 Ministerul Educaţiei și

Cercetării al Republicii Moldova

**Universitatea Tehnică a Moldovei**

Facultatea Calculatoare, Informatică și Microelectronică

**RAPORT**

Lucrare de laborator Nr.3

**Tehnologii Avansate de programare**

**Tema:** Supraîncărcarea și suprascrierea metodelor

A realizat st.gr. CR-221 Panfil Ion

A verificat: Petru Cervac

Chișinău-2024

**Obiective:**

Să se realizeze supraîncărcarea şi suprascrierea a trei metode diferite (utilizarea unei clase cu posibilităţile altei clase), creați și suprascrieți o metodă de căutare a unui subșir după un anumit șablon cu **utilizarea expresiilor regulate.**

**Sarcina:**

Sa se creeze obiectul clasa «Paragraf», folosind clasa «Linie».

**Codul:**

**Linie.java**

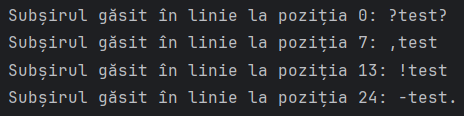
import java.util.regex.Matcher;  
import java.util.regex.Pattern;  
  
public class Linie {  
  
 public void nr\_cuvinte() {  
 System.*out*.println("Numarul de cuvinte a liniei este de 10");  
 }  
  
 public void Aliniere(boolean este\_Aliniere) {  
 if (este\_Aliniere) {  
 System.*out*.println("Linia Este Aliniată");  
 } else {  
 System.*out*.println("Linia Nu Este Aliniată");  
 }  
 }  
  
 // Prima versiune a metodei cautaSubstring (supraîncărcare)  
 public void cautaSubstring(String sablon) {  
 System.*out*.println("Căutare subșir în linie folosind doar șablonul: " + sablon);  
 }  
  
 // A doua versiune a metodei cautaSubstring (supraîncărcare)  
 public void cautaSubstring(String text, String sablon) {  
 Pattern pattern = Pattern.*compile*("([\\p{Punct}]\*)\\b" + sablon + "\\b([\\p{Punct}]\*)");  
 Matcher matcher = pattern.matcher(text);  
  
 while (matcher.find()) {  
 System.*out*.println("Subșirul găsit în linie la poziția " + matcher.start() + ": " + matcher.group());  
 }  
 }  
}

**Paragraf.java**

import java.util.regex.Matcher;  
import java.util.regex.Pattern;  
  
public class Paragraf extends Linie {  
  
 @Override  
 public void cautaSubstring(String text, String sablon) {  
 Pattern pattern = Pattern.*compile*("([\\p{Punct}]\*)\\b" + sablon + "\\b([\\p{Punct}]\*)");  
 Matcher matcher = pattern.matcher(text);  
  
 while (matcher.find()) {  
 System.*out*.println("Subșirul găsit în linie la poziția " + matcher.start() + ": " + matcher.group());  
 }  
 }  
  
 public void Pagina(String numarPagina) {  
 System.*out*.println("Pagina: " + numarPagina);  
 }  
}

**Main.java**public class Main {  
 public static void main(String[] args) {  
 Linie l = new Paragraf();  
 l.nr\_cuvinte();  
 l.Aliniere(true);  
 System.*out*.println();  
  
 Paragraf p = new Paragraf();  
 p.nr\_cuvinte();  
 p.Aliniere(true);  
 p.Pagina("14");  
 System.*out*.println();  
  
 l.cautaSubstring("Acesta este un text de test. Testăm căutarea de subșiruri.");  
 l.cautaSubstring("?test? ,test !test alpha-test. tester atest", "test");  
 }  
}

**Rezultatul compilării:**

****

**Concluzie**

Clasa Linie definește metode pentru a număra cuvintele dintr-o linie și pentru a verifica dacă linia este aliniată. De asemenea, oferă două versiuni ale metodei cautaSubstring, care permit căutarea unui subșir într-un text dat, fie folosind doar șablonul subșirului, fie specificând atât textul cât și șablonul.

Clasa Paragraf extinde clasa Linie și suprascrie metoda cautaSubstring pentru a implementa un comportament specific, cum ar fi ignorarea cazurilor în care subșirul căutat face parte dintr-un alt cuvânt. Această suprascriere demonstrează polimorfismul, unde metoda din clasa de bază este înlocuită cu o implementare specifică în clasa derivată.

Clasa Main demonstrează utilizarea acestor clase prin crearea de obiecte și apelarea metodelor lor pentru a efectua operații de căutare de subșiruri în diferite texte.