



1918

TALLINNA TEHNIKAÜLIKOO
TALLINN UNIVERSITY OF TECHNOLOGY

IDK0051 Objektorienteeritud programmeerimine Javas

Martin Rebane (martin.rebane@ttu.ee)

Praktiline info

- Aine koduleht (IDK0051)
<https://ained.ttu.ee/>
- Registreerumise võti praktikumis
- **Deklareerige praktikumi õppejõu nimele**



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Meeldetuletus

- 8. nädal praktikume ja loenguid ei toimu (see pole ÕIS-is õige, kuna osadel on kirjas ja osadel mitte)



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Veelkord: kuidas seda ainet õppida?

- Osalege võimalikult palju loengutes ja praktikumides
- Planeerige igal nädalal aeg kodutööks
- Kui hakkate maha jääma, tulge kohe konsultatsiooni!
- Küsige kolleegidelt nõu, kuid ärge tehke copy-pastet



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Eelmisel nädalal: static typing

- Muutuja tüüpi on teada kompileerimise ajal – seda muuta ei saa
 - Vähendab vigade hulka programmi töö ajal
 - Java, C++

```
int a = 5;  
a = "Tere";
```



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Staatilised väljad ja meetodid

- Kuuluvad klassi, mitte objekti juurde
- Staatilises meetodis ei saa kasutada mittestaatilisi välju (IDE veateade: ...cannot reference non-static variable from a static context...)



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Static typing ja static võtmesõna

- Static typing ja static võtmesõna ei ole samad asjad – erinevad!
- Static typing on kontseptsioon, mis ütleb, et igal muutujal on kindel tüüp
- *static* võtmesõna tähistab välja või meetodit, mis ei saa kasutada objekti olekut (mittestaatilisi välju) ja mille väärtus on kõikides klassi objektides sama



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Konstruktor

- Lihtne seletus:
 - Sama nimega, mis klass
 - Tagastatava väärtuse tüüpi ei pea määrama
 - Käivitatakse alati kui klassist objekt luuakse
- Lisainfo:
 - Tagastab **viite loodud objektile**
 - Ühel klassil võib olla mitu erinevat konstruktorit
 - Kui konstruktorit ei ole, kutsutakse automaatselt välja ülemklassi konstruktor



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Koodist töötava programmini

Java programmikood

kirjutate Javas
Python,
Coldfusion



Kompilaator: baitkood

javac
Eclipse compiler in
JDT Core



Java VM käivitab baitkoodi

Java Hotspot

NB! Java baitkood ei sõltu platvormist (Win, Linux etc),
vaid Java virtuaalmasin sõltub.

Terminid

Java programmi **käivitamiseks**

- JRE – Java Runtime Environment, sh virtuaalmasin Java käivitamiseks (java), teegid jms

Java programmi **arendamiseks**

- JDK – Java Development Kit, arendusvanend (kompilaator javac, tööriistad, sisaldab ka JRE asju)



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Täna sed teemad

- Pakett e package
- Klasside pärimine (*inheritance*) ja kompositsioon (*composition*)
- Tüübid Javas
- Tüübiteisendus (*casting*)
- Liidesed (*interface*) ja abstraktne klass
- Geneerilised tüübid (*generic types*)



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Pakett

- Viis koodi struktureerimiseks
- Nii nagu loote oma arvutis kaustu, mitte ei hoia kõike sodi desktopil...
- ... nõnda paigutate ka kokkukuuluva koodi samasse paketti



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Pakett

- Paketis vaikimisi vaikenähtavus (*package-private* ehk *default* nähtavus)
- St ilma nähtavuse võtmesõnata koodiühik on kasutatav paketi sees



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Pakett

- Defineerin, et klass kuulub paketti:

```
package minu.paketi.nimi;
```


- Soovin paketti väljastpoolt kasutada:

```
import java.util.List;
```

```
import minu.paketi.*;
```



1 konkreetne klass



Kõik klassid selles paketis



Quiz

- Teil on projektis klassid:
joop.Lecture
joop.lab.Lab
joop.homework.HomeWork
- Millised klassid impordib joop.* ?



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Mitu võimalikku nähtavust on klassil?

