**String класстай ажиллах**

Текстэн өгөгдөл гэж юу вэ ? Энэ нь салангад тэмдэгтүүдийн цуглуулга гэсэн утгатай үг юм. Энэ төрлийн текстэн өгөгдлийг жава д String классыг ашиглан хадгалдаг юм. Энэ текстэн өгөгдлүүд нь давхар хашилтанд бичигдэх ба дотор нь ямар ч төрлийн тэмдэгтүүд орж болно. Эдгээр тэмдэгтэн төрлийг үүсгэсний дараа өөрчилж болохгүй тогтмол утгатай байдаг. Жава д эдгээр тэмдэгтүүд нь String классын объектууд юм. String класс нь дараах төрлийн байгуулагч фунцүүдтэй.

String s = new String ( ); // Хоосон тэмдэгт мөр үүсгэх  
String b = new String(“hello”); //Тэмдэгт мөрөөс үүсгэх  
String s = new String(b);   
char chars[ ] = {‘a’,’b’,’c’}; //Тэмдэгт массиваас үүсгэх  
String s = new String(chars)  
char chars[ ] = {‘a’,’b’,’c’,’d’,’e’,’f’}; //Тэмдэгт массивын тодорхой хэсгээс үүсгэх  
String s = new String (chars, 2, 3);  
byte chars[ ] = {‘a’,’b’,’c’,’d’,’e’,’f’}; //Byte массиваас үүсгэх  
String s = new String (chars);  
byte chars[ ] = {‘a’,’b’,’c’,’d’,’e’,’f’}; //Byte массивын тодорхой хэсгээс үүсгэх  
String s = new String (chars,3,5);

**Тэмдэгт мөрийн урттай ажиллах**

Тэмдэгт мөрийн урт гэдэг нь дотор нь агуулагдах тэмдэгтүүдийн тоо хэмжээг хэлж байгаа юм. Эдгээр утгуудыг авахын тулд length( ) функцийг дууддаг. Энэ функц нь тэмдэгт мөрийн уртыг буцаадаг ба ингэснээрээ массивтай адилхан юм. Харин массиваас ялгаатай тал нь length гэсэн хувьсагчгүй байдаг.

**Тэмдэгт мөрүүдийг нэгтгэхэд + операторыг ашигладаг. Жишээ нь**

**class** ConCat {

**public** **static** **void** main(String args[]) {

String age = "9";

String s = "He is " + age + " years old.";

System.***out***.println(s);

}

}

**Үр дүн нь:**  He is 9 years old.  
**Бусад өгөгдлийн төрөлтэй хамт тэмдэгт мөрийг нэгтгэх.**

**class** ConCat {

**public** **static** **void** main(String args[]) {

**int** age = 9;

String s = "He is " + age + " years old.";

System.***out***.println(s);

}

}

**Үр дүн нь:**  He is 9 years old.

**Тэмдэгт мөрийн хувирал ба toString( ) функц**

Энэ функц нь аливаа объектийг тэмдэгт мөр хэлбэрээр гаргахад хэрэглэгддэг функц. Яагаад гэвэл энэ функц нь Object класст тодорхойлогдсон байдаг тул бид үүнийг дахин тодорхойлох замаар ашиглаж байга гэсэн үг юм.

Жишээ нь:

**class** Box {

**double** width;

**double** height;

**double** depth;

Box(**double** w, **double** h, **double** d) {

width = w;

height = h;

depth = d;

}

**public** String toString() {

**return** "Dimensions are " + width + " by " +

depth + " by " + height + ".";

}

}

**class** toStringDemo {

**public** **static** **void** main(String args[]) {

Box b = **new** Box(10, 12, 14);

String s = "Box b: " + b; // concatenate Box object

System.***out***.println(b); // convert Box to string

System.***out***.println(s);

}

}

**Үр дүн:**

Dimensions are 10.0 by 14.0 by 12.0.

Box b: Dimensions are 10.0 by 14.0 by 12.0.

**charAt( )**

Тэмдэгт мөрөөс дурын тэмдэгтийг салгаж авч болно. Тэмдэгт мөрөөс нэг тэмдэгтийг салгаж авахдаа charAt( ) функцийг хэрэглэдэг. Ерөнхий хэлбэр нь

*char charAt( int where)* ***where*** гэдэг нь хэддүгээр тэмдэгтийг салгаж авах тоо. Энэ нь хасах утга байж болохгүй ба энэ нь тэмдэгт мөр доторх тухайн байршлыг зааж байга юм.

char ch;  
ch = “abc”.charAt(1); Энэ нь ch тэмдэгт мөрийн b гэсэн утга руу хандаж байна гэсэн үг.

**getChars( )**

Хэрвээ та нэгээс илүү тэмдэгтийг салгаж авах хэрэг гарвал getChars( ) функцийг хэрэглэж болно. Энэ нь тэмдэгт мөр хэлбэрээр буцаана.

**class** getCharsDemo {

**public** **static** **void** main(String args[]) {

String s = "This is a demo of the getChars method.";

**int** start = 10;

**int** end = 14;

**char** buf[] = **new** **char**[end - start];

s.getChars(start, end, buf, 0);

System.***out***.println(buf);

}

}

**Үр дүн:** demo

**toCharArray( )** Энэ нь тэмдэгтийг массив байдлаар буцаадаг функц.

**Тэмдэгт мөрийн харьцуулалт**

String класс нь дотроо тэмдэгт мөрүүдийг харьцуулах функцүүдтэй. Эдгээр функцүүд нь equals, equalsIgnoreCase, compareTo, regionMatches юм. Эдгээр функцүүд нь хоёр тэмдэгтийг харьцуулахад мөр бүрийн тэмдэгтүүдийн Unicode –г хооронд нь тулгадаг.

