ÖZYEĞİN UNIVERSITY

CS 102

Object Oriented Programming

# Constructors

Reyyan Yeniterzi

reyyan.yeniterzi@ozyegin.edu.tr

September 26, 2016

# Announcements

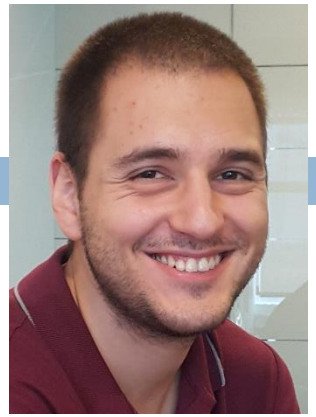


- We will have our first lab (and quiz of course) on tomorrow.
  - Bring your laptops and pencils.
- Slides and codes shown will be uploaded to LMS after lectures.
- Piazza will be up after add/drops.
- Office hours are starting this week.

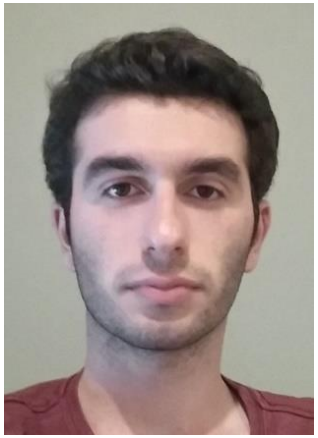
# Teaching Assistants



- Osman Kaya
  - osman.kaya@ozu.edu.tr
- Bahadır Şahin
  - bahadir.sahin@ozu.edu.tr
- Nafiye Polat
  - nafiye.polat@ozu.edu.tr
- Meriç Öztiryaki
  - meric.oztiryaki@ozu.edu.tr
- Cihan Eran
  - cihan.eran@ozu.edu.tr
- Barış Sermet
  - baris.sermet@ozu.edu.tr





# Office Hours (will be held @246 B)



- Monday
  - 11:30-12:40 Bahadır Şahin
  - 14:30-15:40 Bahadır Şahin
- Tuesday
  - 08:40-10:40 CS102 Lab@511
- Wednesday
  - 15:00-17:00 Osman Kaya
- Thursday
  - 08:40-09:30 Cihan Eran
  - 12:30-14:30 Barış Sermet
  - 16:40-17:30 Cihan Eran
- Friday
  - 10:30-12:30 Meriç Öztiryaki
- Saturday
  - 14:00-16:00 Nafiye Polat

# Bank Account – version 3



```java
public static void main(String[] args) {
    Account account1 = new Account();
    account1.number = 1;
    account1.balance = 100;
    account1.currency = "TL";

    Account account2 = new Account();
    account2.number = 2;
    account2.balance = 200;
    account2.currency = "USD";

    account1.report();
    account2.report();

    // Deposit 50TL into account 1
    account1.deposit(50);

    // Deposit 300 USD into account 2
    account2.deposit(300);

    account1.report();
    account2.report();
}
```

# Bank Account – version 3



```
public static void main(String[] args) {
    Account account1 = new Account();
    account1.number = 1;
    account1.balance = 100;
    account1.currency = "TL";

    Account account2 = new Account();
    account2.number = 2;
    account2.balance = 200;
    account2.currency = "USD";

    account1.report();
    account2.report();

    // Deposit 50TL into account 1
    account1.deposit(50);

    // Deposit 300 USD into account 2
    account2.deposit(300);

    account1.report();
    account2.report();
}
```

# Constructors



- Constructors are block of codes which are automatically called when we create objects.

```
// Constructor
public Account(int n, double b, String c) {
    number = n;
    balance = b;
    currency = c;
}
```

- It looks like other methods, but…

# Constructors



- Constructors are block of codes which are automatically called when we create objects.

```
// Constructor
public Account(int n, double b, String c) {
    number = n;
    balance = b;
    currency = c;
}
```

- It looks like other methods, but…
  - It has the same name with the class
  - It does not have a return type

# Calling constructors



```
Account account1 = new Account();
account1.number = 1;
account1.balance = 100;
account1.currency = "TL";
```

```
Account account1 = new Account(1, 100, "TL");
```

```
// Constructor
public Account(int n, double b, String c) {
    number = n;
    balance = b;
    currency = c;
}
```

# No constructors



```
Account account1 = new Account();
account1.number = 1;
account1.balance = 100;
account1.currency = "TL";
```

- We did not have any constructors before?
- How did we create objects without the constructor?

# No constructors



- If there is no explicit constructor, then the default constructor is used.
- Default constructors do not take any parameters.

