Sets and Maps

Java Collections API – Sets and Maps

Advanced Java

SoftUni Team

Technical Trainers

Table of Contents

SoftUni
Foundation

sli.do

#java-advanced

Sets

HashSet<E>, TreeSet<E> and LinkedHashSet<E>

Sets in Java

- A set keeps unique elements
- Provides methods for **adding** / **removing** / **searching** elements
- Offers very fast performance
- Types:
    - **HashSet<E>**
        - The elements are randomly ordered
    - **TreeSet<E>**
        - The elements are ordered incrementally
    - **LinkedHashSet<E>**
        - The order of appearance is preserved

Methods

- Initialization:

```
HashSet<String> hash = new HashSet<String>();
```
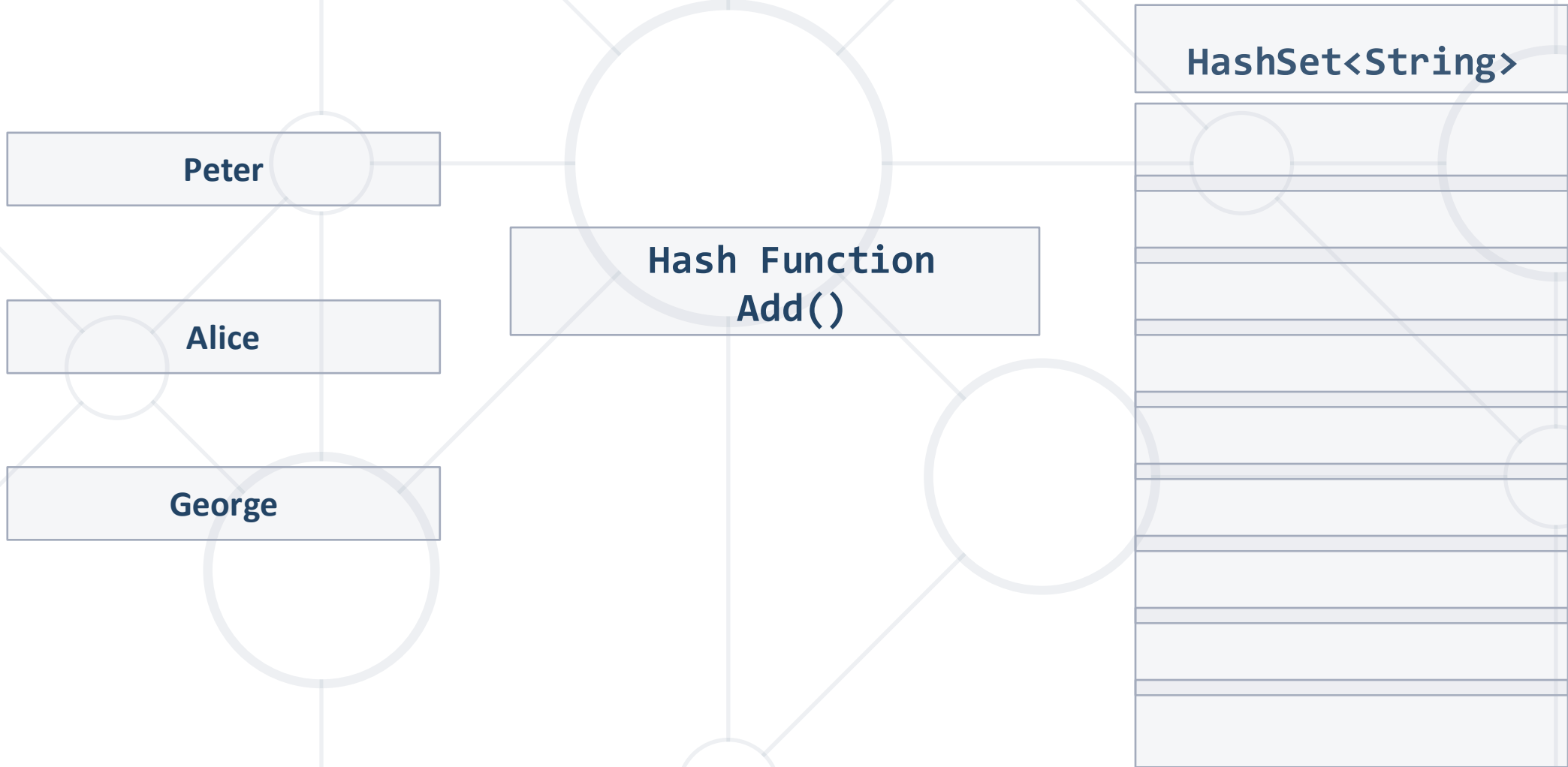
- For easy reading you can use diamond inference syntax:

```
TreeSet<String> tree = new TreeSet<>();
```
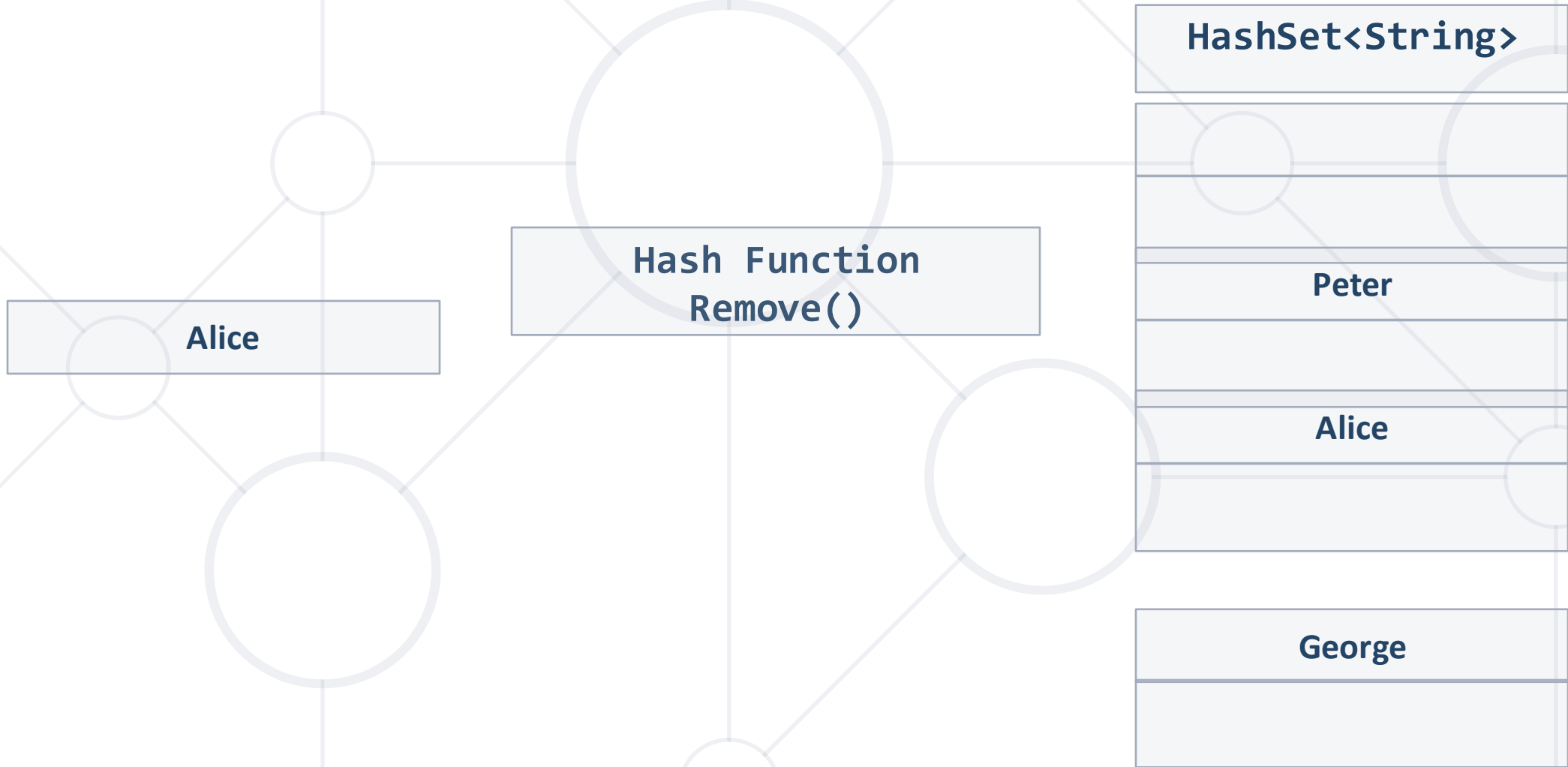
- `.size()`