**class** equalsDemo {

**public** **static** **void** main(String args[]) {

String s1 = "Hello";

String s2 = "Hello";

String s3 = "Good-bye";

String s4 = "HELLO";

System.***out***.println(s1 + " equals " + s2 + " -> " +

s1.equals(s2));

System.***out***.println(s1 + " equals " + s3 + " -> " +

s1.equals(s3));

System.***out***.println(s1 + " equals " + s4 + " -> " +

s1.equals(s4));

System.***out***.println(s1 + " equalsIgnoreCase " + s4 + " -> " +

s1.equalsIgnoreCase(s4));

}

}

**Үр дүн:**

Hello equals Hello -> true

Hello equals Good-bye -> false

Hello equals HELLO -> false

Hello equalsIgnoreCase HELLO -> true

**regionMatches( )** функц нь хоёр тэмдэгт мөр объектийн хэсгийг харьцуулна. **startWish( )** болон **endWish( )** функцүүд нь тухайн тэмдэгтээр эхэлсэн болон төгссөн эсэхийг шалгадаг функц.

**compareTo( )** функц нь

Хэрэв compareTo функцийг дуудсан тэмдэгт мөр нь параметр байдлаар дамжсан тэмдэгт мөрөөсөө бага байвал сөрөг утга буцаана, үгүй бол эерэг утга буцаана. Харин хоёр тэмдэгт мөр тэнцүү бол тэг гэсэн утга буцаана. Бид тэмдэгт мөр хаьцуулах дээр = = операторыг ашиглаж болохгүй, харин өгөгдлийн үндсэн төрөл дээр ашиглах нь зүйтэй. = = операторыг энгийн төрөл дээр ашиглавал утга нь тэнцүү эсэхийг шалгана, заалт төрөлд ашиглавал уг хоёр объект санах ойд нэг объект байгаа бол л үнэн утга буцаадаг. Хэрвээ хоёр тусдаа объект байсан бол ижил утгатай байсан ч худал гэсэн утга буцаадаг.

**Тэмдэгт мөрийн байршил тодорхойлох indexOf( ), lastIndexOf( )**

**idexOf( )** функц нь тухайн тэмдэгтийн тэмдэгт мөр дэх хамгийн эхний байршлыг олно. **lastIndexOf** функц нь тухайн тэмдэгтийн тэмдэгт мөр дэх хамгийн сүүлийн байршлыг олно.

**class** indexOfDemo {

**public** **static** **void** main(String args[]) {

String s = "Now is the time for all good men " +

"to come to the aid of their country.";

System.***out***.println(s);

System.***out***.println("indexOf(t) = " +

s.indexOf('t'));

System.***out***.println("lastIndexOf(t) = " +

s.lastIndexOf('t'));

System.***out***.println("indexOf(the) = " +

s.indexOf("the"));

System.***out***.println("lastIndexOf(the) = " +

s.lastIndexOf("the"));

System.***out***.println("indexOf(t, 10) = " +

s.indexOf('t', 10));

System.***out***.println("lastIndexOf(t, 60) = " +

s.lastIndexOf('t', 60));

System.***out***.println("indexOf(the, 10) = " +

s.indexOf("the", 10));

System.***out***.println("lastIndexOf(the, 60) = " +

s.lastIndexOf("the", 60));

}

}

**Үр дүн:**

Now is the time for all good men to come to the aid of their country.

indexOf(t) = 7

lastIndexOf(t) = 65

indexOf(the) = 7

lastIndexOf(the) = 55

indexOf(t, 10) = 11

lastIndexOf(t, 60) = 55

indexOf(the, 10) = 44

lastIndexOf(the, 60) = 55

**Тэмдэгт мөрөнд өөрчлөлт оруулах**

Эдгээр төрлийн функцүүд нь **substring( ), concat( ), replace( ), trim( )** гэсэн функцүүд орно. **substring( )** функц нь тэмдэгт мөрөөс өөр тэмдэгт мөрийг бий болгоход хэрэглэгддэг. **concat( )** функц нь хоёр тэмдэгт мөрийг залгасан шинэ тэмдэгт мөр үүсгэдэг функц. **replace( )** функц нь тэмдэгт мөрийн тухайн нэг тэмдэгтийг өөр нэгэн тэмдэгтээр солидог функц. **trim( )** функц нь мөрийн эхлэл төгсгөлд байгаа хоосон зайг авч хаядаг функц.