- public ja package-private (*default*)



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Klasside nähtavus paketis

- Klassid, mida peaks välja kutsuma ka väljastpoolt paketti, tuleb deklareerida `public` keywordiga
- Paketisisesed klassid defineerime ilma – kui neid ei ole vaja teistel kasutada



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Kuidas saada miinuspunkte kodutöös?

- Väga lihtne – ärge kasutage pakette
- Asetage kõik kood projekti juurkataloogi
- Miinuspunktid garanteeritud :)



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Java 9 moodulid

- Kes teeb järgmisel nädalal 7-minutilise ettekande?
 - Harjutage kodus läbi, et mahuks 7 minuti sisse
 - Suunake sisu kaastudengitele, mitte õppejõule
 - Andke ülevaade, mis moodulid on ja mida need kaasa toovad

Täna sed teemad

- Pakett e package
- Klasside laiendamine (*inheritance*) ja kompositsioon
- Tüübid Javas
- Tüübiteisendus (*casting*)
- Liidesed (*interface*)
- Geneerilised tüübid (*generic types*)



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Objekti loomine

- Mille tekitab järgmine programmikood: `Student s = new Student(17);`

- Mis on `s`?
- Mis on `Student`?
- Mis on `Student()`?
- Mis on `17`?

Viide objektile

Loodava objekti tüüp

Konstruktori väljakutse, tagastab viite objektile

Argument konstruktorile



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Klasside laiendamine

- Laiendav klass pärib kõik laiendatava klassi (ülemklassi) meetodid, mis ei ole *private*
- Laiendav klass kirjutab üle kõik sama signatuuriga meetodid ülemklassis
- Laiendav klass võib lisada uusi meetodeid



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Klasside laiendamine

- Süntaks *extends* võtmesõnaga:

```
public class Laiendav extends Laiendatav  
{ ... }
```

Ülemtüüp

Alamtüüp

Kasutage laiendamist kui ka tegelikult on olemas ülemtüüp-alamtüüp suhe



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Näited

■ Sobib laiendamiseks:

- Student ja BscStudent
- Fruit ja Apple
- Car ja EstateCar

■ Ei sobi laiendamiseks:

- Car ja Wheel
- Programmable ja Chipset

Ülemtüüp ja alamtüüp

Tervik ja osa

Omadus ja toode



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Kompositsioon

- Kui laiendamine ei sobi, kasutage osa ja terviku korral kompositsiooni
- Üht tüüpi objekti (osa) kasutamine teise objekti sees (tervik)

```
public class Car{  
    Engine engine;  
}
```

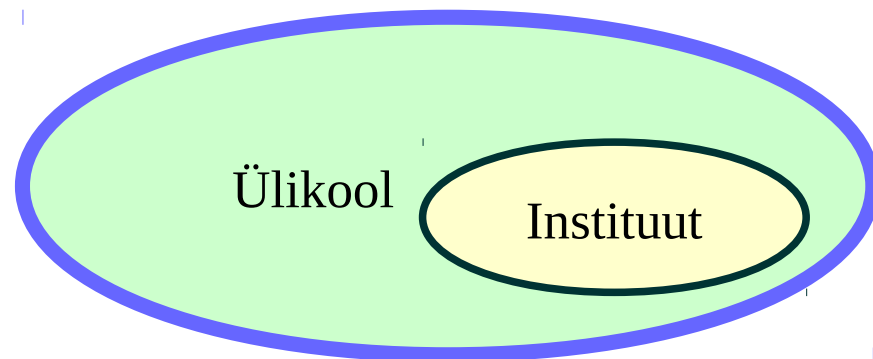
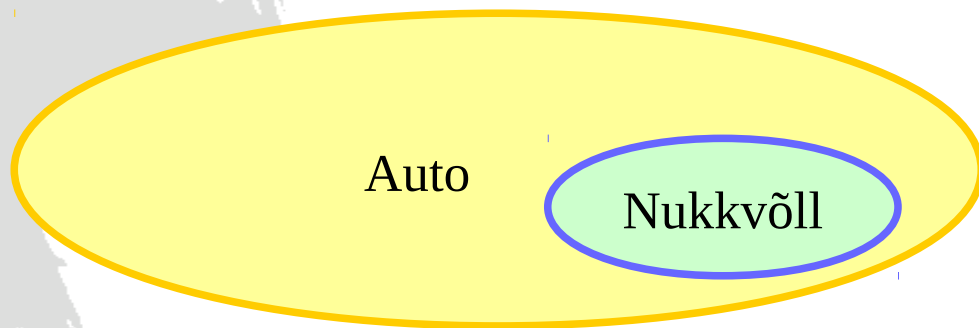


1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Kapseldamine (encapsulation)

- Kas kompositsiooni korral peaksid osad olema public nähtavusega või mitte?