- `.isEmpty()`

```
HashSet<String> hash = new HashSet<>();
System.out.println(hash.size());      // 0
System.out.println(hash.isEmpty());  // True
```

HashSet<E> – add()

Peter

Alice

George

Hash Function
Add()

HashSet<String>

HashSet<E> — remove()

SoftUni Foundation

HashSet<String>

Peter

Alice

George

Hash Function
Remove()

Alice

TreeSet<E> – add()

Peter

Alice

George

Hash Function
Add()

TreeSet<String>

LinkedHashSet<E> – add()

Ivan

Alice

Georgi
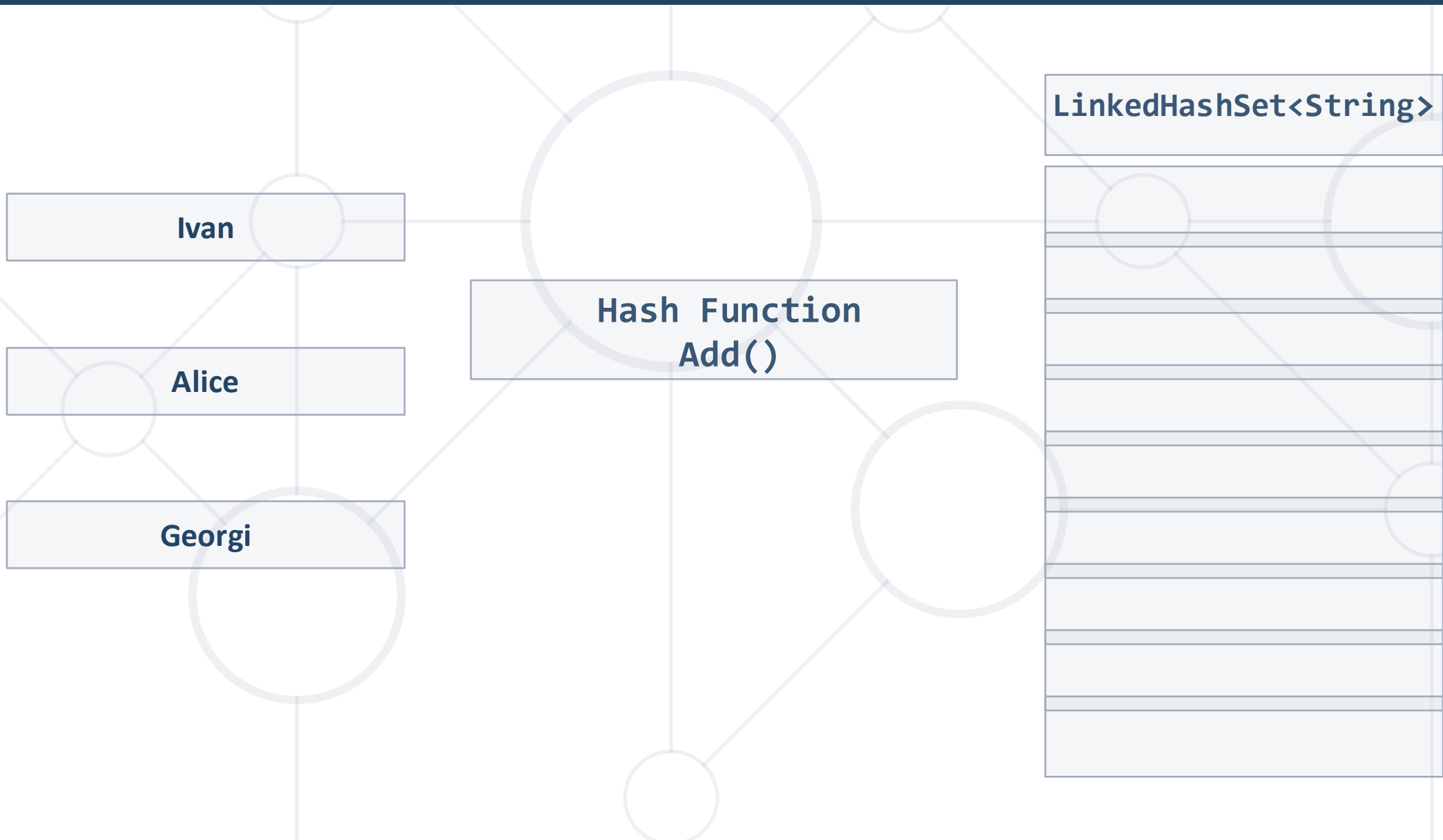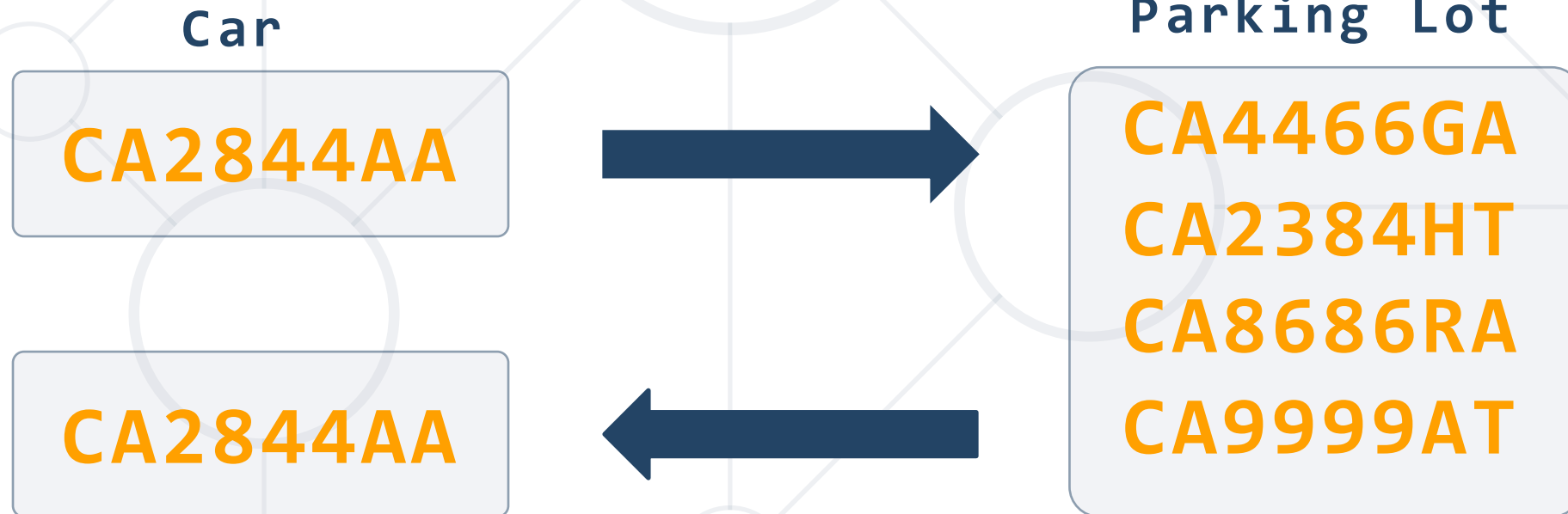
Hash Function
Add()
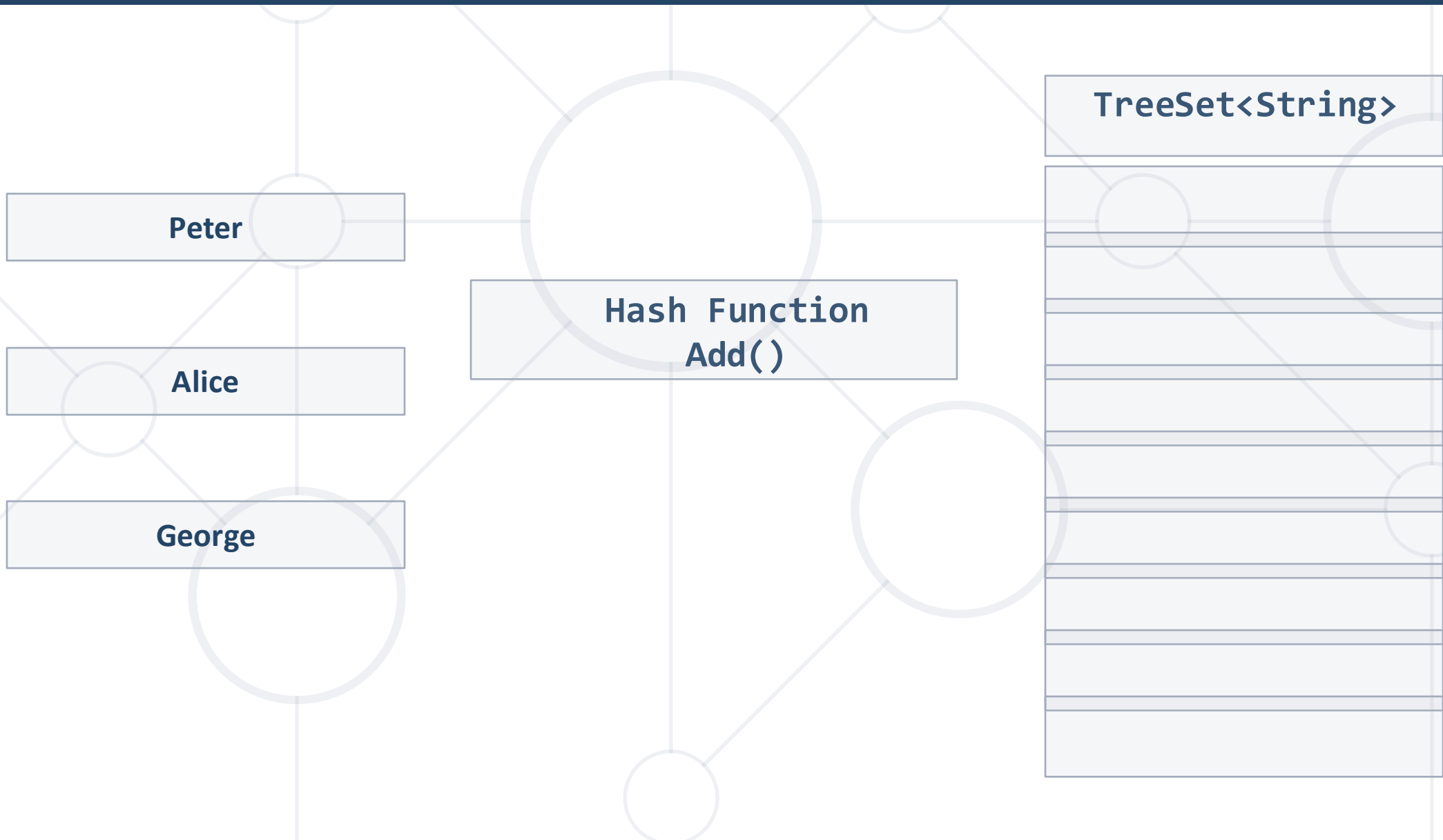
LinkedHashSet<String>

