CS 102

Object Oriented Programming

First Class: Bank Account

Reyyan Yeniterzi

reyyan.yeniterzi@ozyegin.edu.tr

September 20, 2016

Announcements

- You will have your first lab (and quiz of course) on next Tuesday.
- Slides and codes shown will be uploaded to LMS after lectures.

Bank Account

- Lets implement a bank account program
- What type of information do we need for a bank account?

Bank Account

- Lets implement a bank account program
- What type of information do we need for a bank account?
  - Account ID
  - Balance
  - Currency

Bank Account

- Lets implement a bank account program
- What type of information do we need for a bank account?
  - Account ID (int)
  - Balance (double)
  - Currency (String)

Bank Account


```
public class AccountTest {

    public static void main(String [] args){

        int account1ID = 1;
        double account1Balance = 1000;
        String account1Currency="TL";

    }
}
```

Bank Account


```
public class AccountTest {

    public static void main(String [] args){

        int account1ID = 1;
        double account1Balance = 1000;
        String account1Currency="TL";

    }
}
```

- int and double are primitive types
- String is an object type

Bank Account


```
public class AccountTest {

    public static void main(String [] args){

        int account1ID = 1;
        double account1Balance = 1000;
        String account1Currency="TL";

    }
}
```

- int and double are **primitive types**
- String is an **object type**
- What is primitive type? What is object type?

Primitive types

- 8 types
  - byte
  - short (16 bit signed)
  - int (32 bit signed)
  - long (64 bit)
  - float (32 bit floating point)
  - double (64 bit floating point)
  - boolean
  - char

Object types

- Everything else that is not primitive
  - Arrays
  - All other user defined classes
- An object can be created with the **new** keyword
  - int [] myArray = new int [10];
- When **new** keyword is used, some space to store this object is allocated from the memory.
- Where in memory?

Memory Allocation

- When you declare a variable in a program, Java allocates space for that variable from one of several memory regions.
  - Heap
  - Stack

Memory Allocation

- Heap
  - Holds objects created in the program
- Stack
  - Used during the execution of the program
  - Stack holds
    - short lived objects (local primitive types)
      - When a function is called a block of memory (stack frame) is allocated to hold the local variables. It is removed when the execution of function finishes
    - references to other objects in the heap

Memory Allocation

- Heap vs. Stack
  - Heap holds the objects where Stack holds reference to these objects
- Objects
  - When **new** keyword is used, some space to store this object is allocated from the **heap** memory.

Memory Allocation
Variable declaration

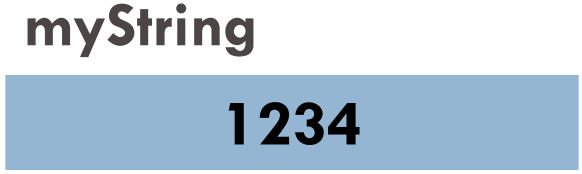
- Primitive type
  - int myInt;

- Object type
  - String myString;

Memory Allocation
Variable assignment

- Primitive type
  - int myInt;
  - myInt = 5;

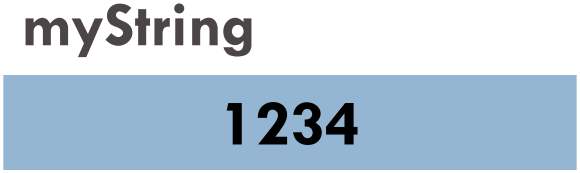
- Object type
  - String myString;
  - myString = new String("Text");



Address of String object in heap

Memory Allocation
Variable assignment

- myString keeps reference to the String object.
  It keeps the location (address) of the object.
- Object type
  - String myString;
  - myString = new String("Text");

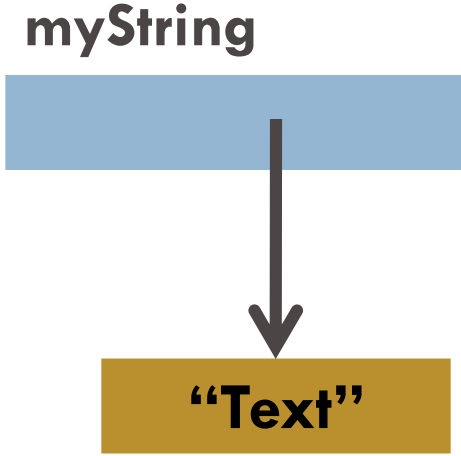


Address of String object in heap

Memory Allocation
Variable assignment

- Primitive type
  - int myInt;
  - myInt = 5;

- Object type
  - String myString;
  - myString = new String("Text");

