https://github.com/drtrainig/cap8

Aici găsiți și exemple de la curs, link-uri utile, sugestii..

https://docs.oracle.com/javase/8/docs/api/java/util/Map.html

https://docs.oracle.com/javase/8/docs/api/java/util/Set.html

https://docs.oracle.com/javase/7/docs/api/java/util/Queue.html

https://docs.oracle.com/javase/7/docs/api/java/util/Stack.html

Aveti si documentatie aici.

https://www.geeksforgeeks.org/set-in-java/

https://www.geeksforgeeks.org/queue-interface-java/

https://www.geeksforgeeks.org/stack-class-in-java/

https://www.javatpoint.com/java-map (O sa vedeti si o alta modalitate de a parcurge un Map).

Exemple problem rezolvate:

MAP:

package com.itfactory.map;

import com.itfactory.Persoana;

import java.util.\*;

public class MapEx {

public static void main(String[] args) {

Map<Integer, String> stringMap = new HashMap<>();

stringMap.put(21231, "Dragos");

// Map<Integer, Persoana> persoanaMap = new HashMap<>();

// persoanaMap.put(123141, new Persoana("Roban", "Dragos"));

//

// Persoana persoana = persoanaMap.get(123141);

//

// if (persoana != null) {

// System.out.println(persoana);

// }

// for (Map.Entry<Integer, String> entry : stringMap.entrySet()) {

// System.out.println(entry.getKey() + " : " + entry.getValue());

// }

// System.out.println(stringMap.get(5));

// if (stringMap.containsKey(1)) {

// System.out.println(stringMap.get(1));

// }

//

//

// Map<Integer, String> stringMapT = new HashMap<>();

// stringMapT.put(1, "Dragos Roban");

// stringMapT.put(3, "Dragos Roban");

// stringMapT.put(2, "Dragos Roban");

//

// for (Map.Entry<Integer, String> entry : stringMapT.entrySet()) {

// System.out.println(entry.getKey() + " : " + entry.getValue());

// }

}

}

QUEUE:

|  |
| --- |
| package com.itfactory.queue; |
|  |  |
|  | import java.util.LinkedList; |
|  | import java.util.Queue; |
|  |  |
|  | public class QueueEx { |
|  | public static void main(String[] args) { |
|  | Queue<String> stringQueue = new LinkedList<>(); |
|  | stringQueue.add("mar"); |
|  | stringQueue.add("par"); |
|  | stringQueue.add("cires"); |
|  |  |
|  | while (!stringQueue.isEmpty()) { |
|  | String elementDinCoada = stringQueue.poll(); |
|  | System.out.println(elementDinCoada); |
|  | } |
|  | } |
|  | } |

SET:

|  |
| --- |
| package com.itfactory.set; |
|  |  |
|  | import com.itfactory.Persoana; |
|  |  |
|  | import java.util.HashSet; |
|  | import java.util.Set; |
|  |  |
|  | public class SetEx { |
|  | public static void main(String[] args) { |
|  |  |
|  | Set<Persoana> persoanaSet = new HashSet<>(); |
|  | persoanaSet.add(new Persoana("Dragos", "Roban")); |
|  | persoanaSet.add(new Persoana("Dragos", "Roban")); |
|  | persoanaSet.add(new Persoana("Dragos", "Roban")); |
|  | persoanaSet.add(new Persoana("Dragos", "Roban")); |
|  | for (Persoana persoana : persoanaSet) { |
|  | System.out.println(persoana); |
|  | } |
|  | } |
|  | } |

STACK:

1.

|  |
| --- |
| package com.itfactory.stack; |
|  |  |
|  | import java.util.Stack; |
|  |  |
|  | public class StackEx { |
|  | public static void main(String[] args) { |
|  | Stack<String> stringStack = new Stack<>(); |
|  | stringStack.push("mar"); |
|  | stringStack.push("par"); |
|  | stringStack.push("cires"); |
|  |  |
|  | while (!stringStack.isEmpty()) { |
|  | String valoare = stringStack.pop(); |
|  | } |
|  | } |
|  | } |

