
**Universiteti Politeknik i Tiranës**

Fakulteti i Teknologjisë së Informacionit

Dega: Inxhinieri Informatike

Grupi: III-B

Viti akademik 2024-2025

# Punë laboratori nr. 3

Lënda: Algoritmike dhe programim i avancuar

Punoi: Piro Gjidhima

Pranoi:Msc Alba Haveriku

# Laborator 3

**Finite mathematical sets.** Your goal is to develop an implementation of the following API for processing finite mathematical sets (use the HashSet class):

| public class MathSET<Key> | |
|---|---|
| MathSET(Key[] universe) | *create the empty set (using given universe)* |
| void add(Key key) | *put key into the set* |
| MathSET<Key> complement() | *set of keys in the universe that are not in this set* |
| void union(MathSET<Key> a) | *put any keys from a into the set that are not already there* |
| void intersection(MathSET<Key> a) | *remove any keys from this set that are not in a* |
| void delete(Key key) | *remove key from the set* |
| boolean contains(Key key) | *is key in the set?* |
| boolean isEmpty() | *is the set empty?* |
| int size() | *number of keys in the set* |

API for a basic finite set data type

# Klasa MathSET

```
package Laboratore.Lab3;

import java.util.HashSet;

public class MathSET<Key> {
   public Key[] universeArray;
   public HashSet<Key> universe;
   public HashSet<Key> set;

   public MathSET(Key[] universe) {
      this.universeArray = universe;
      this.universe = new HashSet<>();
      for (Key key : universe) {
         this.universe.add(key);
      }
      this.set = new HashSet<>();
   }
```

```
public void add(Key key) {
    if (!universe.contains(key)) {
        throw new IllegalArgumentException("Celesi " + key + " nuk eshte pjese e universit");
    } else {
        set.add(key);
    }
}

public MathSET<Key> complement() {
    MathSET<Key> complement = new MathSET<>(this.universeArray);
    for (Key key : universe) {
        if (!set.contains(key)) {
            complement.add(key);
        }
    }
    return complement;

}

void union(MathSET<Key> a) {
    if (!universe.equals(a.universe)) {
        for (Key key : a.universe) {
            universe.add(key);
        }
    }
    for (Key key : a.set) {
        set.add(key);
    }
}

void intersection(MathSET<Key> a) {
    for (Key key : set) {
        if (!a.set.contains(key)) {
            set.remove(key);
        }
    }
}

void delete(Key key) {
    if (!universe.contains(key)) {
        throw new IllegalArgumentException("Celesi " + key + " nuk eshte pjese e universit");
    }
    if (set.contains(key)) {
        set.remove(key);
    } else {
        throw new IllegalArgumentException("Celesi " + key + " nuk eshte pjese e Setit");
    }
```

```
    }

    boolean contains(Key key) {
        return set.contains(key);
    }

    boolean isEmpty() {
        return set.isEmpty();
    }

    int size() {
        return set.size();
    }

}


        }
        if (set.contains(key)) {
            set.remove(key);
        } else {
            throw new IllegalArgumentException("Celesi " + key + " nuk eshte pjese e Setit");
        }

    }

    boolean contains(Key key) {
        return set.contains(key);
    }

    boolean isEmpty() {
        return set.isEmpty();
    }

    int size() {
        return set.size();
    }

}
```

## Output:

# Klasa Lab3

```
package Laboratore.Lab3;

public class Lab3 {
    public static void main(String[] args) {
        MathSET<Integer> mathSet = new MathSET<>(new Integer[]{1,2,3,4,5,6,7,8,9,10});
        for (int i = 1; i < 7; i++){
            mathSet.add(i);
        }
        System.out.println("Universi: "+mathSet.universe);
        System.out.println("Seti: "+mathSet.set);
        System.out.println("Complementari: "+mathSet.complement().set);

        mathSet.delete(2);a
        mathSet.delete(5);
        mathSet.delete(3);
        System.out.println("Pas Fshirjes");
        System.out.println("Universi: "+mathSet.universe);
        System.out.println("Seti: "+mathSet.set);
        System.out.println("Complementari: "+mathSet.complement().set);

    }
}
```

```
Universi: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Seti: [1, 2, 3, 4, 5, 6]
Complementari: [7, 8, 9, 10]
Pas Fshirjes
Universi: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Seti: [1, 4, 6]
Complementari: [2, 3, 5, 7, 8, 9, 10]

Process finished with exit code 0
```