- Sõltub äriloogikast



Täna sed teemad

- Pakett e package
- Klasside laiendamine (*inheritance*)
- Tüübid Javas
- Tüübiteisendus (*casting*)
- Liidesed (*interface*)
- Geneerilised tüübid (*generic types*)



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Tüübid

```
public class Apple extends Fruit {
```

- Objekti saab luua mitmel moel:

```
Apple firstApple = new Apple();  
Fruit secondApple = new Apple();
```



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Üks objekt, erinevad tüübid?

```
Fruit apple = new Apple();
```



Deklareeritud tüüp



Loodud tüüp



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Milline tüüp deklareerida?

- Eelistage alati **üldisemat tüüpi** (kui konkreetne vajadus kitsama tüübi järgi puudub)
- FruitProcessor saab töödelda kõiki puuvilju: Apple, Cherry, Pear... kui neil on sama ülemtüüp



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Täna sed teemad

- Pakett e package
- Klasside laiendamine (*inheritance*)
- Tüübid Javas
- Tüübiteisendus (*casting*)
- Liidesed (*interface*)
- Geneerilised tüübid (*generic types*)



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Tüübiteisendus (ülesteisendus)

- Alamtüüpi objekti saate alati kasutada ülemtüübina (*implicit casting, up-casting*):

```
Apple apple = new Apple();  
Fruit castedApple = apple;
```

- Vastupidine tegevus nõuab selget teisendust (*explicit casting*)



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Tüübiteisendus (*downcasting* e allateisendus)

- Muudate objekti tüüpi kui teil on teada, et mingi (laiemat) tüüpi objekt on tegelikult kitsamat tüüpi:

Deklareerime, et *fruit* on Fruit tüüpi.

JVM loob objekti
ja „unustab”,
et see loodi Applest

```
Fruit fruit = new Apple();  
Apple apple = (Apple)fruit;
```

Kui soovime *fruiti* uuesti Apple'na kasutada,
peame tegema allateisenduse

Miks me selliseid asju vaatama?

- Kuidas on tüübiteisendus, paketid jms seotud infosüsteemide arhitektuuriga?



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Täna sed teemad

- Pakett e package
- Klasside laiendamine (*inheritance*)
- Tüübid Javas
- Tüübiteisendus (*casting*)
- Liidesed (*interface*)
- Geneerilised tüübid (*generic types*)



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY



Mis on liides?



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY



Liides

- Programmi „kasutajaliideseks” arendaja vaatenurgast on tema avalikud väljad ja meetodid
- Mida lihtsam ja selgem süsteem, seda parem kasutada



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Liides e interface

- Kui arendajaid on palju, on vaja omavahelist kokkulepet
- Parim viis kokkuleppeks on programmikood
- Liides on sisuliselt kokkulepe, et arendatav kood peab vastama liideses sätestatule



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Liidese koodinäide

- Liidese **definiitsioon** nagu klassil:

```
public interface Charming { ...
```

- Liidese **kasutamine** klassi defineerimisel:

```
class Princess implements Charming { ...
```



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Tavaline kasutuskoht

- Luuakse mingi süsteem või tarkvara, mida teised liidestada saavad:
 - kokkulepe, kuidas süsteemi kasutada
 - nt **List** interface Java APIs



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Liides kui tüüp

- Liides (*interface*) on samuti Java mõistes tüübi definitsioon.
- Kui klass realiseerib liidest, võib objekti tüübi deklareerida liidese abil:

```
SomeInterface s = new MyClass();
```

Eeldus: `class MyClass implements SomeInterface { ...`

Liides

- Kõik meetodid (olenemata nähtavuse võtmesõnast) on alati *public*
- Võib pakkuda vaikerealisatsiooni *default* meetodiga
- Kõik väljad (sõltumata võtmesõna olemasolust) *final* – ei saa muuta