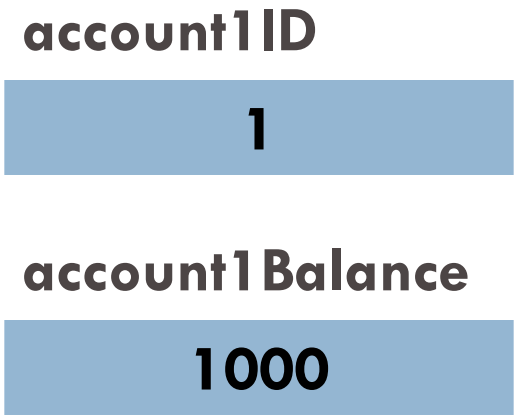


- Instead of showing the address, we will use an arrow

Bank Account

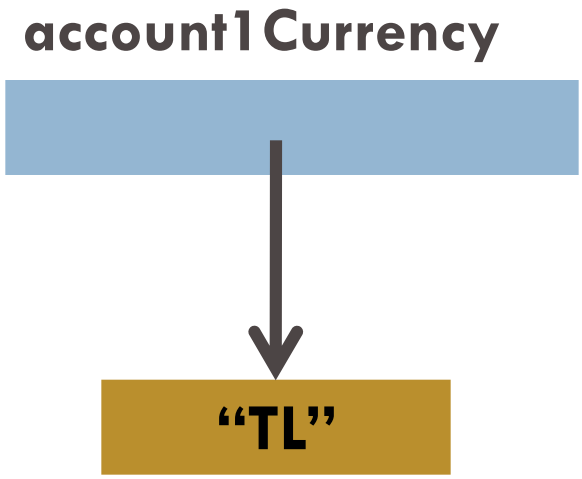

```
public class AccountTest {

    public static void main(String [] args){

        int account1ID = 1;
        double account1Balance = 1000;
        String account1Currency="TL";

    }
}
```

- Primitive types are stored in Stack



Bank Account

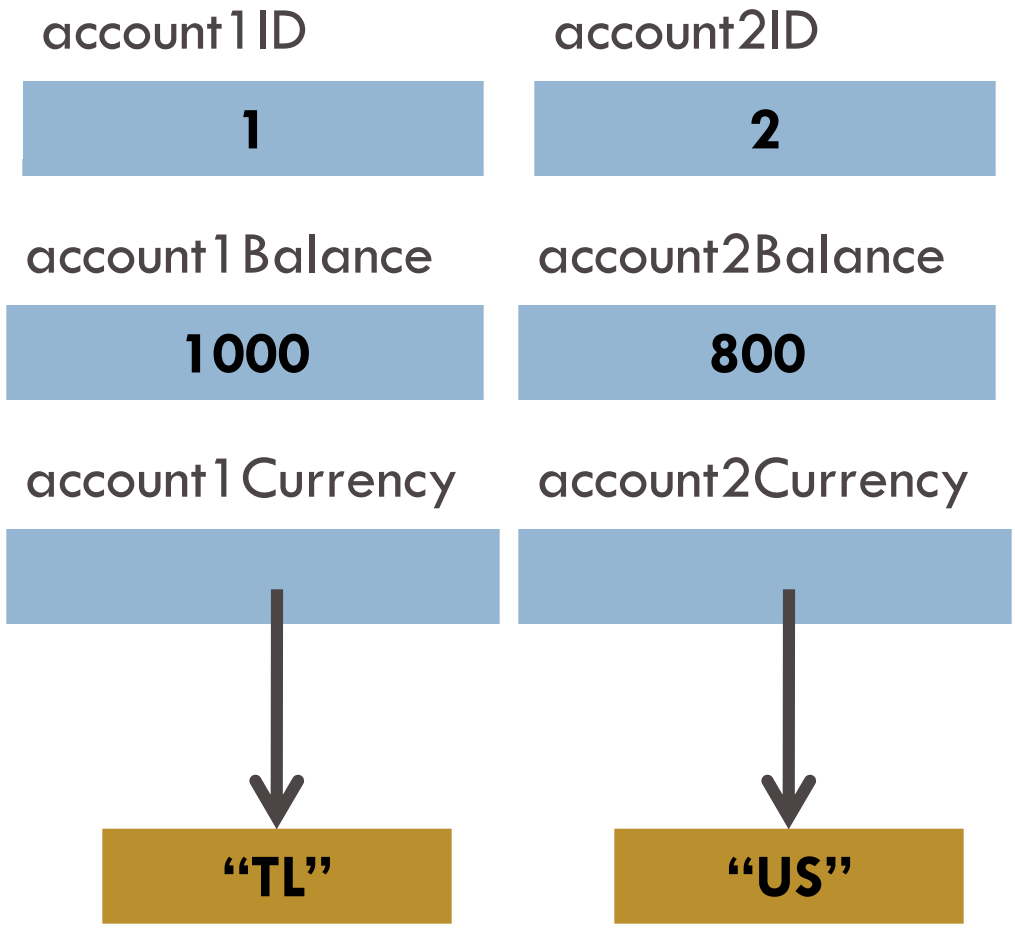

```
public class AccountTest {

    public static void main(String [] args){

        int account1ID = 1;
        double account1Balance = 1000;
        String account1Currency="TL";

    }
}
```