Problem: Parking Lot

- Write a program that:
  - Add car number for every car that enter in parking lot
  - Remove car number when the car go out

Car

CA2844AA

CA2844AA

Parking Lot

CA4466GA
CA2384HT
CA8686RA
CA9999AT

Solution: Parking Lot

```java
HashSet<String> parkingLot = new HashSet<String>();
while(true)
  String input = sc.nextLine();
  if (input.equals("END"))
    break;
  else
    String[] reminder = input.split(", ");
    if (reminder[0].equals("IN"))
      parkingLot.add(reminder[1]);
    else
      parkingLot.remove(reminder[1]);
```

Check your solution here: https://judge.softuni.bg/Contests/1462/Sets-And-Maps-Lab

Problem: SoftUni Party

- Guests are two types:
  - **Regular**
  - **VIPs** – their tickets start with digit
- Until **PARTY** command, you will receive guest invitations
- Until **END** command, you will receive a second list with guests that actually come to the party
- Find how many guests didn't came to the party
- Print all guests that didn't came (VIPs first)

Reservation List

```
7IK9Yo0h
9NoBUajQ
Ce8vwPmE
SVQXQCbc
```

Solution: SoftUni Party

```java
TreeSet<String> vip = new TreeSet<>();
TreeSet<String> regular = new TreeSet<>();
String guestId = scanner.nextLine();

while (!guestId.equals("PARTY")) {
    if (Character.isDigit(guestId.charAt(0)))
        vip.add(guestId);
    else
        regular.add(guestId);
    guestId = scanner.nextLine();
}

//TODO: Remove the guests who came to the party
//TODO: Print results
```

> Return **true** or **false**

Problem: "Voina" – Number Game

- Create a game that is played by two players:

  - Each one **have 20 unique numbers** (read from console, separated wi th space)

  - Every round each player **bet his first number** from deck

  - Player with bigger number win and place both numbers **at the bottom** of his deck

  - Game end after **50 rounds** or when any player have **0 numbers**

Check your solution here: https://judge.softuni.bg/Contests/1462/Sets-And-Maps-Lab

```java
LinkedHashSet<Integer> firstPlayer = getPlayerNumbers();
LinkedHashSet<Integer> secondPlayer = getPlayerNumbers();
for (int i = 0; i < 50; i++) {
    int firstNumber = firstPlayer.iterator().next();
    firstPlayer.remove(firstNumber);
    //TODO: get top number for second player
    if (firstNumber > secondNumber) {
        firstPlayer.add(firstNumber);
        firstPlayer.add(secondNumber);
    } else if (secondNumber > firstNumber)
        //TODO: finish logic about second player win or draw
}
//TODO: print result
```
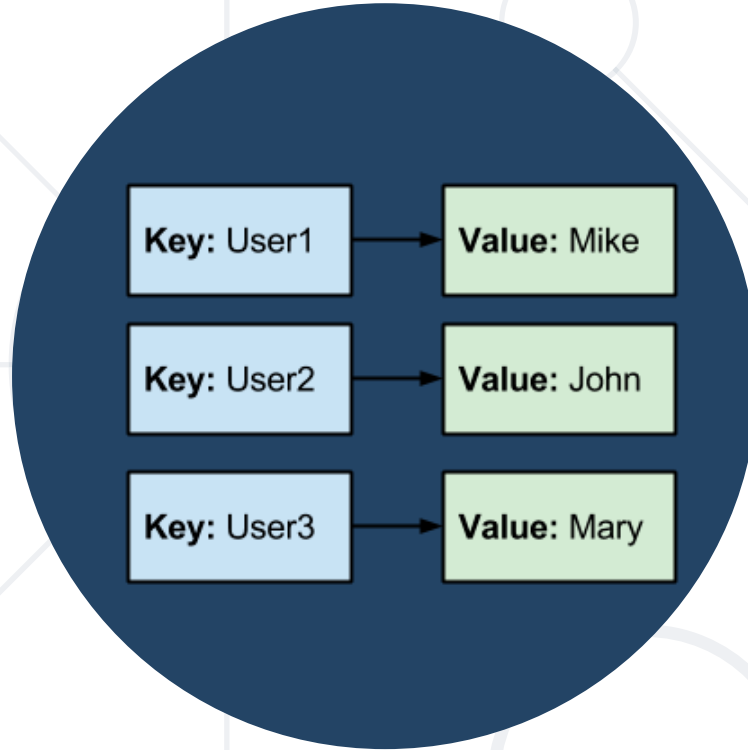
Associative Arrays
HashMap<Key, Value>

Associative Arrays (Maps)

