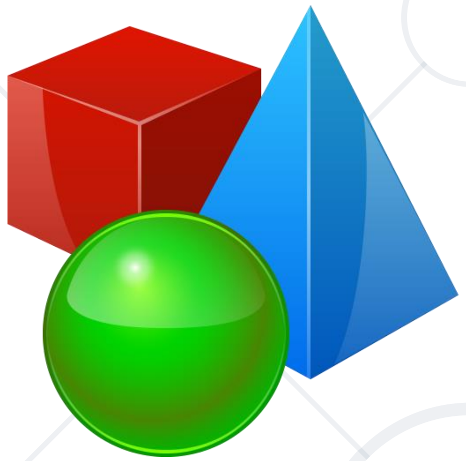
Objects and Classes

Using Objects and Classes

Defining Simple Classes



SoftUni Team

Technical Trainers

Software University

SoftUni

Software University

https://softuni.bg

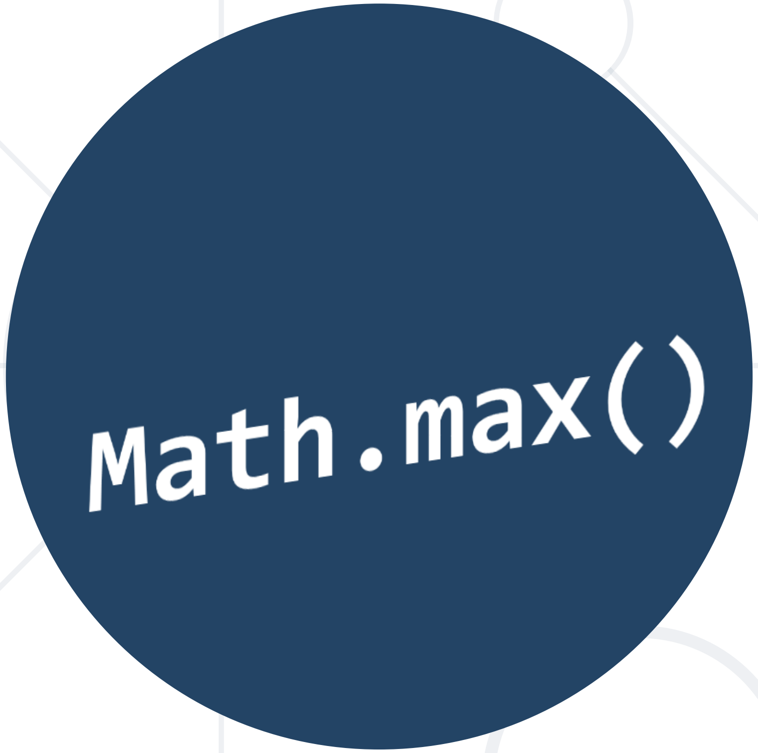
Have a Question?

Software University

sli.do

#fund-java

Table of Contents



Using the Built-In API Classes

Math, Random, BigInteger …

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:

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

- Using non-static Java classes:

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

Problem: Randomize Words

- You are given a list of words
  - Randomize their order and print each word on a separate line

```
a b
```

```
b
a
```

```
PHP Java C#
```

```
Java
PHP
C#
```

Note: the output is a sample. It should always be different!

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

Solution: Randomize Words

```java
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/

Problem: Big Factorial

- Calculate n! (n factorial) for very big n (e.g. 1000)

5 $\Rightarrow$ 120

10 $\Rightarrow$ 3628800

12 $\Rightarrow$ 479001600

50 $\Rightarrow$ 304140932017133780436126081660647688443776415 6
8960512000000000000

88 $\Rightarrow$ 1854826422573984391147968456455462843802209689
4939934668442158098688956218402819931910014124
4804501828416633516851200000000000000000000

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

Solution: Big Factorial

