* + 1. **Clasa String**

Clasa **String** conţine constructorii **String (), String (String str.), String (char [] c), String (asciichar byte [])**. Aceşti constructori sunt folosiţi pentru a iniţializa obiectele din clasă.

Clasa String contine metode pentru a lucra cu şiruri de caractere:

**concat(String s)** sau **+** – concatenează două siruri;

**equals(Object ob), equalsIgnoreCase(String s)** – compară două şiuri cu şi fara inregistrare;

**compareTo(String s), compareToIgnoreCase (String s)** – compară două siruri cu şi fara inregistrare;

**contentEquals(StringBuffer ob)** – compararea sirului şi a conţinutului unui obiect de tip StringBuffer;

**charAt(int n**)– extraje dintr-un şir de caractere un caracter cu indicele specificat;

**substring(int n, int m**)- extrage dintr-un şir de caractere un subşir cu lungimea m-n, începând de la poziţia n;

**length()** – determina lungimea sirului;

**valueOf(obiect)** – transformarea unui obiect în şir de caractere;

**toUpperCase()/ toLowerCase()** – convertiază toate caracterele din şir în litere majuscule / litere minuscule;

**replace(char с1, char с2)** –înlocueşte toate apariţiile în sir a primului caracter cu al doilea;

**getBytes(obiect), getChars(obiect)** –sirul de caractere e transformat în tablou de octeţi sau caractere;

În următorul exemplu este descrisă o serie de numere întregi care sunt convertite în obiecte String, folosind metodele din această clasă.

/ / Exemplu # 3: utilizarea clasei:

**public** **class** DemoString {

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

**char** s[] = { 'J', 'a', 'v', 'a' };

**int** i = 2;

String str = **new** String(s); //str="Java"

i = str.length(); //i=4

String num = String.valueOf(2); //num="2"

str = str.toUpperCase(); //str="JAVA"

num = str.concat(num); //num="JAVA2"

str = str + "C";//str="JAVAC";

**char** ch = str.charAt(2); //ch='V'

i = str.lastIndexOf('A');

//i=3 (-1 daca lipseste)

num = num.replace('2', 'H'); //num="JAVAH"

i = num.compareTo(str);

//i=5(impreuna cu simbolurile 'H' si 'С')

str.substring(0, 3).toLowerCase(); //JAVA

}

Deoarece obiectul a fost transmis de referinţă, orice schimbare a obiectului în metodă ar trebui să fie transmis obiectului original, deoarece ambele referinţe sunt egale. Acest lucru nu se intamplă deoarece apelînd metoda **concat()**, se crează un nou obiect prin care se face referire. Acest obiect este returnat de către operatorul **return**, dar valoarea returnata nu este atribuită, deci toate modificările sunt pierdute. Dacă codul va fi modificat aşa cum se arată în comentarii, toate modificările obiectului realizate în metoda **changeStr()** vor fi salvate în obiectul declarat în **main().**

În următorul exemplu, se examineză modalităţi de stocare şi identificare a obiectelor utilizînd metoda **equals()**, se compară String-ul cu obiectului specificat a metodei **hashCode()**, care apreciauă codul de distribuire a obiectului.

/\*Exemplu # 4: Utilizaţi equals(), hashCode():\*/

EqualStrings.java \*/

**public** **class** EqualStrings {

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

String s1 = "Java";

String s2 = "Java";

String s3 = **new** String(s1);

System.out.println(s1 + "==" + s2 + " : " + (s1==s2));//true

System.out.println(s1 + "==" + s3 + " : " + (s1==s3));//false

System.out.println(s1 + " equals " + s2 + " : " + s1.equals(s2));//true

System.out.println(s1 + " equals " + s3 + " : " + s1.equals(s3));//true

System.out.println(s1.hashCode());

System.out.println(s2.hashCode());

System.out.println(s3.hashCode());

//toate valorile sunt egale!!!

}}

Metoda **HashCode()** este metoda care returneaza codul de distribuire a obiectului, care este apreciat în conformitate cu principiul - diferite obiecte cu conţinut de acelaşi tip au diferite hash-coduri. Este posibil de a redefini referinţa de la obiect la literal, prin apelul metodei **intern()**.