**class** ModifyString {

**public** **static** **void** main(String[] args){

//Advenced Java Syntax

String course = "Java Foundations";

String Scourse = course.substring(5);

System.***out***.println(Scourse);

Scourse = course.substring(5, 11);

System.***out***.println(Scourse);

System.***out***.println("Advenced Java Syntax");

System.***out***.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

String s1 = "Java";

String s2 = "Foundations";

String s3 = "JavaFoundations";

System.***out***.println("s1 = "+s1);

System.***out***.println("s2 = "+s2);

System.***out***.println("s3 = "+s3);

System.***out***.println("--------------------");

s1+=s2; // s1 is now JavaFoundations

**if** (s1.compareTo(s2)>0) { //харьцуулах үйлдэл

System.***out***.println("Greater Than");

}**else**{

System.***out***.println("Less Than");

}

String ss1 = " Java ";

String ss2 = " Foundations ";

String result1;

result1 = ss1.concat(ss2);

System.***out***.println(result1);

result1 = result1.trim();

System.***out***.println(result1);

result1 = result1.replace('a', 'o');

System.***out***.println(result1);

System.***out***.println(result1.replace('a', 'o'));

}

}

**Үр дүн:**

Foundations

Founda

Advenced Java Syntax

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

s1 = Java

s2 = Foundations

s3 = JavaFoundations

--------------------

Greater Than

Java Foundations

Java Foundations

Jovo Foundotions

Jovo Foundotions

**valueOf( ) функц**

Энэ функц нь String классын статик функц. Объект, өгөгдлүүдийг тэмдэгт мөр хэлбэрээр буцаана.

**import** javax.swing.JOptionPane;

**public** **class** StringValueOf {

**public** **static** **void** main(String[] args) {

**char** charArray[]={'a','b','c','d','e','f'};

**boolean** booleanValue=**true**;

**char** characterValue='Z';

**int** integerValue=7;

**long** longValue=10000000;

**float** floatValue=2.5f;

**double** doubleValue=33.333;

Object objectRef="hello";

String output="char array="+String.*valueOf*(charArray)+

"\npart of char array="+String.*valueOf*(charArray,3,3)+

"\nboolean="+String.*valueOf*(booleanValue)+

"\nchar="+String.*valueOf*(characterValue)+

"\nint="+String.*valueOf*(integerValue)+

"\nlong="+String.*valueOf*(longValue)+

"\nfloat="+String.*valueOf*(floatValue)+

"\ndouble="+String.*valueOf*(doubleValue)+

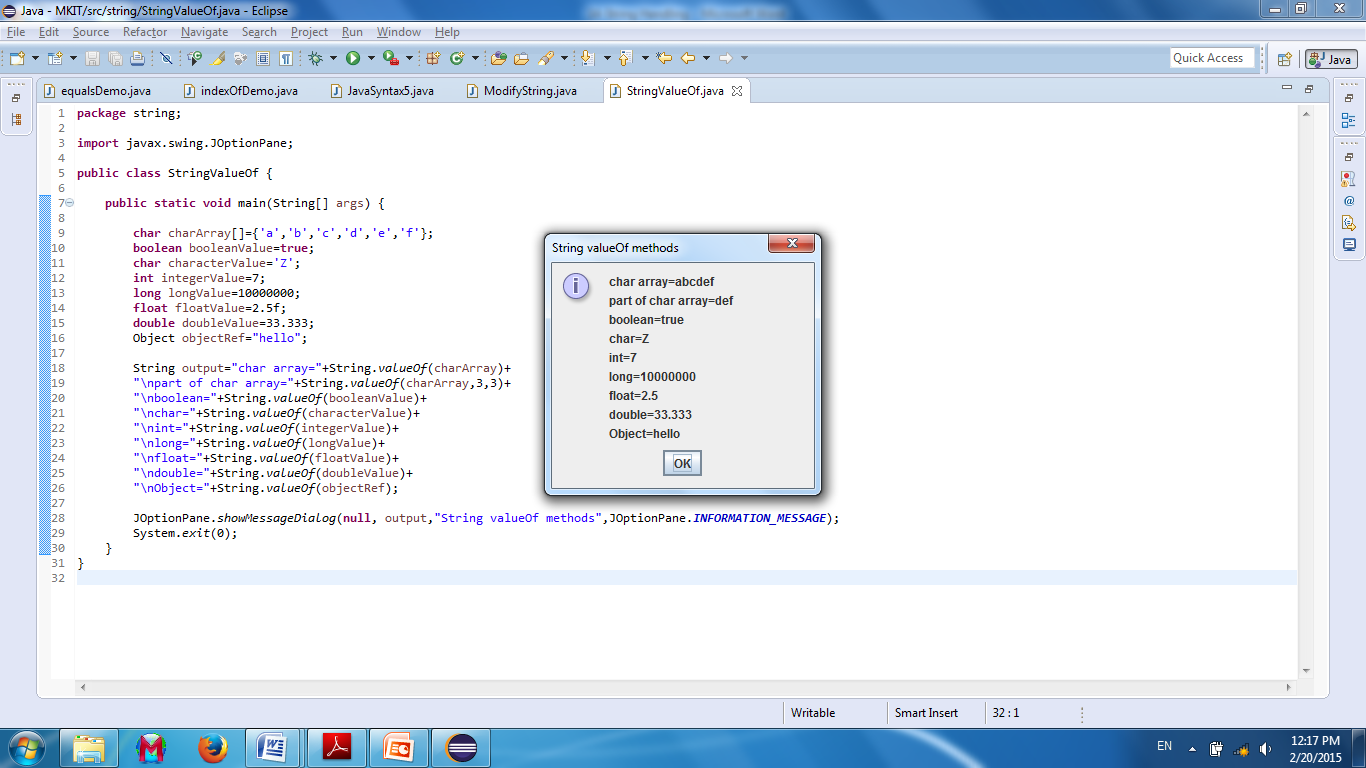
"\nObject="+String.*valueOf*(objectRef);

JOptionPane.*showMessageDialog*(**null**, output,"String valueOf methods",JOptionPane.***INFORMATION\_MESSAGE***);

System.*exit*(0);

}

}



Функц toUpperCase( ) Тэмдэгт мөрийн бүх үсгийг жижиг болгосон мөрийг гарган авч болно.