2.

|  |
| --- |
| package com.itfactory.stack; |
|  |  |
|  | public class Utils { |
|  | public static String[] convertNumberInStringArray(int number) { |
|  |  |
|  | char[] array = String.valueOf(number).toCharArray(); |
|  | String[] strArray = new String[array.length]; |
|  | for (int i = 0; i < array.length; i++) { |
|  | strArray[i] = String.valueOf(array[i]); |
|  | } |
|  | return strArray; |
|  | } |
|  |  |
|  | //Exemplu de transormare |
|  | public static void main(String[] args) { |
|  | String[] arr = Utils.convertNumberInStringArray(1234); |
|  | for (String s : arr) { |
|  | System.out.println(s); |
|  | } |
|  | } |
|  | } |

EXEMPLU CU CLASE:

|  |
| --- |
| package com.itfactory; |
|  |  |
|  | import com.itfactory.stack.Utils; |
|  |  |
|  | import java.util.Objects; |
|  |  |
|  | public class Persoana { |
|  | String nume; |
|  | String prenume; |
|  |  |
|  |  |
|  |  |
|  | public Persoana(String nume, String prenume) { |
|  | this.nume = nume; |
|  | this.prenume = prenume; |
|  | } |
|  |  |
|  | public String getNume() { |
|  | return nume; |
|  | } |
|  |  |
|  | public String getPrenume() { |
|  | return prenume; |
|  | } |
|  |  |
|  | @Override |
|  | public boolean equals(Object o) { |
|  | if (this == o) return true; |
|  | if (o == null || getClass() != o.getClass()) return false; |
|  | Persoana persoana = (Persoana) o; |
|  | return Objects.equals(nume, persoana.nume) && |
|  | Objects.equals(prenume, persoana.prenume); |
|  | } |
|  |  |
|  | @Override |
|  | public int hashCode() { |
|  | return Objects.hash(nume, prenume); |
|  | } |
|  |  |
|  | @Override |
|  | public String toString() { |
|  | return "Persoana{" + |
|  | "nume='" + nume + '\'' + |
|  | ", prenume='" + prenume + '\'' + |
|  | '}'; |
|  | } |
|  | } |

SUGESTII TEMA:

Problema1.

O sa avem nevoie de un Set pentru problema asta deoarece o sa evitam sa adaugam duplicate astfel sa vedem cate cuvinte distincte avem in text.

Array-ul rezultat de la split il vom parcurge si vom insera in set fiecare string. Mai apoi afisam lungimea setului.

Problema2.

Pentru aceasta verificare va fi util sa folositi un Queue si un Stack.

Prima data puneti fiecare cifra (parcurgand array-ul) in Queue si in Stack (in acelasi timp).

Apoi parcurgeti Queue-ul si Stack-ul si fiecare element scos din Queue trebuie sa fie egal cu fiecare element scos din Stack.

Daca 2 elemente scoase nu sunt egale, numarul nu este palindrom.

Exemplul palindrom

1 2 3 4 3 2 1 (citit normal)

1 2 3 4 3 2 1 (citit invers)

Exemplul care nu este palindrom

1 2 3 4 5 6 7 (citit normal)

7 6 5 4 3 2 1 (citit invers)

Problema 3.

Pentru Map se va folosi implementarea HashMap<>();

Se vor crea 3 obiecte de tip Persoana si un obiect de tip Departament.

Se vor introduce cele 3 persoane cu acelasi departament in obiectul de tip Map creat

Verificarea daca exista persoana se face cu containsKey()

Verificarea daca persoana are ca valoare departametnul introdus:

1.se va folosi get pentru a obtine Deparamentul

2.se va verifica daca departamentul obtinut din map este acelasi cu cel citit.

TEME:

1. Se da un text citit de la tastatura (sau dintr-o variabila String). Pentru delimitarea cuvintelor se vor folosi doar spatii.

Se cere sa se determine cate cuvinte distincte sunt in textul dat.