- Objects are stored in Heap
- Their reference is stored in Stack

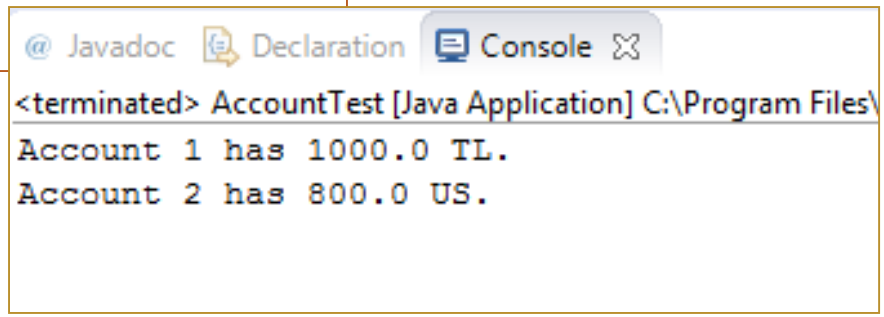
Bank Account


```
int account1ID = 1;
double account1Balance = 1000;
String account1Currency="TL";

int account2ID = 2;
double account2Balance = 800;
String account2Currency="US";
```



Bank Account


```
int account1ID = 1;
double account1Balance = 1000;
String account1Currency="TL";

int account2ID = 2;
double account2Balance = 800;
String account2Currency="US";

System.out.println("Account "+account1ID+
        " has "+account1Balance+" "+
        account1Currency+".");

System.out.println("Account "+account2ID+
        " has "+account2Balance+" "+
        account2Currency+".");
```



Depositing Money


```
// Deposit 50TL into account 1
account1Balance = account1Balance + 50;

// Deposit 300 USD into account 2
account2Balance = account2Balance + 300;
```

- Before

Depositing Money

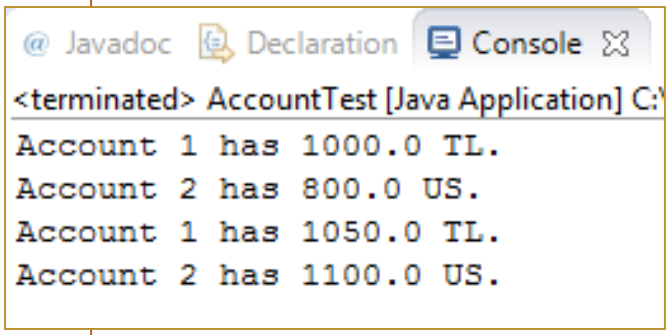

```
// Deposit 50TL into account 1
account1Balance = account1Balance + 50;

// Deposit 300 USD into account 2
account2Balance = account2Balance + 300;
```

- After

account1Balance	account2Balance
1050	1100

Printing Account Details


```
System.out.println("Account "+account1ID+
        " has "+account1Balance+" "+
        account1Currency+".");

System.out.println("Account "+account2ID+
        " has "+account2Balance+" "+
        account2Currency+".");

// Deposit 50TL into account 1
account1Balance = account1Balance + 50;

// Deposit 300 USD into account 2
account2Balance = account2Balance + 300;

System.out.println("Account "+account1ID+
        " has "+account1Balance+" "+
        account1Currency+".");

System.out.println("Account "+account2ID+
        " has "+account2Balance+" "+
        account2Currency+".");
```



Account Class

- Each account has an ID, balance and currency.
  - These can be thought as attributes of an account.
- Can we have an account object which holds all these necessary information together?

Account Class

- Each account has an ID, balance and currency.
- Can we have an account object which holds all these necessary information together?

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

Account Class

- Each account has an ID, balance and currency.
- Can we have an account object which holds all these necessary information together?

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

- These are called **member variables** of the class or **fields.**
- They are also referred to as **instance variables.**

Bank Account – version 1


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

Account.java

AccountTest.java

```
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";
```

Bank Account – version 1


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

Account.java

This is the class which provides the specifics of the Account object

AccountTest.java

In here, we have two Account objects: account1 and account2

```
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";
```

Version 1 - Memory Layout


```
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";
```