- **Associative arrays** are arrays indexed by keys

  - Not by the numbers 0, 1, 2, …

- Hold a set of pairs **<key, value>**

- **Traditional**

key	0	1	2	3	4
value	8	-3	12	408	33

- **Associative array**

key	value
John Smith	+1-555-8976
Lisa Smith	+1-555-1234
Sam Doe	+1-555-5030

Methods

- Initialization

```
HashSet<String, Integer> hash = new HashSet<String>();
```
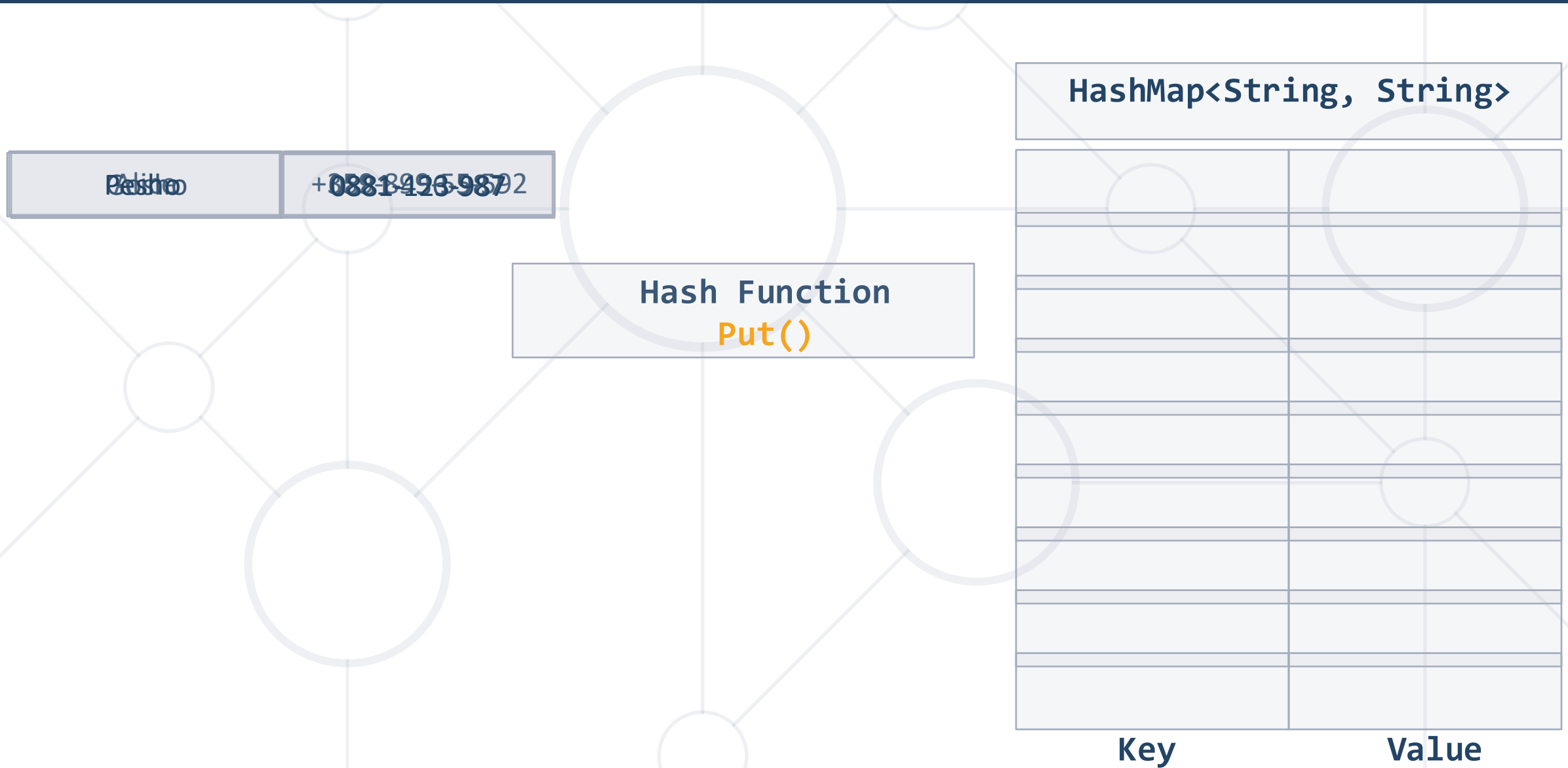
Type of keys

Type of values

- `.size()`

- `.isEmpty()`

```
HashSet<String> hash = new HashSet<>();
System.out.println(hash.size());      // 0
System.out.println(hash.isEmpty());  // True
```

HashMap<K, V> – put()

HashMap<String, String>

Peter
Pesho
Palitko

+081 406 987
0881-426-987

Hash Function
Put()

Key	Value

HashMap<K, V> – remove()

Pesho

Hash Function
Remove()

HashMap<String, String>	
Pesho	0881-123-987
Alice	**+359-899-55-592**
Key	**Value**

```java
HashMap<String, Integer> vehicles = new HashMap<>();
vehicles.put("BMW", 5);
vehicles.put("Mercedes", 3);
vehicles.put("Audi", 4);
vehicles.put("BMW", 10);
for(String key: vehicles.keySet())
    System.out.println(key + " - " + vehicles.get(key));
```

Override first value

Return set of all keys
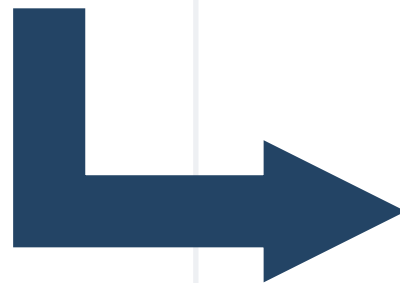
Return value for key

```
Audi - 4
Mercedes - 3
BMW - 10
```

Problem: Count Real Numbers

- Write a program that **counts** in a given array of **double** values the number of occurrences of each value

-2.5	4	3	-2.5	-5.5	4	3	3	-2.5	3

```
-2.5 -> 3
4.0 -> 2
3.0 -> 4
-5.5 -> 1
```

```java
Map<Double, Integer> result = new LinkedHashMap<>();
for (Double number : input) {
  if (!result.containsKey(number)) {
    result.put(number, 1);
  } else {
    result.put(number, result.get(number) + 1);
  }
}

for (Double key : result.keySet()) {
  System.out.println(key + " -> " + result.get(key));
}
```

