****

Latvijas Universitātes Vadības un uzņēmējdarbības mācību centrs

(LU VUMC)

**IEVADS JAVA PROGRAMMĒŠANĀ**

**NODARBĪBA #9**

Mārtiņš Ceske/Vjačeslavs Vasiļevskis

2021

ESF projekts Nr. 8.4.1.0/16/l/001  
‘’Nodarbināto personu profesionālās kompetences pilnveide”



# 1. mājas darbs

# Vārdu skaitītājs

### **Noteikumi:**

* Lietotājs standarta ievadā ievada rindas ar vārdiem
* Programmai ir jānolasa vārdi un ir jāizdrukā ievadīto vārdu biežumus.

### **Soļi:**

* Izveidojam jaunu klasi WordCounter ar metodi “main”
* No “main” metodes izsaucam run metodi, kurā ievietojam doto kodu. Run metode lasa no termināļa ievadītos vārdus

// . . .  
   
 Map<String,Integer> wordCounts = new HashMap<>();  
   
 public void run(){  
 Scanner input=new Scanner(System.in);  
 while(true){  
 System.out.println("Please enter line of words or type quit:");  
 String line=input.nextLine();  
 if(line.contains("quit"))break;  
 for(String word:line.split(" ")){  
 addWord(word,wordCounts);  
 }  
 }  
   
 for(String word:wordCounts.keySet()){  
 System.out.println(word+" = "+wordCounts.get(word));  
 }  
   
 input.close();  
 }  
 // . . .

* Ievadīto rindu sadalām vārdos “for” ciklā un izsaucam “addWord” metodi, kura ievietos vārdu iekš Map kolekcijas.
* Uzdevums ir implementēt “ddWords” metodi, kura ievadītos vārdus saglabās Map kolekcijā. Atslēga = vārds, vērtība = biežums cik reizess vārds tika nolasīts no ievada.
* Lūdzu implementējiet addWord metodi. Metode saņem vārdu un Map kolekciju. Ja vārds jau atrodas kolekcijā, vērtību palielinām par 1, pretējā gadījumā pievienojam jaunu elementu. Atslēga = vārds, vērtība = 1.

public static void addWord(String word, Map<String, Integer> wordMap){  
 // ...   
 }

**Alternatīva datu ievadei**

* Ja lasīšana no termināļa mulsina, tad ir iespējams lietot citu run() metodes versiju. Šai versijā vārdus ievada kā konstantes. Izpētiet kā izveido masīvu un to pārveido par List (sarakstu):

void run(){  
  
 String[] wordArray = new String[]  
 {"Zoologija",  
 "Matematika",  
 "Zoologija",  
 "Literatura",  
 "Matematika"};  
   
 List<String> wordList=Arrays.asList(wordArray);  
  
 for(String word:wordList){  
 addWord(word,wordMap);  
 }  
 System.out.println(wordMap);  
 }

# 2. mājas darbs

# Sakārtots vārdu skaitītājs

### 

### **Noteikumi:**

* Lietotājs standarta ievadā ievada rindas ar vārdiem
* Programmai ir jānolasa vārdi un ir jāizdrukā ievadīto vārdu biežumus augošā secībā.

P.S. Par pamatu izmanotjiet kodu no ierpiekšējā uzdevumu

### **Soļi:**

* Pievienojam metodi, kurai nododam Map kolekciju. Metode atdos pēc atslēgas sakārtotu sarakstu augošā secībā.

List<String> sortByKey(Map<String,Integer> map){  
 // ...  
 }

* Izveidojam ArrayList (List kolekciju) no atslēgu Set kolekcijas (Map.keySet()) to nododot konstruktorā.

**Piemērs**, kā Set pārveidot par List

List<String> listOfKeys = new ArrayList<>(theMap.keySet());

* Sakārtojam sarakstu.

P.S.To var izdarīt ar Collections.sort() vai Arrays.sort()

Collections.sort(listOfKeys);

* Metode atgriež sortēto List (sarakstu).
* Metodē “run” pārstaigājiet Map kolekciju atslēgas, lasot no sakārtotā saraksta:

for(String key: sortKeys(wordMap)){  
 System.out.println("["+key+"="+wordMap.get(key)+"]");  
}