//Exemplu # 5 : utilizarea metodei intern ()

**public** **class** DemoIntern {

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

String s1 = "Java";

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

System.out.println(s1 == s2); //false

s2 = s2.intern();

System.out.println(s1 == s2); //true

}}

În următorul exemplu are loc sortarea sirurilor de caractere.

/ \* Exemplu # 6: Sortare: \*/

**public** **class** SortArray {

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

String a[] = {" Vika", "Natasha ", " Alina", " Dima ", "Denis "};

**for**(**int** j = 0; j < a.length; j++) a[j].trim();

**for**(**int** j = 0; j < a.length; j++)

**for**(**int** i = j + 1; i < a.length; i++)

**if**(a[i].compareTo(a[j]) < 0) {

String t = a[j];

a[j] = a[i];

a[i] = t;

}

**for**(**int** i = 0; i < a.length; i++)

System.out.print(a[i] + " ");

}}

Metoda **trim()** asigură eliminarea tuturor lăcunilor de la începutul şi sfîrsitul şirului. Metoda **CompareTo()**-efectuază o comparaţie a şirurilor după principiul Unicode.

* + 1. **Clasa StringBuffer**

Clasa **StringBuffer** este asemănătoare cu clasa **String**, dar, spre deosebire, conţinutul şi mărimea obiectelor din clasa **StringBuffer** pot fi schimbate. Constructorul poate primi ca parametru un obiect **String** sau mărimea obiectului. Obiectele acestei clase pot fi transformate în clasa **String** prin metoda **toString()** sau folosind constructorul clasei **String**. Setarea lungimii buferului este efectuată de metoda **setLength(int n)**. Pentru a adăuga simboluri, valori de tipuri standarde, tablouri, şiruri de caractere, folosim metoda **append(parametri)**. Inserarea unui simbol, unui obiect sau a unui şir în poziţia indicată se realizează cu metoda **insert(opţiuni)**.

/\*Exemplu # 7: Crearea unui obiect StringBuffer şi a proprietăţilor sale: \*/

**public** **class** DemoStringBuffer {

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

StringBuffer sb = **new** StringBuffer();

System.out.println("lungimea-"+sb.length());

System.out.println("mărimea"+sb.capacity());

//sb = "Java"

//error, numai pentru clasa String

sb.append("Java");

System.out.println("şirul ->" + sb);

System.out.println("lungimea-"+sb.length());

System.out.println("mărimea"+sb.capacity());

System.out.println("revers-"+sb.reverse());

}}

Rezultatul codul de mai sus este:

Lungimea ->0

Size ->16

String ->JAVA

Lungime ->4

Size ->16

Revers ->avaJ

Folosind metoda **reverse()** se poate schimba rapid ordinea caracterelor in obiect. În cazul în care metoda activată pentru un obiect StringBuffer, produce modificări în conţinutul obiectului, nu se creează un obiect nou, ca în cazul cu obiecte din clasa **String**, dar modifică obiectul curent **StringBuffer**.

/\*Exemplu # 8: Modificarea obiectului StringBuffer: \* /

**public** **class** RefStringBuffer {

**static** **void** changeStr(StringBuffer s) {

s.append(" Microsystems");

}

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

StringBuffer str = **new** StringBuffer("Sun");

changeStr(str);

System.out.println(str);

}}

Rezultatul realizării:

**Sun Microsystems**

Pentru clasa **StringBuffer** nu poate fi realizată suprascrierea a metodelor **equal()** şi **hashCode()**, adică compararea conţinutului a două obiecte nu este posibil, chiar şi codurile hash a tuturor obiectelor se apreciază exact ca şi în clasa Object.

/\*Exemplu # 9: compararea obiectelor StringBuffer şi codurile hash \* /

**public** **class** EqualsStringBuffer {

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

StringBuffer sb1 = **new** StringBuffer("Sun");

StringBuffer sb2 = **new** StringBuffer("Sun"); System.out.println(sb1.equals(sb2));

System.out.println(sb1.hashCode() ==sb2.hashCode());

}}

Rezultatul acestui program va fi :

**false**

**false**

**Lucrare de laborator nr. 1**

**1. Tema lucrării:**

Clase, obiecte, constructori.