Tüübierarhia

interface Oriented



Tüüp: Oriented või Programming

class Programming implements Oriented



Tüüp: Oriented, Programming
või MyJava

class MyJava extends Programming



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Abstraktne klass

Abstraktne klass

- Praktiline info: sellisest klassist ei saa luua objekti
- Sellist klassi saab laiendada ja defineerida abstraktseid meetodeid – sarnane liidesega



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Abstraktne klass vs liides

- Abstraktset klassi kasutame reeglina ülemtüübina, kus saame meetoditele ka konkreetse realisatsiooni pakkuda
- Liidest kasutame eelkõige sellise käitumise spetsifitseerimiseks, mis on laiemaks kasutamiseks



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Abstraktne klass vs liides

- Abstraktne meetod – meetod, millel on signatuur, kuid puudub sisu

```
public abstract class TestGround {  
  
    public abstract void testMe();  
  
}
```



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Täna sed teemad

- Pakett e package
- Klasside laiendamine (*inheritance*)
- Tüübid Javas
- Tüübiteisendus (*casting*)
- Liidesed (*interface*)
- Geneerilised tüübid (*generic types*)



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Geneerilised tüübid

- Kui me konkreetset tüüpi ei tea, siis mõnel juhul saame kasutada geneerilist tüüpi (*generic type*)
 - kasutame tüübimuutujat
 - Tüübimuutuja nimi on kokkuleppeline, nt T



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Geneerilised tüübid

- Levinumad:
 - Kolleksioonid (List<**T**>, Map<T> etc): List<**Item**> myItems
 - Optional<**T**>: Optional<**String**> middleName
 - Stream<**T**> (voog): Stream<**Student**> students...

Geneerilise tüübi loomine

- Geneerilise tüübi saate ka ise luua

```
public class Storage<T> {
```

```
    private T t;
```

```
    public T get() {  
        return t;  
    }
```

```
}
```

Geneerilise tüübi
levinuid kokku-
leppeline tähis
on T

Kasutamine:
Storage<Food> store = new Storage<>();

Toortüüp geneerilisest klassist

- Geneerilisest klassist saab luua ka nn **toortüüpe** (*raw type*) – jättes parameetri väärtustamata

```
List items = new ArrayList<>();
```

Toortüübi (raw type) kasutamine

```
List<String> items = new ArrayList<>();
```

String-tüüpi argumendiga parametrizeeritud tüüp

Geneerilised tüübid

- Eelistage geneerilise tüübi kasutamisel argumendi tüübi määramist (mitte toortüüpi – raw type)
- Eelised: kompilaator teab, millise tüübiga on tegu, väldite tüübiteisendusi ja seonduvaid vigu, nt `ClassCastException`



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Typod

- Importige õige list! Kumb sobib objektide nimistu jaoks?

```
import java.awt.List;
```

Ei...

```
import java.util.List;
```

Jah!



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Clean code - clarity is king

- Kodutöö

```
Apple b = new Apple();  
Fruit c = b;
```

vs

```
Apple apple = new Apple();  
Fruit fruit = apple;
```



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Enne järgmise kodutöö tegemist:

- **Läbi lugeda ptk 1 ja 2**

Clean code : a handbook of agile software craftsmanship

Robert C. Martin [et al.] c 2009, 2015

- Kättesaadav online TTÜ võrgust (või üle VPN):

<http://www.ttu.ee/asutused/raamatukogu/10640/andmebaasid-databases/e-raamatud/>

(klikkige **Safari Books Online**, ligipääs „Library access” ja sisenedes otsige pealkirja järgi)

Otselink:

<http://proquestcombo.safaribooksonline.com/9780136083238/firstchapter>



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Miks *clean code*?

- „*Leave the campground cleaner than you found it!*” (see the book)
- Teooriatestis ka *clean code* küsimused
- Clean code on osa kodutööde ja kontrolltööde hindamisest – rohkem me seda loengus ja praksis eraldi ei käsitle



1918

TALLINNA TEHNIKAÜLIKOOL
TALLINN UNIVERSITY OF TECHNOLOGY

Konsultatsioon

- Reedel kl 10:30 ICT-637 (kõik rühmad)
- Eelregistreeruge neljapäeva õhtuks:
<https://doodle.com/poll/22zrrvfu6va84ikp>



1918

TALLINNA TEHNIKAÜLIKOO
TALLINN UNIVERSITY OF TECHNOLOGY