```
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);
```

Use the java.math.BigInteger

N!

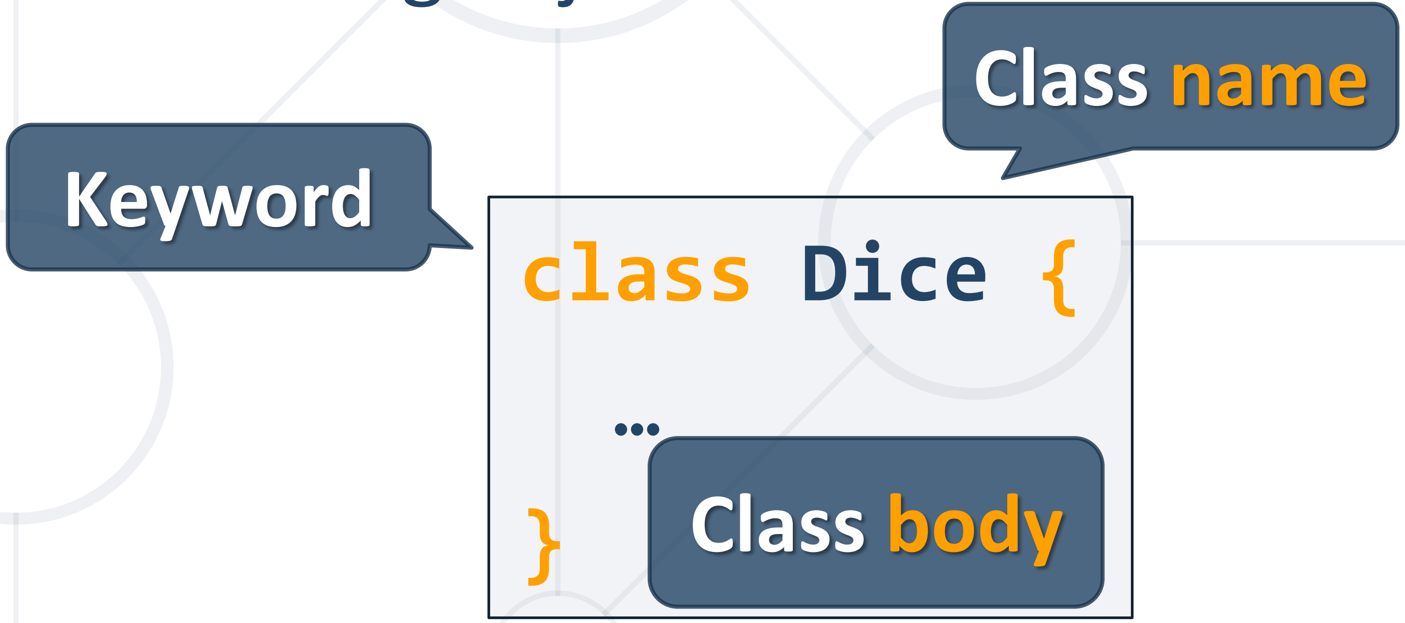
Check your solution here: https://judge.softuni.org/Contests/1319/

Defining Classes

Creating Custom Classes

Defining Simple Classes

- Specification of a given type of objects from the real-world
- Classes provide structure for describing and creating objects



```
class Dice {
    …
}
```

Naming Classes

- Use **PascalCase** naming
- Use **descriptive** nouns
- Avoid abbreviations (except widely known, e.g. URL, HTTP, etc.)

✓

```
class Dice { … }
class BankAccount { … }
class IntegerCalculator { … }
```

🚫