**2.**  **Scopul lucrării:**

* Însuşirea modalităţilor de creare a claselor, obiectelor în Java;
* Însuşirea modalităţilor de prelucrare a şirurilor de caractere utilizând clasele de bază;

**3. Etapele de realizare:**

1. Crearea unor clase noi;
2. Crearea şi iniţializarea obiectelor
3. Utilizarea metodelor claselor Sting şi StringBuffer pentru modificarea şirurilor;
4. Crearea interfeţii programului;
5. Prezentarea lucrării.

**4. Exemplu de realizare:**

import java.io.BufferedInputStream;

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStream;

import java.io.InputStreamReader;

public class Main

{

public static void main(String[] args){

String sentence = "";

int wordLenght = 0;

String myWord = "";

InputStreamReader is =

new InputStreamReader(System.in);

BufferedReader bis =

new BufferedReader(is);

try

{

System.out.println("Itrodu propoziţia: ");

sentence = bis.readLine();

System.out.println("Introdu lungimia cuvântului înlocuit");

wordLenght = Integer.parseInt(bis.readLine());

System.out.println("Introdu cuvîntul care trebue înlocuit");

myWord = bis.readLine();

}

catch (IOException e)

{

e.printStackTrace();

}

Text myText = new Text(myWord, sentence, wordLenght);

myText.changeSentence();

System.out.println("Propoziţia nouă" + myText.getSentence());

}}

class Text

{

private String mySentence;

private int charNumber;

private String wordToChange;

private String newSentence = "1.";

public Text(String wordToChange,

String mySentece, int charNumber) {

this.mySentence = mySentece;

this.wordToChange = wordToChange;

this.charNumber = charNumber;

}

public String getSentence()

{

return newSentence;

}

public void changeSentence()

{

int firstPos = 0;

int i;

for (i = 0; i < mySentence.length(); i++)

{

if (mySentence.charAt(i) == ' ')

{

if (i - firstPos == charNumber)

{

newSentence = newSentence.concat(wordToChange+" ");

firstPos = i+1;

}

else

{

newSentence = newSentence.concat

(mySentence.substring(firstPos, i+1));

firstPos=i+1;

}}

else if(i == mySentence.length()-1)

{

if (i - firstPos == charNumber)

{

newSentence = newSentence.concat(wordToChange+" ");

firstPos = i+1;

}

else

{

newSentence = newSentence.concat (mySentence.substring(firstPos, i+1));

firstPos=i+1;

}}}}}

Rezultatul realizării:

Îtrodu propoziţia:

Flori de tei apar în mai.

Introdu lungimea cuvântului înlocuit :

3

Introdu cuvântul spre înlocuire :

măr

Propoziţia noua este : 1. Flori de măr apar în măr.

1. **Probleme propuse:**

Pentru toate sarcinile sa se creeze un Fremes sau o fereastră pentru lucrul cu obiectele.

1. În fiecare cuvânt din textul dat simbolul de pe poziţia indicată a cuvântului de înlocuit cu un simbol dat. Dacă poziţia indicată este mai mare ca lungimea cuvântului, corecţia nu se face.

2. Fiecare litera din textul dat, sa fie înlocuită cu poziţia numerică din alfabet. Textul sa se scrie într-o singură linie cu doua spatii intre litere. In linia următoarea sub fiecare literă sa se indice poziţia.

3. În textul dat, cuvintele cu lungimea indicată, sa se înlocuiască cu un subşir specificat, lungimea căruia poate să nu coincidă cu lungimea cuvântului.

4. În textul dat după fiecare caracter indicat, să se introducă un subşir specificat.

5. După fiecare cuvânt din textul dat, care se termina cu un subşir specificat, sa se adauge cuvântul indicat.

6. În dependenţă de valoarea introdusă (0 sau 1), în textul dat de extras caracterul indicat, ori de câte ori apare, sau sa fie introdus după simbolul cu indicele specificat.

7. Extrageţi din textul dat toate cuvintele cu lungimea specificată, care încep cu o consoană.

8. Afişaţi de câte ori sunt repetate cuvintele care apar în textul dat.

9. Afişaţi, care litere, vocalele sau consoanele, sunt mai multe in fiecare propoziţie din textul dat.

10. Din textul dat găsiţi şi extrageţi numărul de cuvinte care încep şi se finalizează cu vocale.

11. De afişat fără repetare cuvântul din textul dat, care încep si se termina cu aceea literă.

12. De extras din textul dat primul subşir cu lungime maximă, care nu conţine litere.