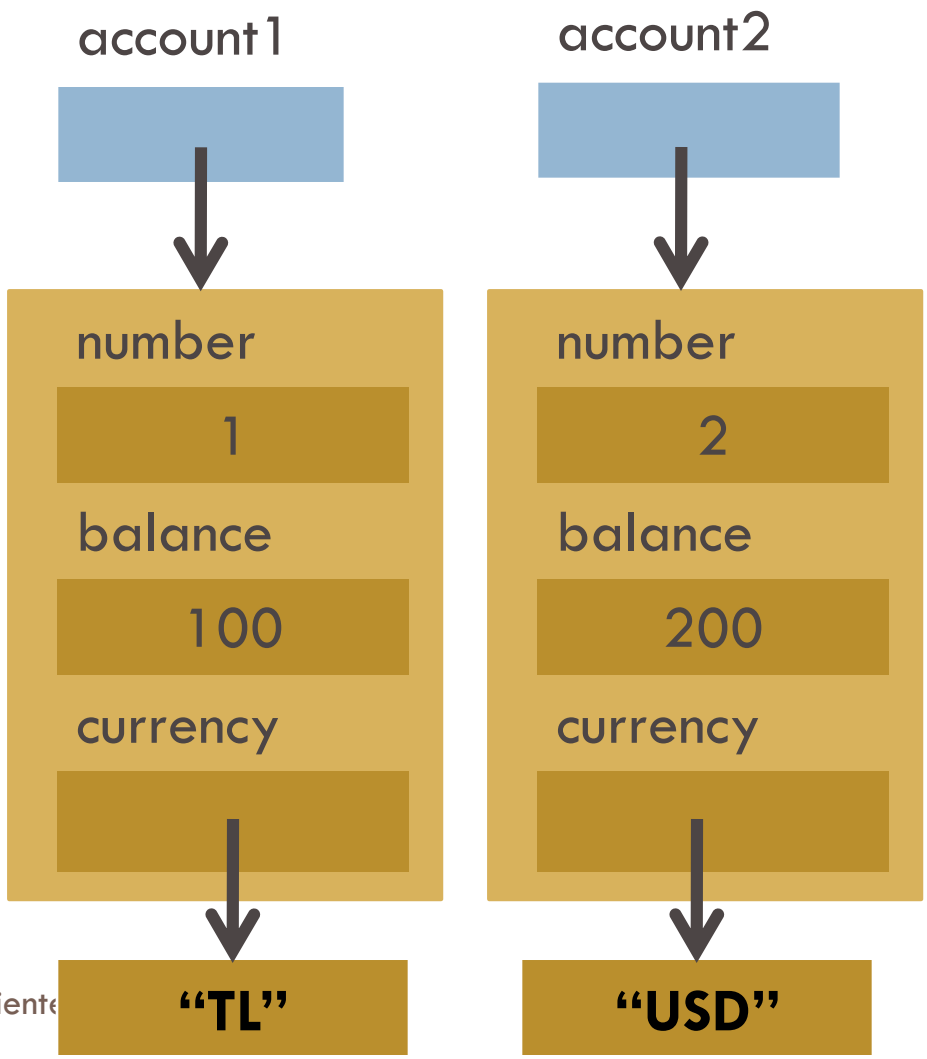
Any idea how this is going to be kept in memory?

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

Version 1 - Memory Layout

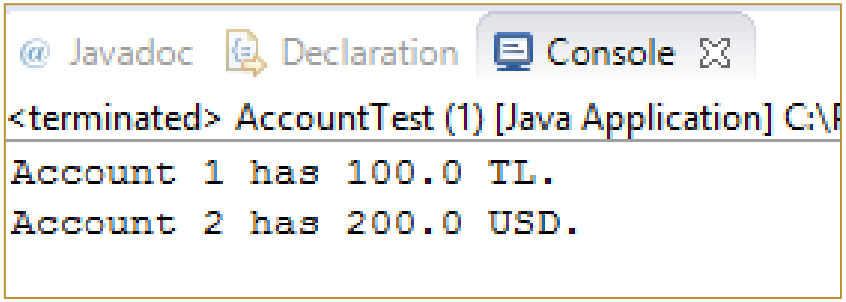

```
Account account1 = new Account();
account1.number = 1;
account1.balance = 100;
account1.currency = "TL";

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

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



Printing Class Variables


```
System.out.println("Account " + account1.number
        + " has " + account1.balance
        + " " + account1.currency + ".");
System.out.println("Account " + account2.number
        + " has " + account2.balance
        + " " + account2.currency + ".");
```



What will be the output?


```
System.out.println("Account " + account1.number
        + " has " + account1.balance
        + " " + account1.currency + ".");
System.out.println("Account " + account2.number
        + " has " + account2.balance
        + " " + account2.currency + ".");

// Deposit 50TL into account 1
account1.balance = account1.balance + 50;

// Deposit 300 USD into account 2
account2.balance = account2.balance + 300;

System.out.println("Account " + account1.number
        + " has " + account1.balance
        + " " + account1.currency + ".");
System.out.println("Account " + account2.number
        + " has " + account2.balance
        + " " + account2.currency + ".");
```

What will be the output?


```
System.out.println("Account " + account1.number
        + " has " + account1.balance
        + " " + account1.currency + ".");
System.out.println("Account " + account2.number
        + " has " + account2.balance
        + " " + account2.currency + ".");

// Deposit 50TL into account 1
account1.balance = account1.balance + 50;

// Deposit 300 USD into account 2
account2.balance = account2.balance + 300;