```
class TPMF { … }
class bankaccount { … }
class intcalc { … }
```

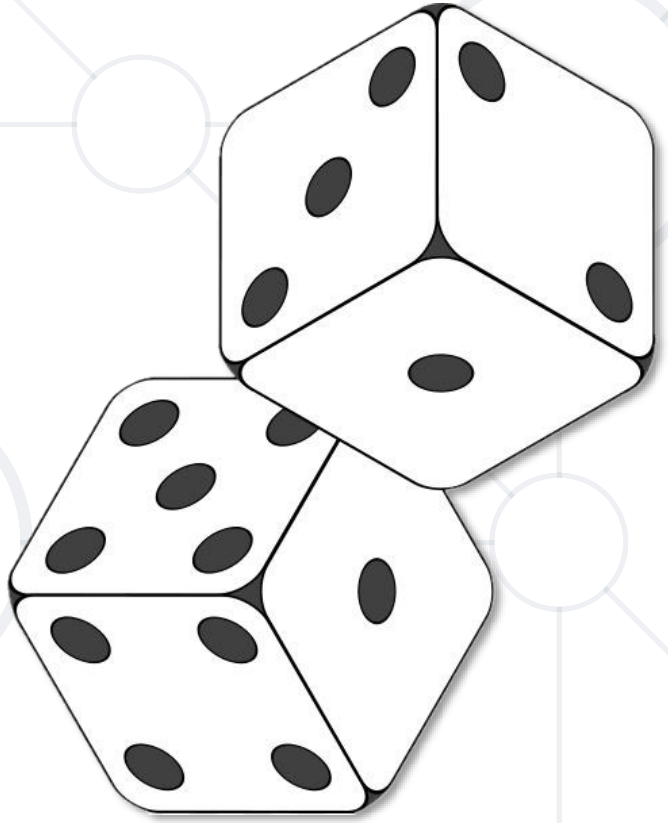
Class Members

- Class is made up of **state** and **behavior**
- Fields **store values**
- Methods **describe behavior**

```
class Dice {
  private int sides;
  public void roll() { … }
}
```

Field

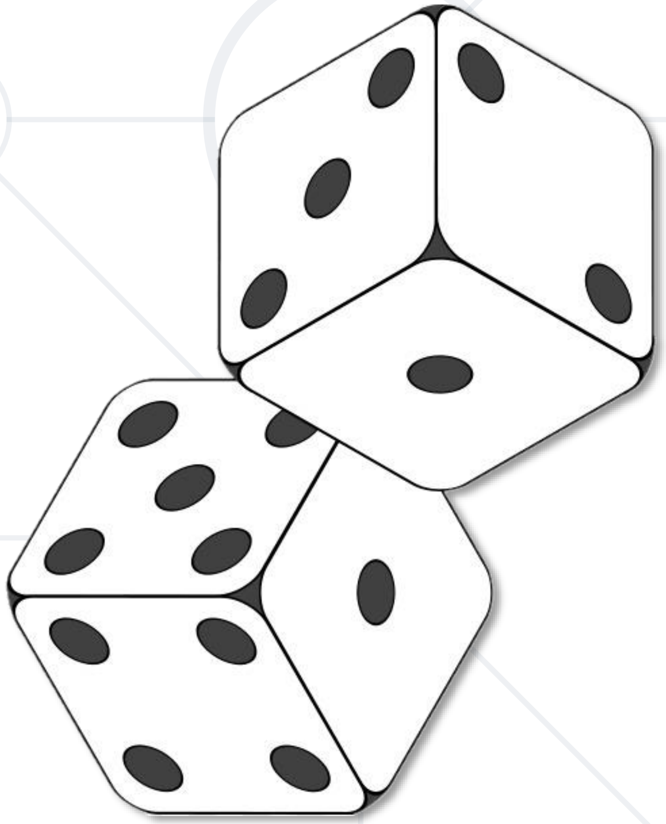
Method



Methods

- Store executable code (algorithm)

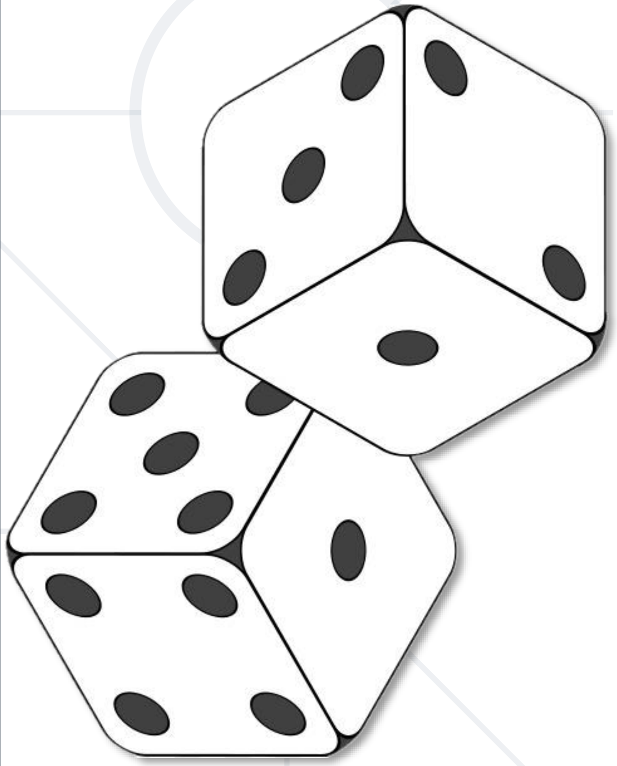
```
class Dice {
  public int sides;
  public int roll() {
    Random rnd = new Random();
    int sides = rnd.nextInt(this.sides + 1);
    return sides;
  }
}
```



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



Creating an Object

- A class can have many **instances** (objects)

