Built-In API Classes in Java

- Java provides **ready-to-use** classes:
  - Organized inside Packages like:
    `java.util.Scanner`, `java.utils.List`, etc.
- Using static class members:

```java
LocalDateTime today = LocalDateTime.now();
double cosine = Math.cos(Math.PI);
```

- Using non-static Java classes:

```java
Random rnd = new Random();
int randomNumber = rnd.nextInt(99);
```

Solution: Randomize Words

```
Scanner sc = new Scanner(System.in);
String[] words = sc.nextLine().split(" ");
Random rnd = new Random();
for (int pos1 = 0; pos1 < words.length; pos1++) {
   int pos2 = rnd.nextInt(words.length);
   //TODO: Swap words[pos1] with words[pos2]
}
System.out.println(String.join(
                       System.lineSeparator(), words));
```

Check your solution here: https://judge.softuni.org/Contests/1319/

Solution: Big Factorial

Use the java.math.BigInteger

```
import java.math.BigInteger;
...
int n = Integer.parseInt(sc.nextLine());
BigInteger f = new BigInteger(String.valueOf(1));
for (int i = 1; i <= n; i++) {
  f = f.multiply(BigInteger
       .valueOf(Integer.parseInt(String.valueOf(i))));
}
System.out.println(f);
```

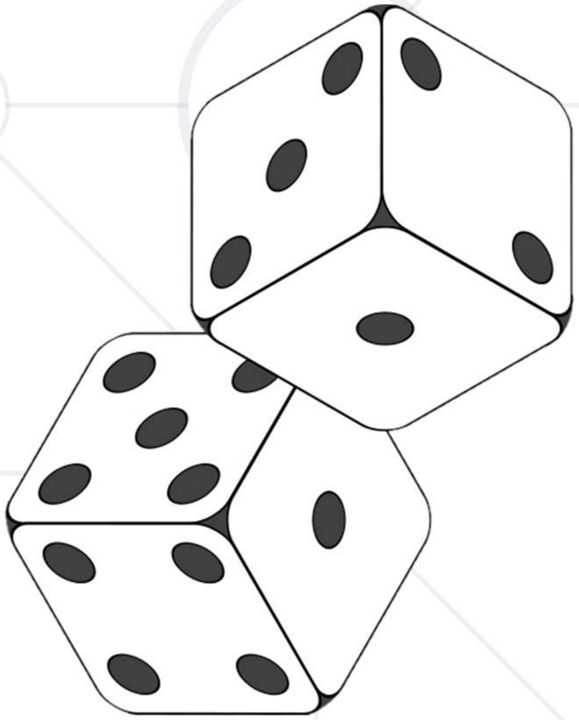
N!

Check your solution here: https://judge.softuni.org/Contests/1319/

Methods

- Store executable code (algorithm)

```
class Dice {
  public int sides;
  public int roll() {
    Random rnd = new Random();
    int sides = rnd.nextInt(this.sides + 1);
    return sides;
  }
}
```



Getters and Setters

```
class Dice {
  . . .
  public int getSides() { return this.sides; }
  public void setSides(int sides) {
    this.sides = sides;
  }
  public String getType() { return this.type; }
  public void setType(String type) {
    this.type = type;
  }
}
```

Getters & Setters

Constructors

- Special methods, executed during object creation

```
class Dice {
  public int sides;
  public Dice() {
    this.sides = 6;
  }
}
```

Constructor name is the same as the name of the class

Overloading default constructor

Constructors (2)

- You can have multiple constructors in the same class

```
class Dice {
  public int sides;
  public Dice() { }
  public Dice(int sides) {
    this.sides = sides;
  }
}
```

```
class StartUp {
public static void main(String[] args) {
  Dice dice1 = new Dice();
  Dice dice2 = new Dice(7);
  }
}
```

Solution: Students (1)

```
public Student(String firstName, String lastName,
                                    int age, String city){
    this.firstName = firstName;
    this.lastName = lastName;
    this.age = age;
    this.city = city;
    // TODO: Implement Getters and Setters
}
```

Solution: Students (2)

```
List<Student> students = new ArrayList<>();
String line;
while (!line.equals("end")) {
  // TODO: Extract firstName, lastName, age, city from the input
  Student existingStudent = getStudent(students, firstName, lastName);
  if(existingStudent != null) {
    existingStudent.setAge(age);
    existingStudent.setCity(city);
  } else {
    Student student = new Student(firstName, lastName, age, city);
    students.add(student);
  }

  line = sc.nextLine();
}
```

Solution: Students (3)

```
static Student getStudent(List<Student> students, String firstName,
                                                  String lastName) {
  for (Student student : students){
    if(student.getFirstName().equals(firstName)
      && student.getLastName().equals(lastName))
      return student;
  }

  return null;
}
```