System.out.println("Account " +
        + " has " + account1.bal
        + " " + account1.currenc
System.out.println("Account " +
        + " has " + account2.bal
        + " " + account2.currenc
```

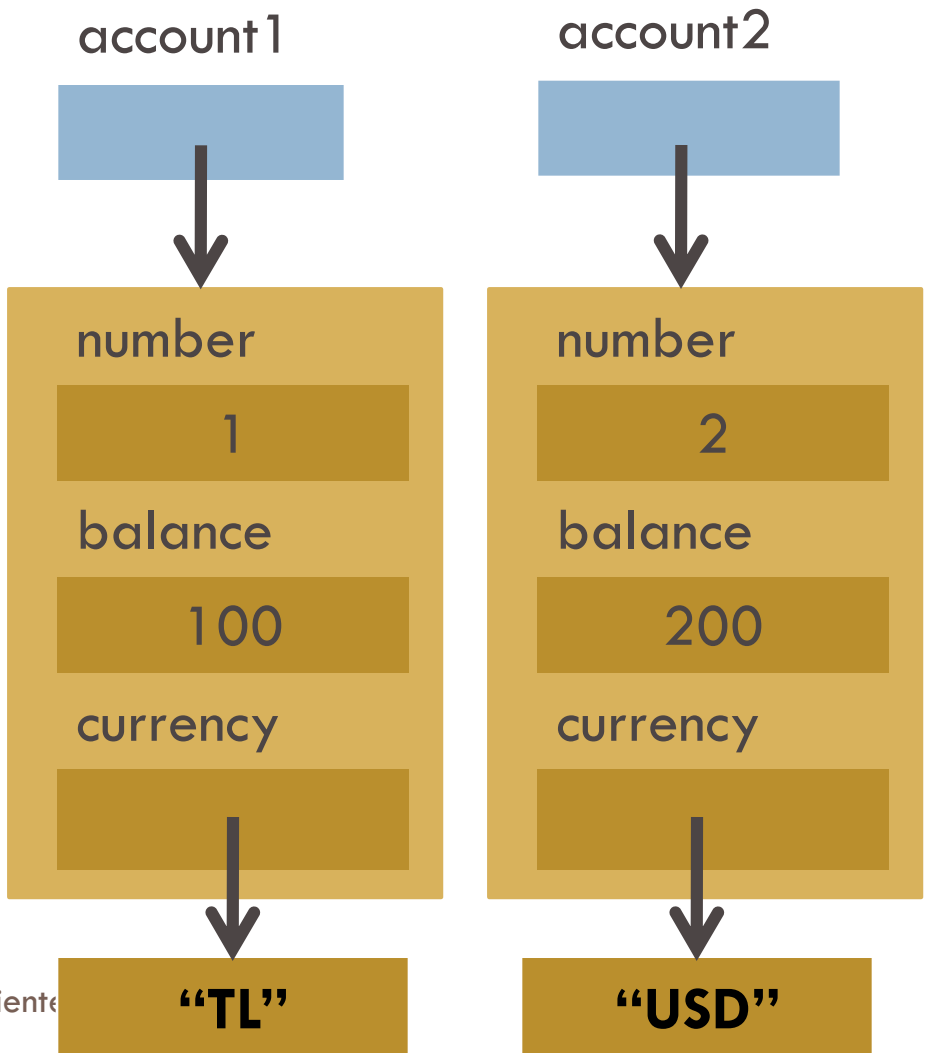
Javadoc | Declaration | Console

<terminated> AccountTest (1) [Java Application] C:\

