Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Основи об'єктно-орієнтованого програмування

Лабораторна робота №4

«Класи в мові програмування Java»

Виконала:

студентка групи ІВ-71

Молчанова В. С.

Залікова книжка № ІВ-7110

Перевірив Подрубайло О. О.

Київ

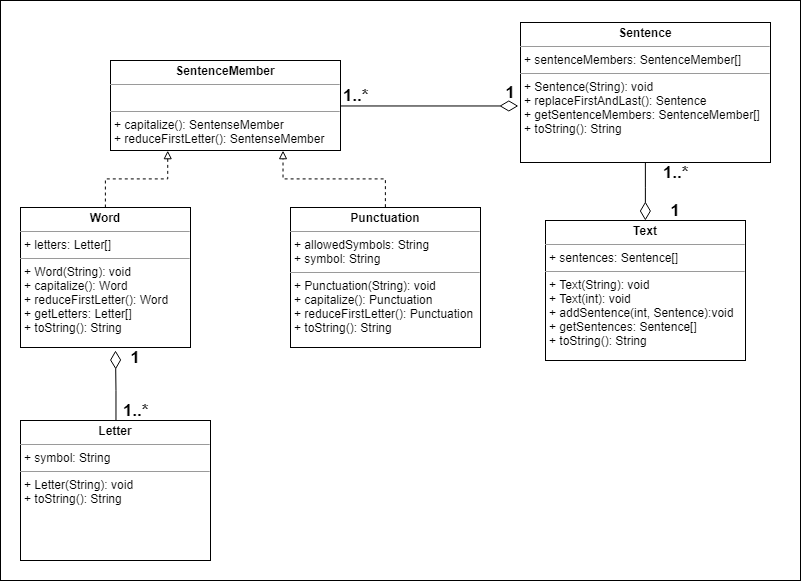
2018 р.

# Власний варіант:

С17 = 7110 % 17 = 4

|  |  |  |
| --- | --- | --- |
| 4 | Дія з рядком | В кожному реченні заданого тексту змінити місцями перше та останнє слово, не змінивши довжини речення. |

# Діаграма класів:



# Код програми:

**public class** Main {  
  
 **public static void** main(String[] args) {  
 Text text1 = **new** Text(**"Hello, everyone! Thanks for still reading this. Really."**);  
 System.***out***.println(text1);  
  
 Sentence[] temp = text1.getSentences();  
 Text text2 = **new** Text(temp.**length**);  
 **for**(**int** i = 0; i < temp.**length**; i++){  
 text2.addSentence(i, temp[i].swapFirstAndLast());  
 }  
 System.***out***.println(text2);  
 }  
}

**public class** Text {  
 **private** Sentence[] **sentences**;  
  
 **public** Text(String text) {  
 text = text.replaceAll(**"[ \\t]{2,}"**, **" "**);  
 String[] split = text.split(**"(?<=[?!.]\\s)"**);  
 **sentences** = **new** Sentence[split.**length**];  
 **for** (**int** i = 0; i < split.**length**; i++) {  
 **sentences**[i] = **new** Sentence(split[i]);  
 }  
 }  
  
 **public** Text(**int** size){  
 **sentences** = **new** Sentence[size];  
 }  
  
 **public void** addSentence(**int** position, Sentence sentence){  
 **sentences**[position] = sentence;  
 }  
  
 **public** Sentence[] getSentences(){  
 **return sentences**;  
 }  
  
 @Override  
 **public** String toString(){  
 StringBuilder text = **new** StringBuilder();  
 **for**(Sentence sentence:**sentences**){  
 text.append(sentence+**" "**);  
 }  
 **return** text.toString();  
 }  
}

**public class** Sentence {  
 **private** SentenceMember[] **sentenceMembers**;  
  
 **public** Sentence(String s) {  
 String[] split = s.split(**"(?=,|\\.|!|\\?)|\\s"**);  
 **sentenceMembers** = **new** SentenceMember[split.**length**];  
 **for** (**int** i = 0; i < split.**length**; i++) {  
 **if** (Punctuation.*allowedSymbols*.contains(split[i])) {  
 **sentenceMembers**[i] = **new** Punctuation(split[i]);  
 } **else** {  
 **sentenceMembers**[i] = **new** Word(split[i]);  
 }  
 }  
 }  
  