Check your solution here: https://judge.softuni.bg/Contests/1462/Sets-And-Maps-Lab

SoftUni Foundation

Pesho Alisho	+888-123-9872 +359-123-9872

TreeMap Function
Put()

Tree Map<String, String>	
Key	**Value**

25
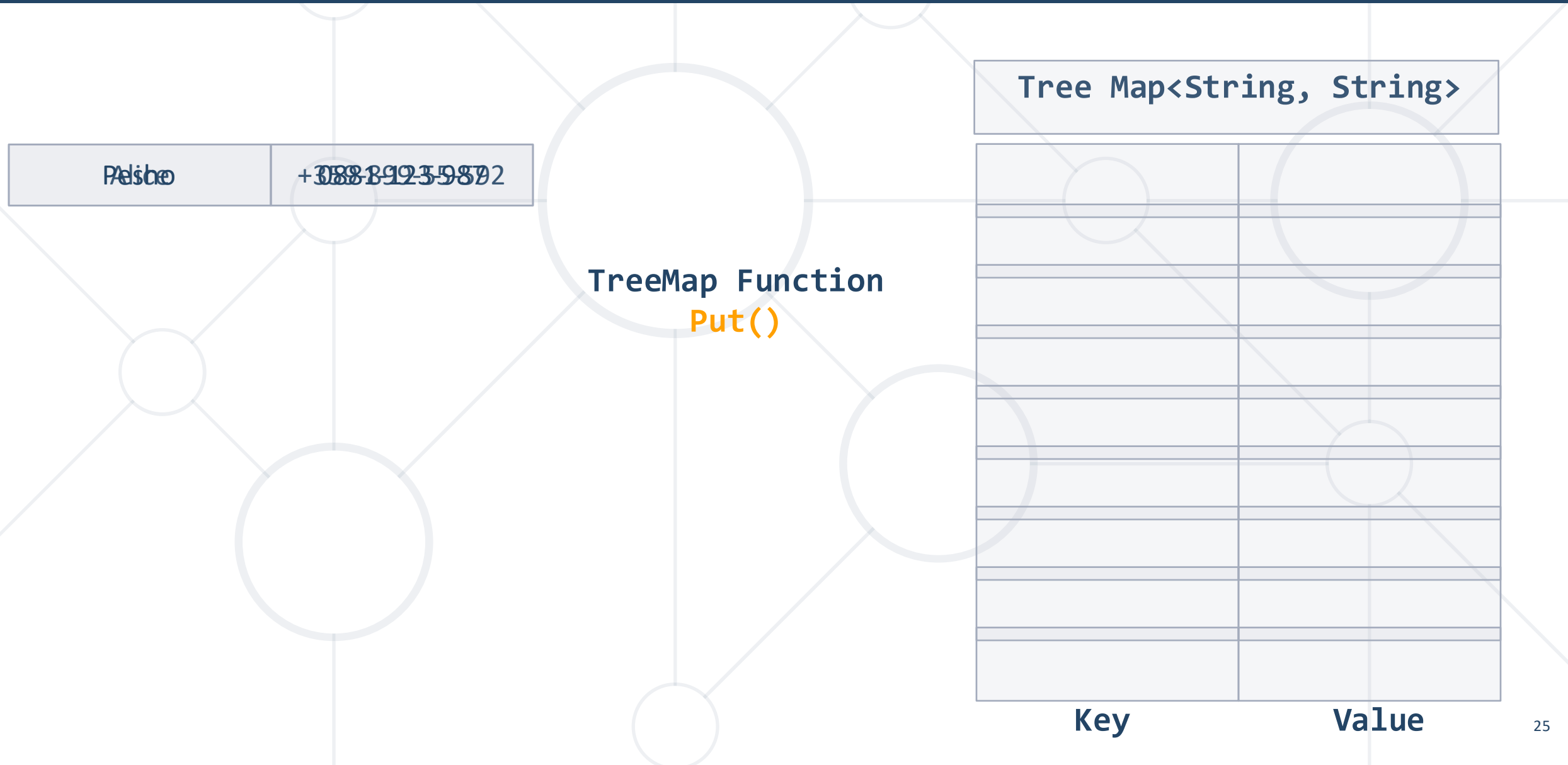
Problem: Academy Graduation

- Write a program that:
  - Read list of students and their score for some courses
  - Print on console sorted list with average score for each student