```
Account 1 has 100.0 TL.
Account 2 has 200.0 USD.
Account 1 has 150.0 TL.
Account 2 has 500.0 USD.
```

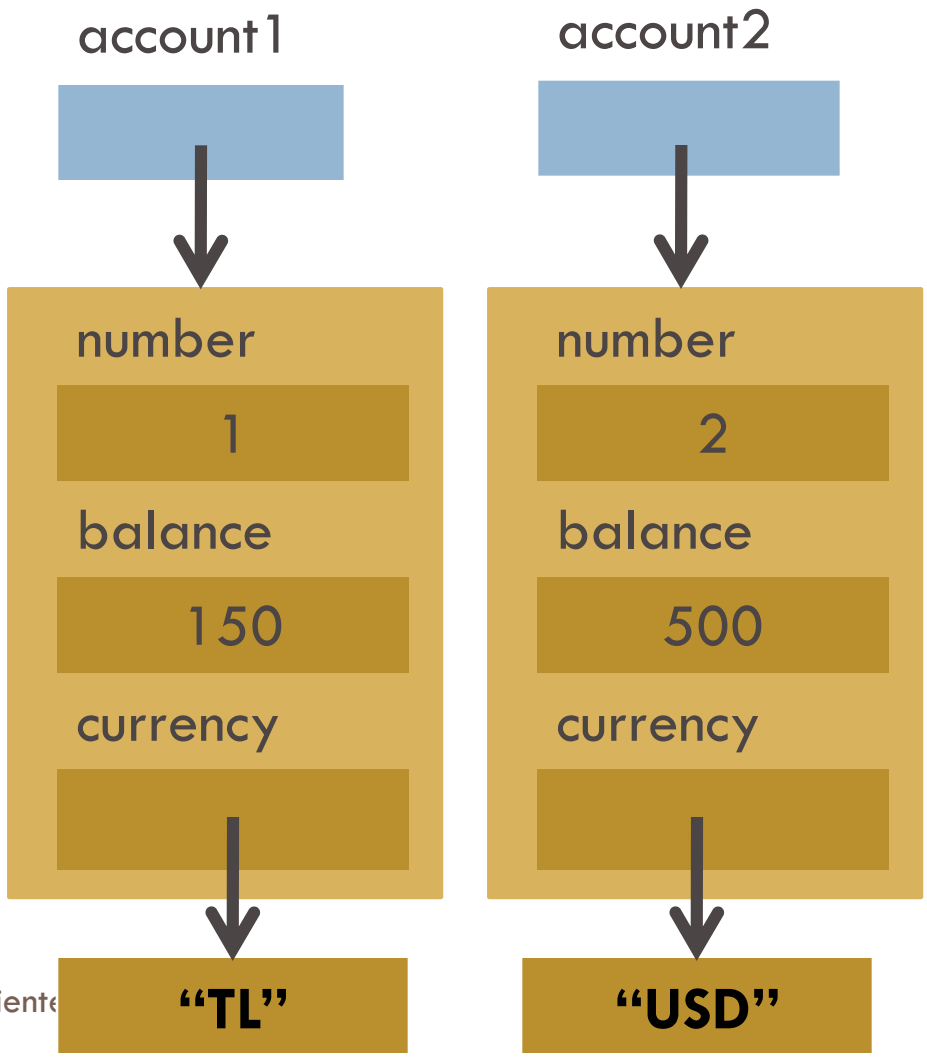
Change in memory…

- Before the deposit:



Change in memory…

- After the deposit:



Object Functionality

- Depositing money to an account is actually a functionality of an account.
  - All accounts can be deposited some amounts of money
- How can we make depositing money a functionality of account object?

Object Functionality

- Depositing money to an account is actually a functionality of an account.
  - All accounts can be deposited some amounts of money
- How can we make depositing money a functionality of account object?
  - By defining it as a **member function**…

Bank Account – version 2


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

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

deposit member function

- Before **deposit** member function

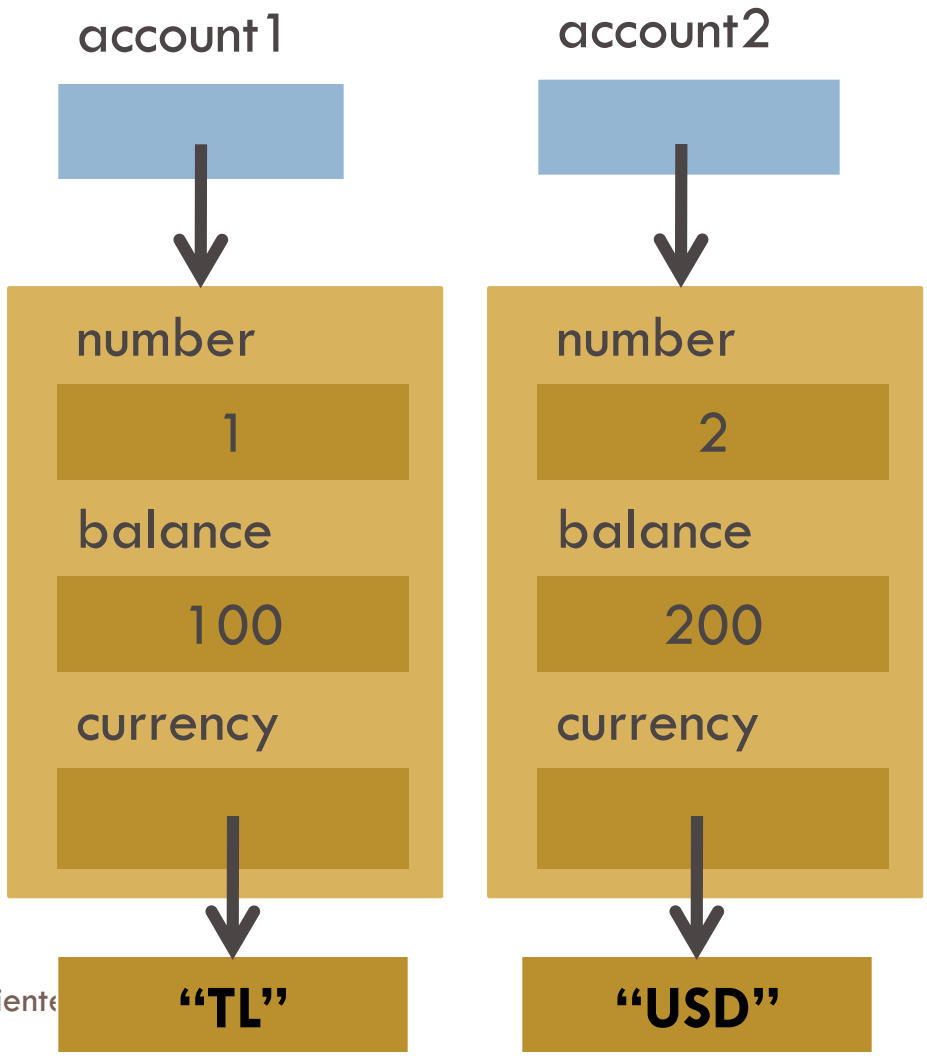
```
// Deposit 50TL into account 1
account1.balance = account1.balance + 50;