Функц toLowerCase( ) Тэмдэгт мөрийн бүх үсгийг жижиг болгосон мөрийг гарган авч болно.

**import** javax.swing.JOptionPane;

**public** **class** StringMiscellaneous {

**public** **static** **void** main(String[] args) {

String s1=**new** String("hello");

String s2=**new** String("GOODBYE");

String s3=**new** String(" spaces ");

String output="s1="+s1+"\ns2="+s2+"\ns3="+s3;

output+="\n\nReplace 'l' with 'L' in s1:"+s1.replace('l', 'L');

output+="\n\ns1.toUpperCase()="+s1.toUpperCase()+"\ns2.toLowerCase()="+s2.toLowerCase();

output+="\n\ns3 after trim=\""+s3.trim()+"\"";

**char** charArray[]=s1.toCharArray();

output+="\n\ns1 as a character array=";

**for** (**int** count = 0; count < charArray.length; ++count)

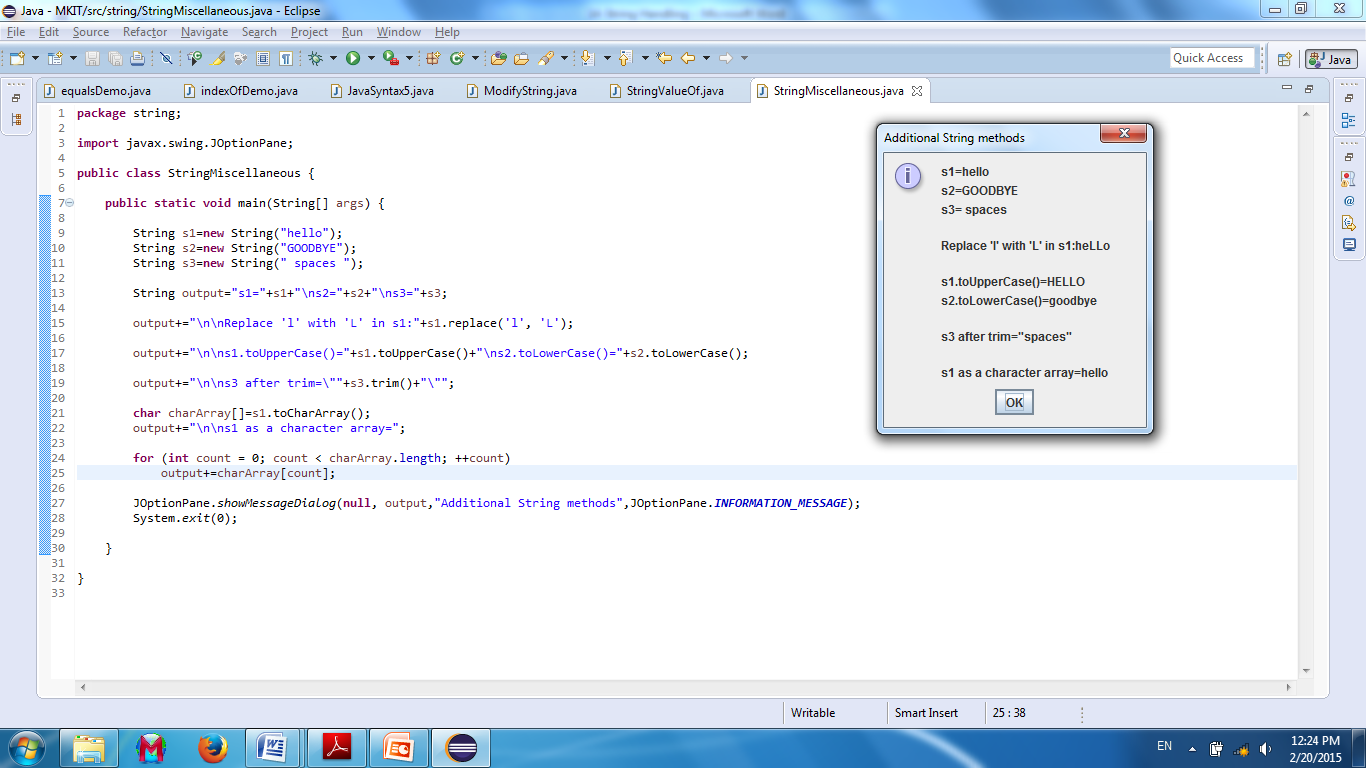
output+=charArray[count];

JOptionPane.*showMessageDialog*(**null**, output,"Additional String methods",JOptionPane.***INFORMATION\_MESSAGE***);

System.*exit*(0);

}

}



**Тэмдэгт мөрүүдийг холбох join( ) функц**

Энэ функц нь JDK 8 –дээр String классд нэмэгдэж орж ирсэн болно. Энэ функцийг хоёр болон түүнээс олон тэмдэгтүүдийг нийлүүлэхэд хэргэлдэг.

*static String join(CharSequence delim, CharSequence . . . strs)* Энд *delim* гэдэг нь тэмдэгт болгоныг тусгаарлагч зүйл хоосон зай болон таслал шиг. s*trs*  гэдэг нь таны нийлүүлэх гэж байгаа тэмдэгт мөрүүд юм.