Student	Java Advanced	Java OOP
Gosho	3.75	5
Mara	4.25	6
Pesho	6	4.5

Student	Average
Gosho	4,375
Mara	5,125
Pesho	5,25

Check your solution here: https://judge.softuni.bg/Contests/1462/Sets-And-Maps-Lab

```java
TreeMap <String,Double[]> graduationList = new TreeMap<>();
for (int i = 0; i < numberOfStudents; i++) {
    String name = scanner.nextLine();
    String[] scoresStrings = scanner.nextLine().split(" ");
    Double[] scores = new Double[scoresStrings.length];

    for (int j = 0; j < scoresStrings.length; j++) {
        scores[j] = Double.parseDouble(scoresStrings[j]);
    }
    graduationList.put(name, scores);
}
//TODO: print results
```
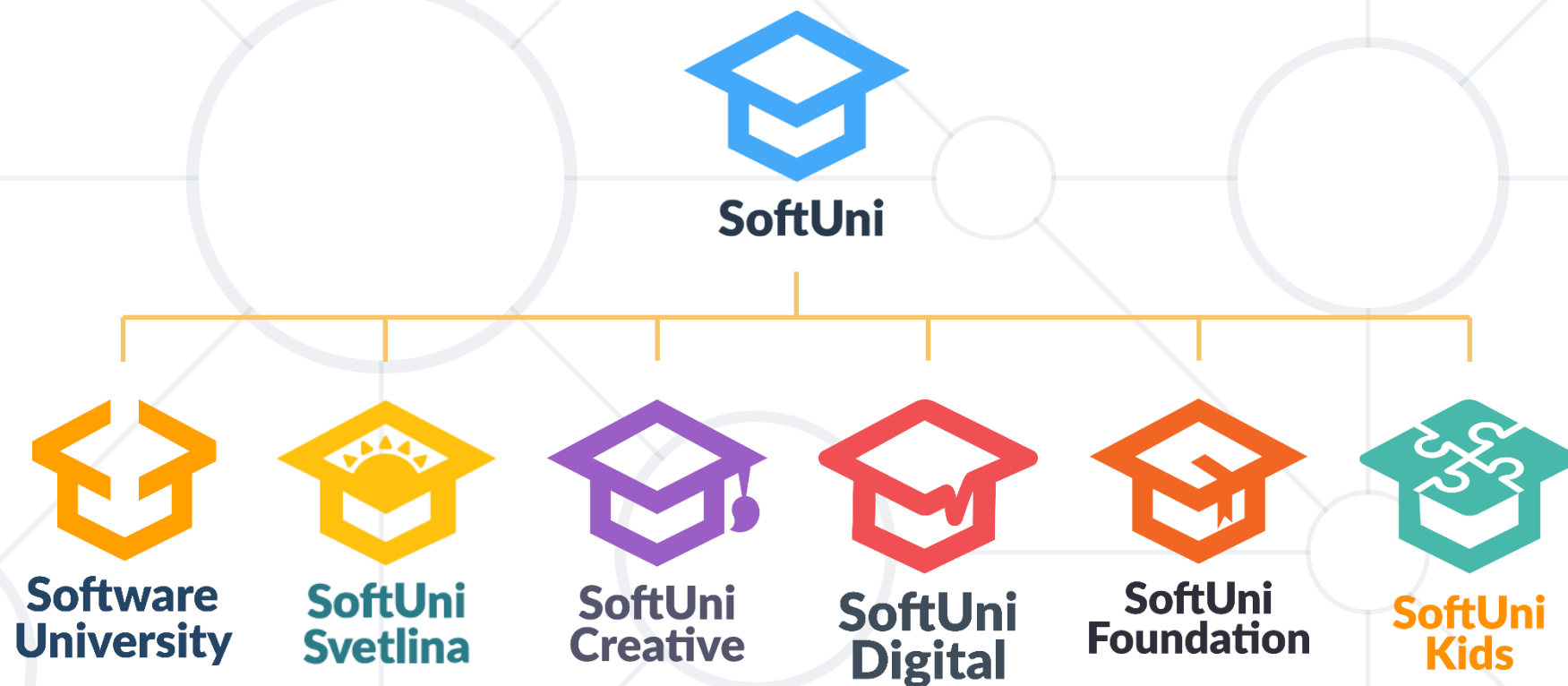
Check your solution here: https://judge.softuni.bg/Contests/1462/Sets-And-Maps-Lab

- **size()** – the number of key-value pairs

- **keySet()** – a set of unique keys

- **values()** – a collection of all values

- Basic operations – **put()**, **remove()**, **clear()**

- Boolean methods:

  - **containsKey()** – checks if a key is present in the Map

  - **containsValue()** – checks if a value is present in the Map

Summary

- **HashSet<E>, TreeSet<E>** and **LinkedHashSet<E>** hold unique elements and are very fast

- **HashMap<K, V>, TreeMap<K, V>** and **LinkedHashMap<K, V>** are an associative arrays where a value is accessed by its key

Questions?

SoftUni

Software University

SoftUni Svetlina

SoftUni Creative

SoftUni Digital

SoftUni Foundation

SoftUni Kids

https://softuni.bg/modules/59/java-advanced

SoftUni Diamond Partners

SoftUni Foundation

XSsoftware

SBTech
we know sports

telenor

SoftwareGroup
doing it right

NETPEAK

SmartIT

Postbank
Решения за твоето утре

SUPER
HOSTING
.BG

INDEAVR
Serving the high achievers

INFRAGISTICS

STEMO
Computer Systems & Software

M

SoftUni Organizational Partners

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education and Employment Opportunities
  - softuni.bg
- Software University Foundation
  - http://softuni.foundation/
- Software University @ Facebook
  - facebook.com/SoftwareUniversity
- Software University Forums
  - forum.softuni.bg

License

- This course (slides, examples, demos, videos, homework, etc.) is licensed under the "Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International" license