# Bank Account – version 5



```
public class Account {
    int number;
    double balance;
    String currency;

    // Constructor
    public Account(int n, double b, String c) {
        number = n;
        balance = b;
        currency = c;
    }

    public void deposit(double d) {
        balance = balance + d;
    }

    public void report() {
        System.out.println("Account " + number
                + " has " + balance
                + " " + currency + ".");
    }
}
```

# Bank Account – version 5



```
public static void main(String[] args) {
    Account account1 = new Account(1, 100, "TL");

    Account account2 = new Account(2, 200, "USD");

    account1.report();
    account2.report();

    // Deposit 50TL into account 1
    account1.deposit(50);

    // Deposit 300 USD into account 2
    account2.deposit(300);

    account1.report();
    account2.report();
}
```

# Why do we get the following error?



```
public class AccountTest {
    public static void main(String[] args) {
        Account account1 = new Account(1, 100, "TL");

        Account account2 = new Account();
        account2.number = 2;
        account2.balance = 200;
        account2.currency = "USD";

        account1.report();
        account2.report();

        // Deposit 50TL into account 1
        account1.deposit(50);

        // Deposit 300 USD into account 2
        account2.deposit(300);

        account1.report();
        account2.report();
    }
}
```

# Why do we get the following error?



```
public class AccountTest {
    public static void main(String[] args) {
        Account account1 = new Account(1, 100, "TL");

        Account account2 = new Account();
```

The constructor Account() is undefined

3 quick fixes available:

- Add arguments to match 'Account(int, double, String)'
- Change constructor 'Account(int, double, String)': Remove parameters 'int, double, String'
- Create constructor 'Account()'

Press 'F2' for focus

```
        // Deposit 50TL into account 1
        account1.deposit(50);

        // Deposit 300 USD into account 2
        account2.deposit(300);

        account1.report();
        account2.report();
    }
}
```

# Default Constructor



- When a constructor (with parameters) is implemented, then the system does not provide a default (without parameters) constructor.

# Default Constructor



- When a constructor (with parameters) is implemented, then the system does not provide a default (without parameters) constructor.
- Can we implement our own constructor without parameters?

# Multiple Constructors



- Yes, we can…
- A class can have multiple constructors.
- This is possible by overloading constructors.

# Multiple Constructors



- Yes, we can…
- A class can have multiple constructors.
- This is possible by overloading constructors.

**Function overloading** gives us the capability to implement a particular function in different ways. Overloaded functions will have the same name but different function arguments.

# Multiple Constructors



```
// Constructors
public Account() {

}
public Account(int n, double b, String c) {
    number = n;
    balance = b;
    currency = c;
}
```

# Multiple Constructors



```
// Constructors
public Account() {

}
public Account(int n, double b, String c) {
    number = n;
    balance = b;
    currency = c;
}
```

```
public class AccountTest {
    public static void main(String[] args) {
        Account account1 = new Account(1, 100, "TL");

        Account account2 = new Account();
        account2.number = 2;
        account2.balance = 200;
        account2.currency = "USD";
```

# Multiple Constructors



- Can we have more than two overloaded constructors?

# Multiple Constructors



- Can we have more than two overloaded constructors?
- Yes we can…

# Multiple Constructors



```
// Constructors
public Account() {

}
public Account(int n, double b, String c) {
    number = n;
    balance = b;
    currency = c;
}
public Account(int n, String c) {
    number = n;
    balance = 0;
    currency = c;
}
public Account(int n) {
    number = n;
    balance = 0;
    currency = "TL";
}
```

# Multiple Constructors



```
// Constructors
public Account() {

}
public Account(int n, double b, String c) {
    number = n;
    balance = b;
    currency = c;
}
public Account(int n, String c) {
    number = n;
    balance = 0;
    currency = c;
}
public Account(int n) {
    number = n;
    balance = 0;
    currency = "TL";
}
```

All these constructors do the same thing which is creating an object, but what they assign to the class instances are different.

# Bank Account – version 6



```
public class AccountTest {
    public static void main(String[] args) {
        Account account1 = new Account(1, 100, "TL");

        Account account2 = new Account();
        account2.number = 2;
        account2.balance = 200;
        account2.currency = "USD";

        Account account3 = new Account(3);

        Account account4 = new Account(4, "EURO");

        account1.report();
        account2.report();
        account3.report();
        account4.report();
    }
}
```

# Bank Account – version 6



```java
public class AccountTest {
    public static void main(String[] args) {
        Account account1 = new Account(1, 100, "TL");

        Account account2 = new Account();
        account2.number = 2;
        account2.balance = 200;
        account2.currency = "USD";

        Account account3 = new Account(3);

        Account account4 = new Account(4, "EURO");

        account1.report();
        account2.report();
        account3.report();
        account4.report();
    }
}
```

Javadoc | Declaration | Console

```
<terminated> AccountTest (5) [Java Application]
Account 1 has 100.0 TL.
Account 2 has 200.0 USD.
Account 3 has 0.0 TL.
Account 4 has 0.0 EURO.
```

# Lets add more to our account!



- interest rate (double)

```
int number;
double balance;
String currency;
double interestRate;
```

# Modify the constructors



```
// Constructors
public Account() {

}
public Account(int n, double b, String c, double i) {
    number = n;
    balance = b;
    currency = c;
    interestRate = i;
}
public Account(int n, String c) {
    number = n;
    balance = 0;
    currency = c;
    interestRate = 0;
}
public Account(int n) {
    number = n;
    balance = 0;
    currency = "TL";
    interestRate = 0;
}
```

# Add more constructors - I



```
public Account(int n, double b, String c) {
    number = n;
    balance = b;
    currency = c;
    interestRate = 0;
}
```

```
Account account5 = new Account(5, 200, "TL");
```