```
class Program {
  public static void main(String[] args) {
    Dice diceD6 = new Dice();
    Dice diceD8 = new Dice();
  }
}
```

Use the **new** keyword

Variable stores a **reference**

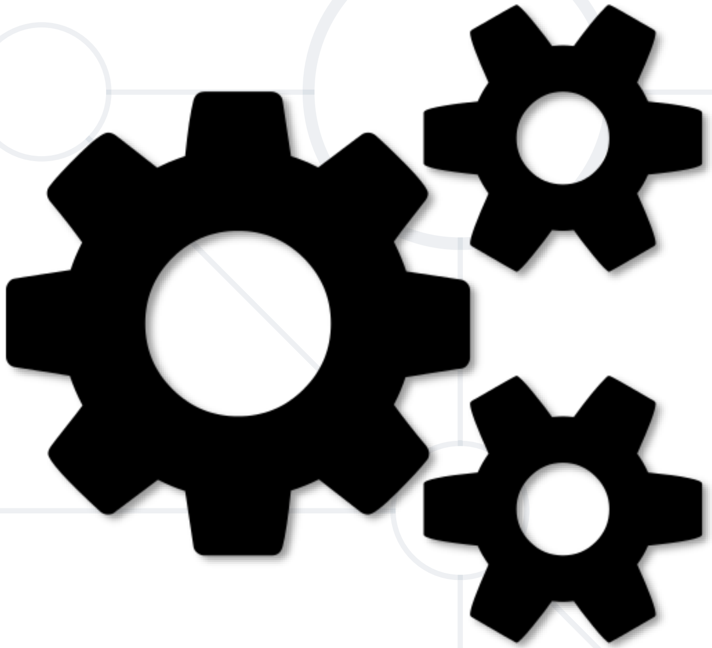
Constructors

- 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

- 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);
  }
}
```

Problem: Students

- Read students until you receive "**end**" in the following format:
  - "`{firstName} {lastName} {age} {hometown}`"
  - Define a class **`Student`**, which holds the needed information
  - If you receive a student which already exists (matching **`firstName`** and **`lastName`**), overwrite the information
- After the end command, you will receive a city name
- Print students which are from the given city in the format:
  "`{firstName} {lastName} is {age} years old.`"

Solution: Students

```java
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

```java
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

```
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;
}
```

Summary

- Classes define templates for object
  - Fields
  - Constructors
  - Methods
- Objects
  - Hold a set of named values
  - Instance of a class

Software University

Questions?

SoftUni

Software University

SoftUni Creative

SoftUni Digital

SoftUni Foundation

SoftUni Kids

Finance Academy

SoftUni Diamond Partners

Software University

SmartIT

Coca-Cola HBC Bulgaria

SOFTWARE GROUP

DXC TECHNOLOGY

createX

Postbank

Решения за твоето утре

SUPER HOSTING .BG

DRAFT KINGS

VIVACOM

INDEAVR

Serving the high achievers

AMBITIONED

PHAR VISION

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education, Profession and Job for Software Developers
  - softuni.bg, about.softuni.bg
- Software University Foundation
  - softuni.foundation
- Software University @ Facebook
  - facebook.com/SoftwareUniversity
- Software University Forums
  - forum.softuni.bg

License