// Deposit 300 USD into account 2
account2.balance = account2.balance + 300;
```

- After **deposit** member function

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

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

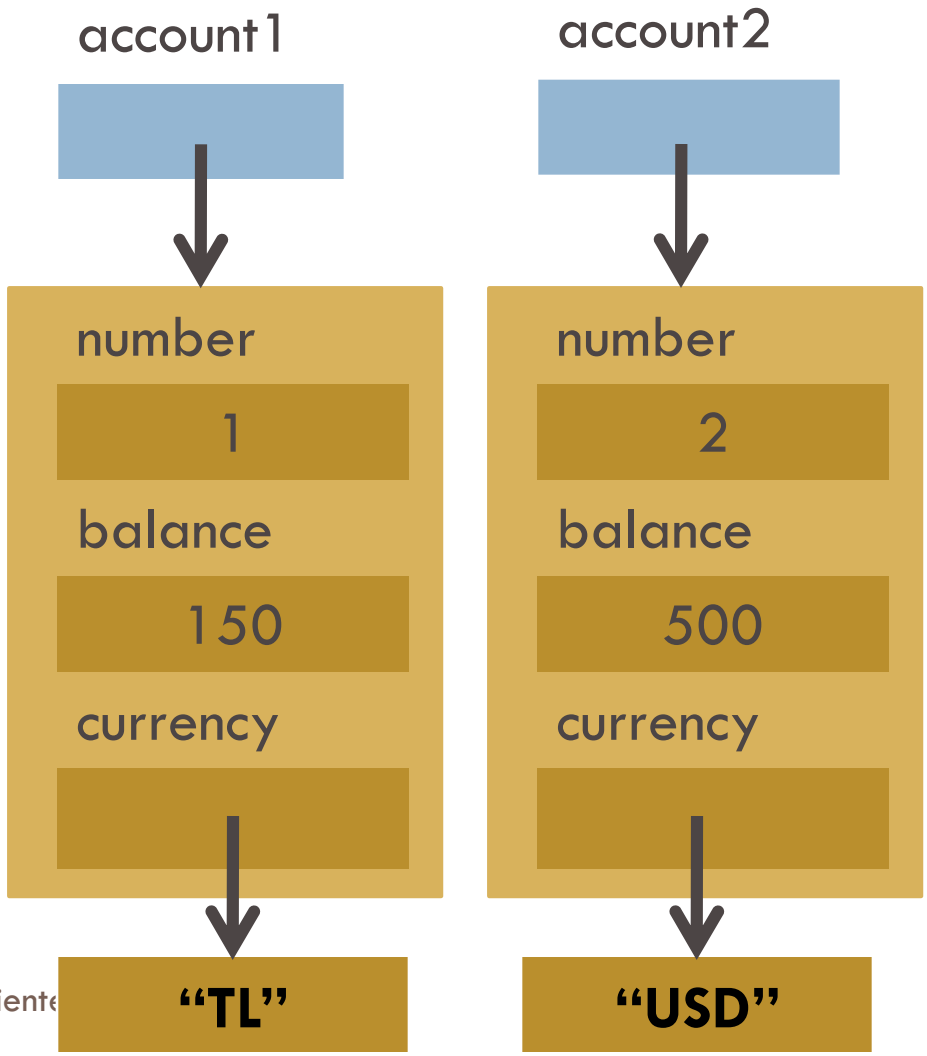
Change in memory… (same as before)

- Before the deposit:



Change in memory… (same as before)

- After the deposit:



Class

- We have written our first class!

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

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

Class

- We have written our first class!

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

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

member variables or instance variables

member functions

From last lecture…

- Our proposed programs need to **match to the problem** we are trying to solve
  - In the problem, what are the real world objects?
    - what kind of information do they hold? (attributes)
    - what kind of functionalities they have? (behavior)
  - Solve the problem in terms of these objects
    - Objects in the real world ~ Objects in our programs
    - Low representational gap
  - **The object oriented programming**

Source: "Core: Defining Classes and Creating Objects" Lecture Notes from UCSD Course on OOP in Java

Class

- We have written our first class!

attributes

member variables or instance variables

behaviors

member functions

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

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

Objects

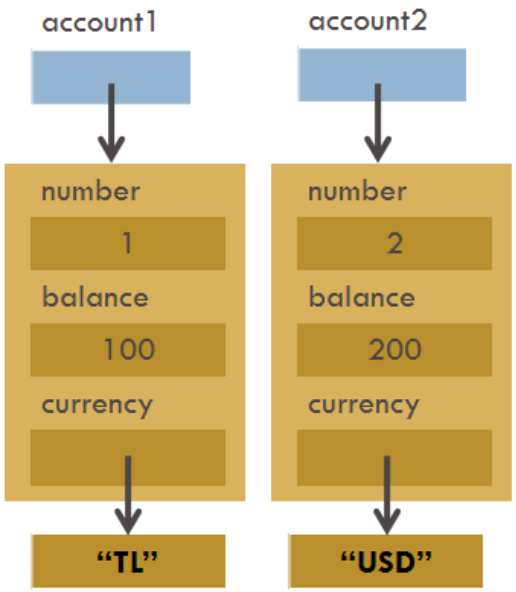
- We have also used this class to create objects!

