converted into one or more bytes using some encoding scheme. Then each byte is represented by the 3-character string "%*xy*", where *xy* is the two-digit hexadecimal representation of the byte. The recommended encoding scheme to use is UTF-8. However, for compatibility reasons, if an encoding is not specified, then the default encoding of the platform is used.

For example using UTF-8 as the encoding scheme the string "The string ü@foo-bar" would get converted to "The+string+%C3%BC%40foo-bar" because in UTF-8 the character ü is encoded as two bytes C3 (hex) and BC (hex), and the character @ is encoded as one byte 40 (hex).

**Modifier and type**

static String

**Method Description**

**encode(String s ,String enc)**

The passed string s with be encoded according the applied encoding mechanism enc. Here we can pass UTF-8 as enc and any string as s to encode String s to utf-8 encoding

public static [String](https://docs.oracle.com/javase/7/docs/api/java/lang/String.html) encode([String](https://docs.oracle.com/javase/7/docs/api/java/lang/String.html) s,[String](https://docs.oracle.com/javase/7/docs/api/java/lang/String.html) enc)

throws [UnsupportedEncodingException](https://docs.oracle.com/javase/7/docs/api/java/io/UnsupportedEncodingException.html)

Translates a string into application/x-www-form-urlencoded format using a specific encoding scheme. This method uses the supplied encoding scheme to obtain the bytes for unsafe characters.