**Жишээ нь:**

**class** StringJoinDemo {

**public** **static** **void** main(String args[]) {

String result = String.*join*(", ", "Alpha", "Beta", "Gamma");

System.***out***.println(result);

result = String.*join*(", ", "Odkhuu", "ID#: 999", "E-mail: l.odkhuu@mkit.com");

System.***out***.println(result);

}

}

**Үр дүн:**

Alpha, Beta, Gamma

Odkhuu, ID#: 999, E-mail: l.odkhuu@mkit.com

**StringBuffer класс**

Бид дээр String классын объектийг үүсгэхэд дараа нь уг объектийн утгыг өөрчилж болдоггүй талаар ярьж байсан. Тэгвэл энэ **StringBuffer** классыг ашиглан утгыг нь өөрчилдөг тэмдэгт мөрийг үүсгэж болно. StringBuffer класс нь доорх дөрвөн байгуулагч функцээр тодорхойлдог.

*StringBuffer( )*

*StringBuffer( int size)*

*StringBuffer( String str)*

*StringBuffer( CharSequence chars)*

StringBuffer классын объектод агуулах багтаамж гэж байх ба хэрвээ багтаамжаас илүү урттай тэмдэгт мөр олговол багтаамж нь автоматаар нэмэгддэг.Объектийн багтаамж нь объектийн анхны утгын уртаас 16 тэмдэгтээр илүүгээр тавигдда. StringBuffer классын тэмдэгт мөрүүдийг + оператор ашиглан шууд залгаж болдог.

**StringBuffer классын функцүүд:**

Функц **length** - StringBuffer-н уртыг буцаана. Функц **capacity** - StringBuffer багтаамжийг буцаана. Функц **setLength** - StringBuffer-н уртыг ихэсгэж багасгана. Функц **ensureCapacity** - StringBuffer-н багтаамжийг өөрчилнө. StringBuffer-д минимум багтаамж өгнө. Функц **charAt** - Заасан индекс дэх StringBuffer –н тэмдэгтийг буцаана. Функц **setCharAt** - Заасан индекс дэх StringBuffer –н тэмдэгтийг өөрчилнө. Функц **getChars** - StringBuffer –с тэмдэгтийн массив буцаана. Функц **reverse** - StringBuffer-н утгуудын дарааллыг эсрэгээр нь болгоно. Функц **append** - StringBuffer-д утга нэмж оруулна. Функц **insert** - StringBufferкласст утга оруулах

Функц **delete, deleteCharAt** - StringBuffer-с утга устгах.

**import** javax.swing.JOptionPane;

**public** **class** StringBufferConstructors {

**public** **static** **void** main(String[] args) {

StringBuffer buffer1=**new** StringBuffer();

StringBuffer buffer2=**new** StringBuffer(10);

StringBuffer buffer3=**new** StringBuffer("hello");

String output="buffer1=\""+buffer1.toString()+"\""+ "\nbuffer2=\""+

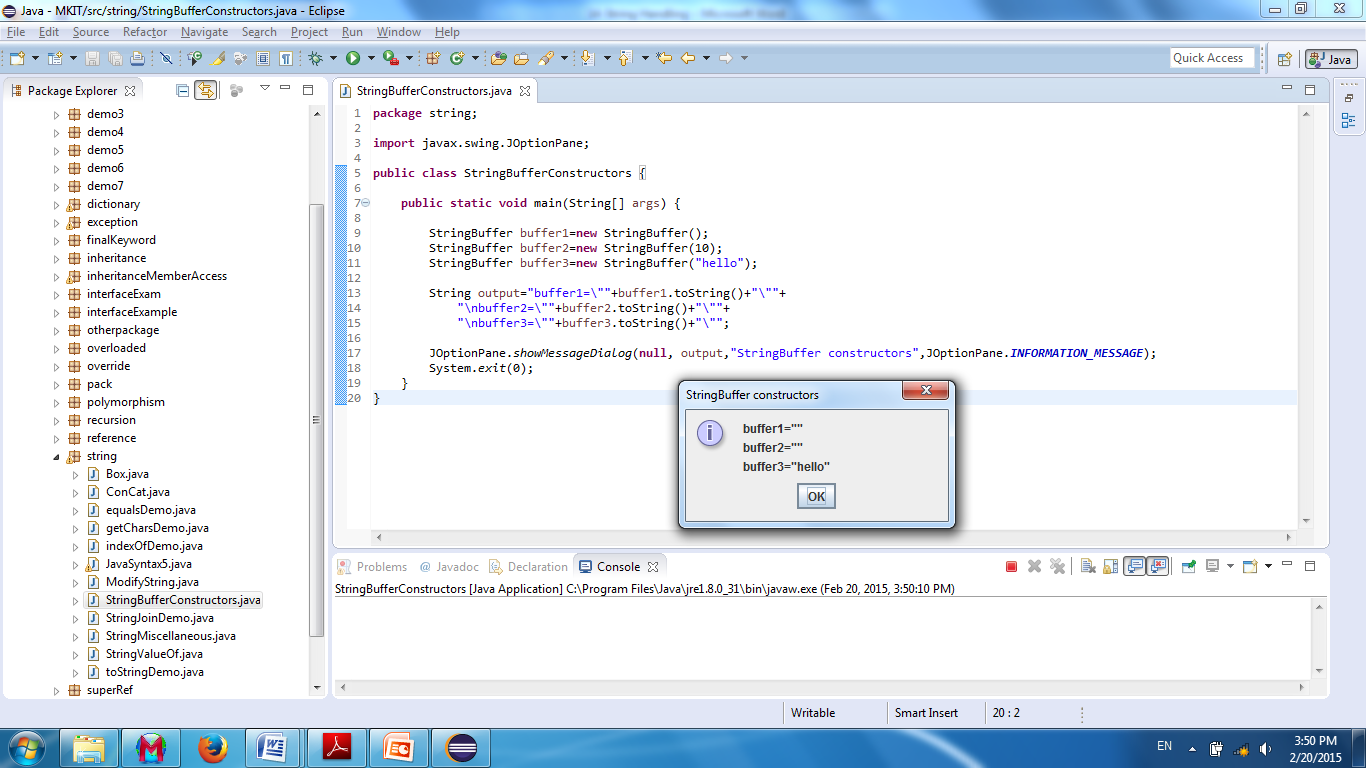
buffer2.toString()+"\""+"\nbuffer3=\""+buffer3.toString()+"\"";