# Add more constructors - II



```
public Account(int n, double i, String c) {
    number = n;
    balance = 0;
    currency = c;
    interestRate = i;
}
```

```
Account account6 = new Account(5, 0.02, "TL");
```

# Any problem you see?



```
public Account(int n, double b, String c) {
    number = n;
    balance = b;
    currency = c;
    interestRate = 0;
}
```

```
Account account5 = new Account(5, 200, "TL");
```

```
public Account(int n, double i, String c) {
    number = n;
    balance = 0;
    currency = c;
    interestRate = i;
}
```

```
Account account6 = new Account(5, 0.02, "TL");
```

# Multiple Constructors



- You can have multiple constructors as long as they have different argument lists.
- System differentiates constructors based on their argument lists, therefore two constructors with same argument list cause compiler error.

# Duplicate method error!



```
public Account(int n, double b, String c) {
    number = n;
    balance = b;
    currency = c;
    interestRate = 0;
}
public Account(int n, double i, String c) {
    num
    bal
    cur
    int
}
```

Duplicate method Account(int, double, String) in type Account

1 quick fix available:

Rename method 'Account' (Ctrl+2, R)

Press 'F2' for focus

# Duplicate method error!



```
public Account(int n, double b, String c) {
    number = n;
    balance = b;
    currency = c;
    interestRate = 0;
}
public Account(int n, double i, String c) {
    num
    bal
    cur
    int
}
```

Duplicate method Account(int, double, String) in type Account

1 quick fix available:

Rename method 'Account' (Ctrl+2, R)

Press 'F2' for focus

- Any idea to fix this?

# Multiple Constructors



```
public Account(int n, double b, String c) {
    number = n;
    balance = b;
    currency = c;
    interestRate = 0;
}
public Account(int n, String c, double i) {
    number = n;
    balance = 0;
    currency = c;
    interestRate = i;
}
```

- Same type of arguments, but their order is different!

# Bank Account – version 7



- Be careful when calling these functions!

```
Account account1 = new Account(1, 100, "TL");

Account account2 = new Account();

Account account3 = new Account(3);

Account account4 = new Account(4, "EURO");

Account account5 = new Account(5, 200, "TL");

Account account6 = new Account(5, "TL", 0.02);
```

```java
public Account() {

}
public Account(int n, double b, String c, double i) {
    number = n;
    balance = b;
    currency = c;
    interestRate = i;
}
public Account(int n, String c) {
    number = n;
    balance = 0;
    currency = c;
    interestRate = 0;
}
public Account(int n) {
    number = n;
    balance = 0;
    currency = "TL";
    interestRate = 0;
}
```

```java
public Account(int n, double b, String c) {
    number = n;
    balance = b;
    currency = c;
    interestRate = 0;
}
public Account(int n, String c, double i) {
    number = n;
    balance = 0;
    currency = c;
    interestRate = i;
}
```

```java
Account account1 = new Account(1, 100, "TL");

Account account2 = new Account();

Account account3 = new Account(3);

Account account4 = new Account(4, "EURO");

Account account5 = new Account(5, 200, "TL");

Account account6 = new Account(5, "TL", 0.02);
```

# Overloading Functions



- We have overloaded the constructor.
- Can we overload other functions as well?

# Overloading Functions



- We have overloaded the constructor.
- Can we overload other functions as well?
- Yes, we can…

# Overloading deposit function



```
public void deposit(double d) {
    balance = balance + d;
}

public void deposit() {
    balance = balance + 0;
}
```

# Bank Account – version 8



```
public void deposit(double d) {
    balance = balance + d;
}

public void deposit() {
    balance = balance + 0;
}
```

```
public static void main(String[] args) {

    Account account1 = new Account(1, 100, "TL");
    Account account2 = new Account(2);

    account2.deposit(100);
    account1.deposit();

    account1.report();
    account2.report();
}
```

# Bank Account – version 8



```java
public void deposit(double d) {
    balance = balance + d;
}

public void deposit() {
    balance = balance + 0;
}
```

Javadoc | Declaration | Console

```
<terminated> AccountTest (7) [Java Application
Account 1 has 100.0 TL.
Account 2 has 100.0 TL.
```

```java
public static void main(String[] args) {

    Account account1 = new Account(1, 100, "TL");
    Account account2 = new Account(2);

    account2.deposit(100);
    account1.deposit();

    account1.report();
    account2.report();
}
```

# Overloading deposit function



- In addition to our two deposit functions, can we have the following function as well?

```
public double deposit(double m) {
    balance = balance + m;
    return balance;
}
```

# Overloading deposit function



```
public void deposit(double d) {
    balance = balance + d;
}
public double deposit(double m) {
    balance =
    return bal
}
public void de
    balance = balance + 0;
}
```

Duplicate method deposit(double) in type Account

1 quick fix available:

Rename method 'deposit' (Ctrl+2, R)

Press 'F2' for focus

- Overloaded methods need to have different function arguments (parameter list)
- If the arguments are same but the return type is different, we will still get compiler error

# Any Questions ?