 **public** Sentence swapFirstAndLast(){  
 **if**(**sentenceMembers**.**length** > 2){  
 Sentence result = **new** Sentence(**this**.toString());  
 result.**sentenceMembers**[0] = **sentenceMembers**[**sentenceMembers**.**length** - 2].capitalize();  
 result.**sentenceMembers**[**sentenceMembers**.**length** - 2] = **sentenceMembers**[0].reduceFirstLetter();  
 **return** result;  
 }  
 **return this**;  
 }  
  
 **public** SentenceMember[] getSentenceMembers() {  
 **return sentenceMembers**;  
 }  
  
 @Override  
 **public** String toString(){  
 StringBuilder sentence = **new** StringBuilder();  
 **for**(**int** i = 0; i < **sentenceMembers**.**length** - 1; i++){  
 sentence.append(**sentenceMembers**[i]);  
 **if**(!(**sentenceMembers**[i+1] **instanceof** Punctuation)){  
 sentence.append(**" "**);  
 }  
 }  
 sentence.append(**sentenceMembers**[**sentenceMembers**.**length** - 1]);  
 **return** sentence.toString();  
 }  
}

**public abstract class** SentenceMember {  
 **public abstract** SentenceMember capitalize();  
  
 **public abstract** SentenceMember reduceFirstLetter();  
}

**public class** Punctuation **extends** SentenceMember{  
 **public static** String *allowedSymbols* = **" ,.!?-;:"**;  
 **private** String **symbol**;  
  
 **public** Punctuation(String symbol){  
 **this**.**symbol** = symbol;  
 }  
  
 **public** Punctuation capitalize(){  
 **return this**;  
 }  
  
 **public** Punctuation reduceFirstLetter(){  
 **return this**;  
 }  
  
 @Override  
 **public** String toString(){  
 **return symbol**;  
 }  
}

**public class** Word **extends** SentenceMember{  
 **private** Letter[] **letters**;  
  
 **public** Word(String word){  
 **letters** = **new** Letter[word.length()];  
 **for**(**int** i = 0; i < word.length(); i++){  
 **letters**[i] = **new** Letter(word.substring(i, i+1));  
 }  
 }  
  
 **public** Word capitalize(){  
 StringBuilder s = **new** StringBuilder(**letters**[0].toString().toUpperCase());  
 **for**(**int** i = 1; i < **letters**.**length**; i++){  
 s.append(**letters**[i].toString());  
 }  
 **return new** Word(s.toString());  
 }  
  
 **public** Word reduceFirstLetter(){  
 StringBuilder s = **new** StringBuilder(**letters**[0].toString().toLowerCase());  
 **for**(**int** i = 1; i < **letters**.**length**; i++){  
 s.append(**letters**[i].toString());  
 }  
 **return new** Word(s.toString());  
 }  
  
 **public** Letter[] getLetters(){  
 **return letters**;  
 }  
  
 @Override  
 **public** String toString(){  
 StringBuilder word = **new** StringBuilder();  
 **for**(Letter letter: **letters**){  
 word.append(letter);  
 }  
 **return** word.toString();  
 }  
}

**public class** Letter {  
 **private** String **symbol**;  
  
 **public** Letter(String symbol){  
 **this**.**symbol** = symbol;  
 }  
  
 @Override  
 **public** String toString(){  
 **return symbol**;  
 }  
}

# Висновок

На мою думку, в контексті цієї роботи було доречно не розглядати пробіли, як окремі розділові знаки, а просто додавати їх у потрібних місцях вже при виводі. Також я вирішила, що програма буде більш універсальною, якщо клас Sentence буде мати метод, який не змінює вихідне речення, а повертає нове. Тому я додала можливості створювати пустий текст із заданою кількістю речень та змінювати конкретне речення в тексті.

До того ж я вирішила покращити роботу алгоритма з попередньої роботи, і тепер при заміні слів, перше слово в реченні пишеться з великої літери, а останнє – з маленької.