JOptionPane.*showMessageDialog*(**null**, output,"StringBuffer constructors", JOptionPane.***INFORMATION\_MESSAGE***);

System.*exit*(0);

}

}



**import** javax.swing.JOptionPane;

**public** **class** StringBufferCapLen {

**public** **static** **void** main(String[] args) {

StringBuffer buffer=**new** StringBuffer("Hello, how are you?");

String output="buffer="+buffer.toString()+"\nlength="+buffer.length()+

"\ncapacity="+ buffer.capacity();

buffer.ensureCapacity(75);

output+="\n\nNew capacity="+buffer.capacity();

buffer.setLength(10);

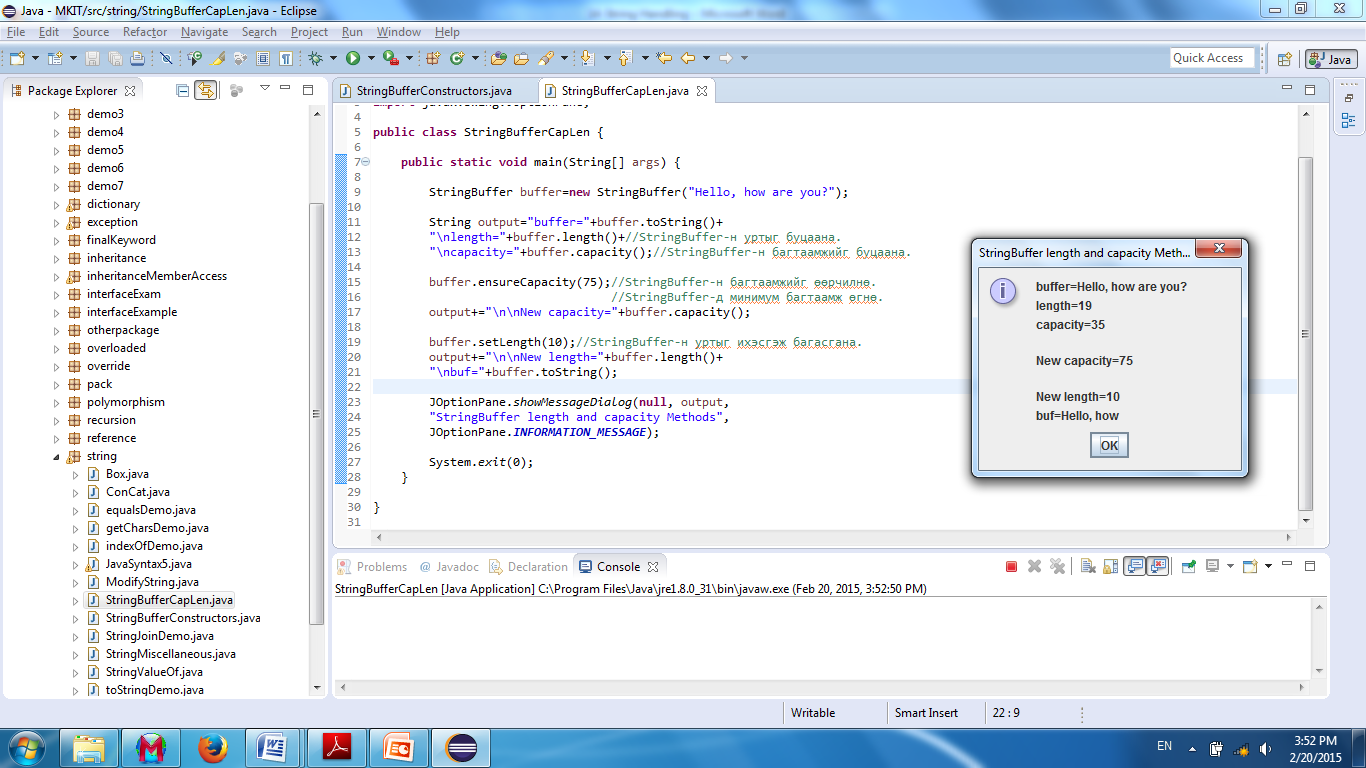
output+="\n\nNew length="+buffer.length()+ "\nbuf="+buffer.toString();

JOptionPane.*showMessageDialog*(**null**, output,"StringBuffer length and capacity Methods",JOptionPane.***INFORMATION\_MESSAGE***);

System.*exit*(0);