Exemplu de text. "mar mar par par cireasa". Sunt 3 cuvinte distincte in acest text. (mar, par, cireasa).

Pentru toata lumea: Pentru a sparge un text in cuvinte ne folosim de metoda split din String. metoda split ne va returna un array de string-uri. (vezi documentatie).

String[] cuvinte = textulMeu.split("\\s+") (O sa spargem textul in cuvinte in functie de spatiile dintre cuvinte)

https://www.geeksforgeeks.org/split-string-java-examples/

<https://docs.oracle.com/javase/7/docs/api/java/lang/String.html#split(java.lang.String)>

SUGESTII TEMA:

Problema1.

O sa avem nevoie de un Set pentru problema asta deoarece o sa evitam sa adaugam duplicate astfel sa vedem cate cuvinte distincte avem in text.

Array-ul rezultat de la split il vom parcurge si vom insera in set fiecare string. Mai apoi afisam lungimea setului.

2. Sa se verifice daca un numar citit de la tastatura este palindrom sau nu.

Exemplu: 991199 este palindrom, deoarece numarul citit invers este ca cel initial. Alte exemple de palindrom: 121, 22322, 556655, 4432344

In clasa Utils aveti o metoda ajutatoarea care va transofmra un numar de tip int in array de String continand fiecare cifra in ordine. Puteti rula metoda main din clasa Utils sa va convingeti.

Sugestie

Problema2.

Pentru aceasta verificare va fi util sa folositi un Queue si un Stack.

Prima data puneti fiecare cifra (parcurgand array-ul) in Queue si in Stack (in acelasi timp).

Apoi parcurgeti Queue-ul si Stack-ul si fiecare element scos din Queue trebuie sa fie egal cu fiecare element scos din Stack.

Daca 2 elemente scoase nu sunt egale, numarul nu este palindrom.

Exemplul palindrom

1 2 3 4 3 2 1 (citit normal)

1 2 3 4 3 2 1 (citit invers)

Exemplul care nu este palindrom

1 2 3 4 5 6 7 (citit normal)

7 6 5 4 3 2 1 (citit invers)

3. Sa se creeze doua clase in Java si anume clasa Persoana cu fieldurile [id, nume] si clasa Departament cu fieldurile [numeDepartament]. (Atentie la metodele pe care trebuie sa le generati/suprascrieti in clase)

Intr-o clasa Main cu metoda main se va crea un map in care cheile (key) vor fi de tip Persoana iar valorile (value) vor fi de tip Departament.

Se vor introduce initial (nu trebuie citit de la tastatura neaparat) un numar de persoane din acelasi departament. (Minim 3 persoane).

Mai departe vom simula schimbarea departamentului unei persoane.

Se cere sa se introduca de la tastatura id-ul si numele unei persoane. Daca Id-ul si numele coincid cu una din Persoanele introduse, se cere mai apoi sa se introduca numele departamentului nou in care o sa fie persoana respectiva.

Daca departamentul deja exista asignat persoanei deja, se va afisa un mesaj de eroare si programul se va incheia. Daca deparamentul nu este asignat persoanei respective, se va crea departamentul si se va asigna persoanei respective.

Probleme Optionale:

1. Se da o lista de String-uri (Sau Integer, sau obiecte preferabile). Printr-o linie de cod, transformati lista intr-un set.

Atentie: In lista se vor regasi elemente duplicate, in Set acele elemente duplicate vor aparea o singura data.

2. Se da un map unde cheile sunt de tip int (atentie la int) iar valorile sunt de tip lista de int (atentie la int).

Fiecare element va fi o pereche de forma (int, list de int).

Se cere ca pentru fiecare element din map sa se afiseze cheia, si maximul din lista asociata.

3. Se dau doua structuri de date: un queue si un stack amblee continand numere intregi.

Se cere sa se implementeze un program prin care la fiecare extragere a unui numar din stack, atatea elemente vor fi scoase din queue.

Atunci cand nu mai sunt elemente ramase intr-una din structuri se va afisa care din ele a ramas empty. :)