}

}



**import** javax.swing.JOptionPane;

**public** **class** StringBufferChars {

**public** **static** **void** main(String[] args) {

StringBuffer buffer=**new** StringBuffer("hello there");

String output="buffer="+buffer.toString()+

"\nCharacter at 0:"+buffer.charAt(0)+

"\nCharacter at 4:"+buffer.charAt(4);

**char** charArray[]=**new** **char**[buffer.length()];

buffer.getChars(0, buffer.length(), charArray, 0);

output+="\n\nThe characters are:";

**for** (**int** count = 0; count < charArray.length; ++count)

output+=charArray[count];

buffer.setCharAt(0, 'H');

buffer.setCharAt(6, 'T');

output+="\n\nbuf="+buffer.toString();

buffer.reverse();

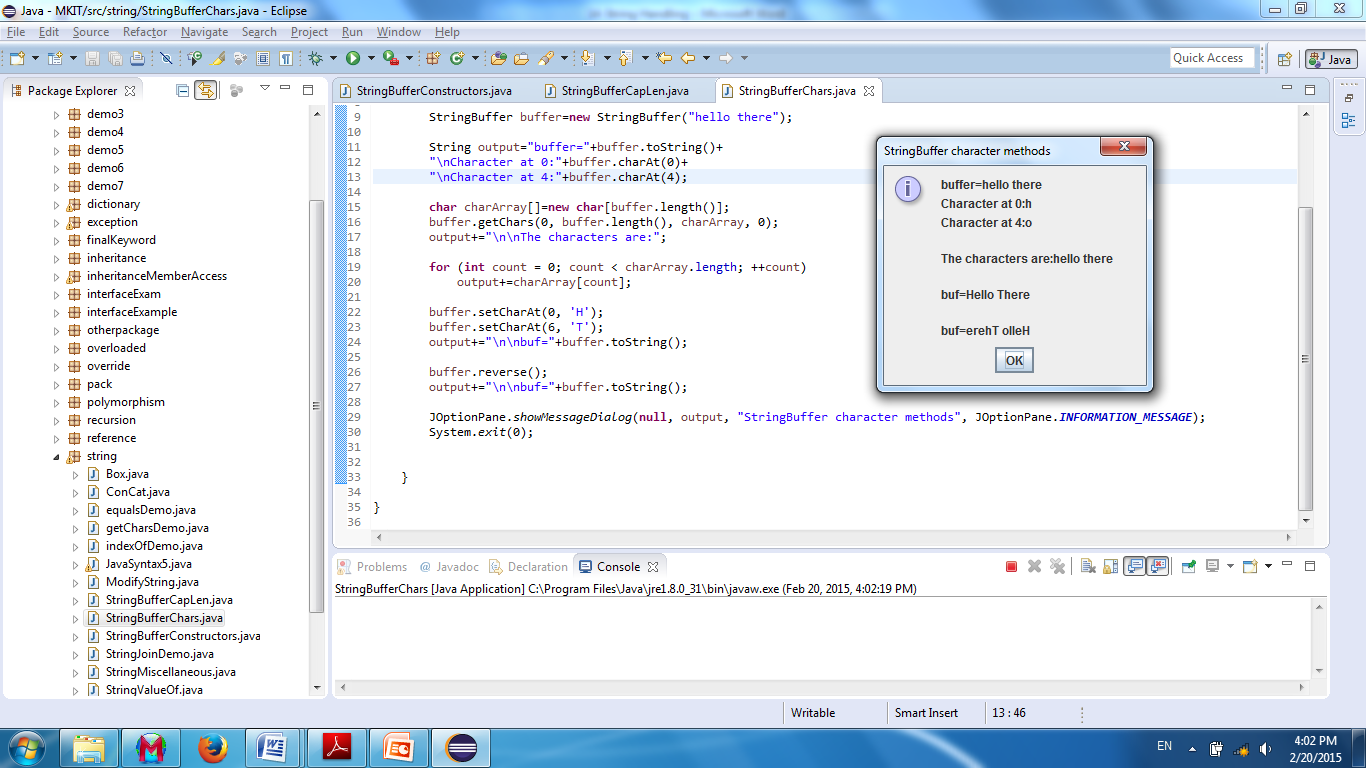
output+="\n\nbuf="+buffer.toString();

JOptionPane.*showMessageDialog*(**null**, output, "StringBuffer character methods", JOptionPane.***INFORMATION\_MESSAGE***);

System.*exit*(0);

}

}



**import** javax.swing.JOptionPane;

**public** **class** StringBufferAppend {

**public** **static** **void** main(String[] args) {

Object objectRef="hello";

String string="goodbye";

**char** charArray[]={'a','b','c','d','e','f'};

**boolean** booleanValue=**true**;

**char** characterValue='Z';

**int** integerValue=7;

**long** longValue=10000000;

**float** floatValue=2.5f;

**double** doubleValue=33.333;

StringBuffer lastBuffer=**new** StringBuffer("last StringBuffer");

StringBuffer buffer=**new** StringBuffer();

buffer.append(objectRef);

buffer.append(" ");

buffer.append(string);

buffer.append(" ");

buffer.append(charArray);

buffer.append(" ");

buffer.append(charArray,0,3);

buffer.append(" ");

buffer.append(booleanValue);

buffer.append(" ");

buffer.append(characterValue);

buffer.append(" ");

buffer.append(integerValue);

buffer.append(" ");

buffer.append(longValue);

buffer.append(" ");

buffer.append(floatValue);

buffer.append(" ");

buffer.append(doubleValue);

buffer.append(" ");

buffer.append(lastBuffer);

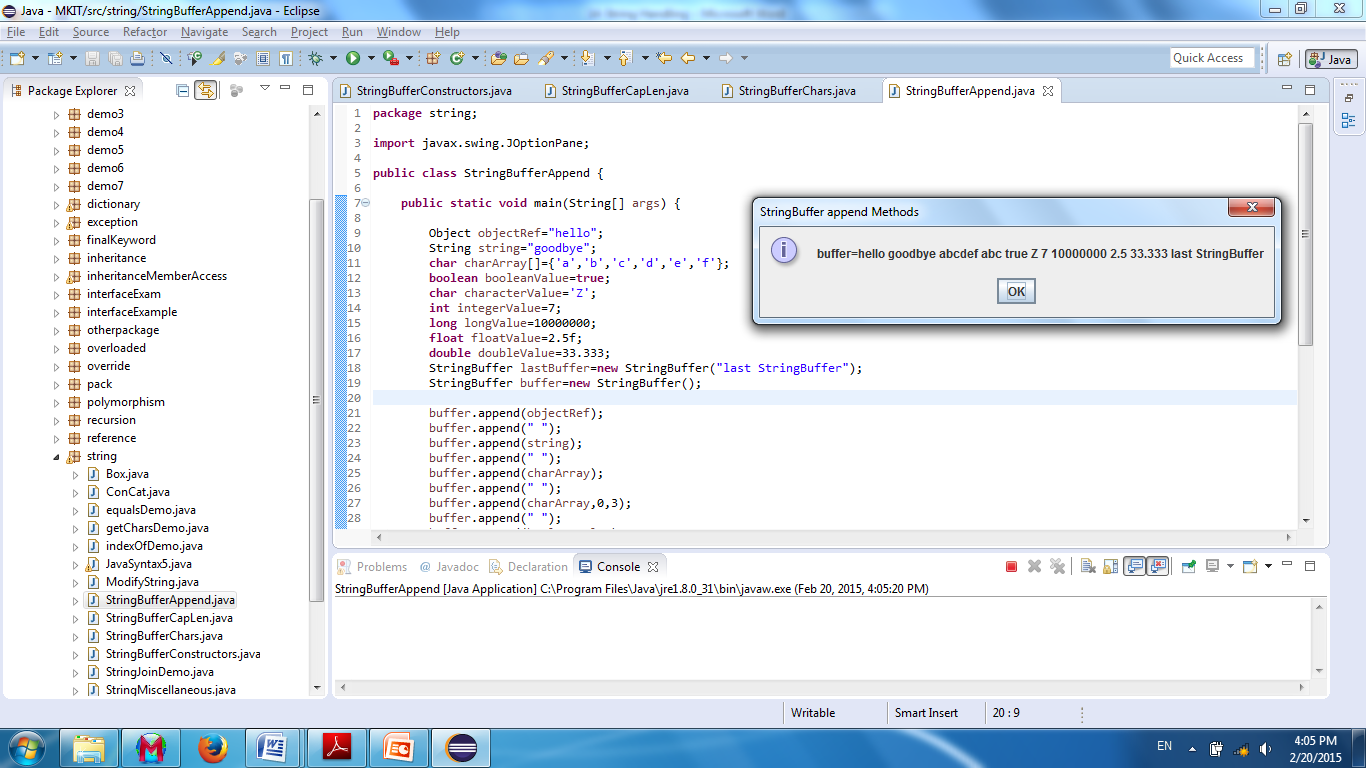
JOptionPane.*showMessageDialog*(**null**, "buffer="+buffer.toString(),

"StringBuffer append Methods", JOptionPane.***INFORMATION\_MESSAGE***);

System.*exit*(0);

}

}



**import** javax.swing.JOptionPane;

**public** **class** StringBufferInsert {

**public** **static** **void** main(String[] args) {

Object objectRef="hello";

String string="goodbye";

**char** charArray[]={'a','b','c','d','e','f'};

**boolean** booleanValue=**true**;

**char** characterValue='K';

**int** integerValue=7;

**long** longValue=10000000;

**float** floatValue=2.5f;

**double** doubleValue=33.333;

StringBuffer buffer=**new** StringBuffer();

buffer.insert(0,objectRef);

buffer.insert(0," ");

buffer.insert(0,string);

buffer.insert(0," ");

buffer.insert(0,charArray);

buffer.insert(0," ");

buffer.insert(0,charArray,3,3);

buffer.insert(0," ");

buffer.insert(0,booleanValue);

buffer.insert(0," ");

buffer.insert(0,characterValue);

buffer.insert(0," ");

buffer.insert(0,integerValue);

buffer.insert(0," ");

buffer.insert(0,longValue);

buffer.insert(0," ");

buffer.insert(0,floatValue);

buffer.insert(0," ");

buffer.insert(0,doubleValue);

String output="buffer after inserts:\n"+buffer.toString();

buffer.deleteCharAt(9);//delete 5 in 2.5

buffer.delete(2,6);//delete .333 in 33.333

output+="\n\nbuffer after deletes:\n"+buffer.toString();

JOptionPane.*showMessageDialog*(**null**, output,

"StringBuffer insert/delete", JOptionPane.***INFORMATION\_MESSAGE***);

System.*exit*(0);

}

}