```
public class AccountTest {

    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";

        System.out.println("Account " + account1.number
                + " has " + account1.balance
                + " " + account1.currency + ".");
        System.out.println("Account " + account2.number
                + " has " + account2.balance
                + " " + account2.currency + ".");
```

Objects

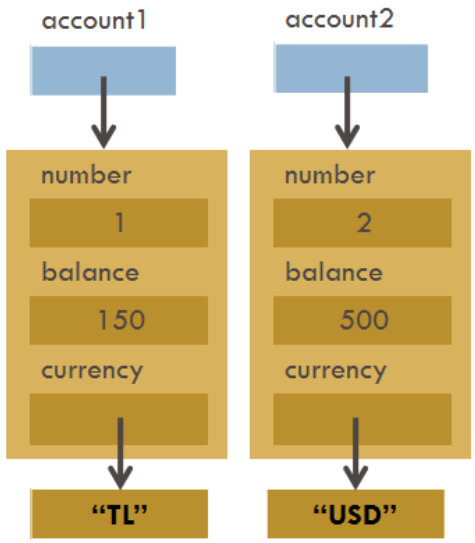
- We have also used this class to create objects!

```
public class AccountTest {

    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";

        System.out.println("Account " + account1.number
                + " has " + account1.balance
                + " " + account1.currency + ".");
        System.out.println("Account " + account2.number
                + " has " + account2.balance
                + " " + account2.currency + ".");
```



Objects

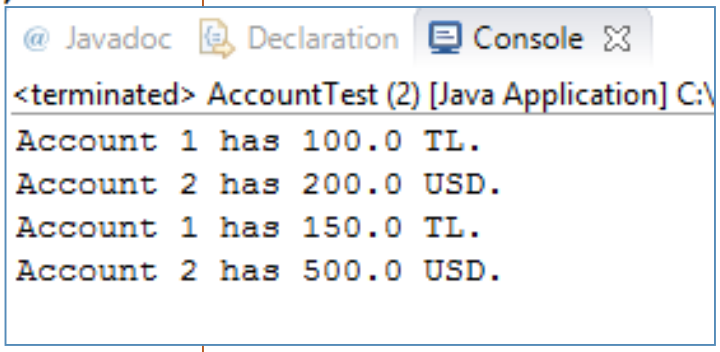
- We have also used this class to creat




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

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

        System.out.println("Account " + account1.number
                + " has " + account1.balance
                + " " + account1.currency + ".");
        System.out.println("Account " + account2.number
                + " has " + account2.balance
                + " " + account2.currency + ".");

    }
}
```

Program Output


```
System.out.println("Account " + account1.number
        + " has " + account1.balance
        + " " + account1.currency + ".");
System.out.println("Account " + account2.number
        + " has " + account2.balance
        + " " + account2.currency + ".");

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

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

System.out.println("Account " + account1.number
        + " has " + account1.balance
        + " " + account1.currency + ".");
System.out.println("Account " + account2.number
        + " has " + account2.balance
        + " " + account2.currency + ".");
```



Report Account Information


```
System.out.println("Account " + account1.number
        + " has " + account1.balance
        + " " + account1.currency + ".");
System.out.println("Account " + account2.number
        + " has " + account2.balance
        + " " + account2.currency + ".");

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

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

System.out.println("Account " + account1.number
        + " has " + account1.balance
        + " " + account1.currency + ".");
System.out.println("Account " + account2.number
        + " has " + account2.balance
        + " " + account2.currency + ".");
```

Report Account Information


```
System.out.println("Account " + account1.number
        + " has " + account1.balance
        + " " + account1.currency + ".");
System.out.println("Account " + account2.number
        + " has " + account2.balance
        + " " + account2.currency + ".");

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

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

System.out.println("Account " + account1.number
        + " has " + account1.balance
        + " " + account1.currency + ".");
System.out.println("Account " + account2.number
        + " has " + account2.balance
        + " " + account2.currency + ".");
```

How can we fix this?

Report Account Information

- Reporting its information can be a functionality of accounts.
- **report** member function!

Bank Account – version 3

- **report** member function!

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

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

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

Bank Account – version 3

- **report** member function!

```
System.out.println("Account " + account1.number
        + " has " + account1.balance
        + " " + account1.currency + ".");
System.out.println("Account " + account2.number
        + " has " + account2.balance
        + " " + account2.currency + ".");
```

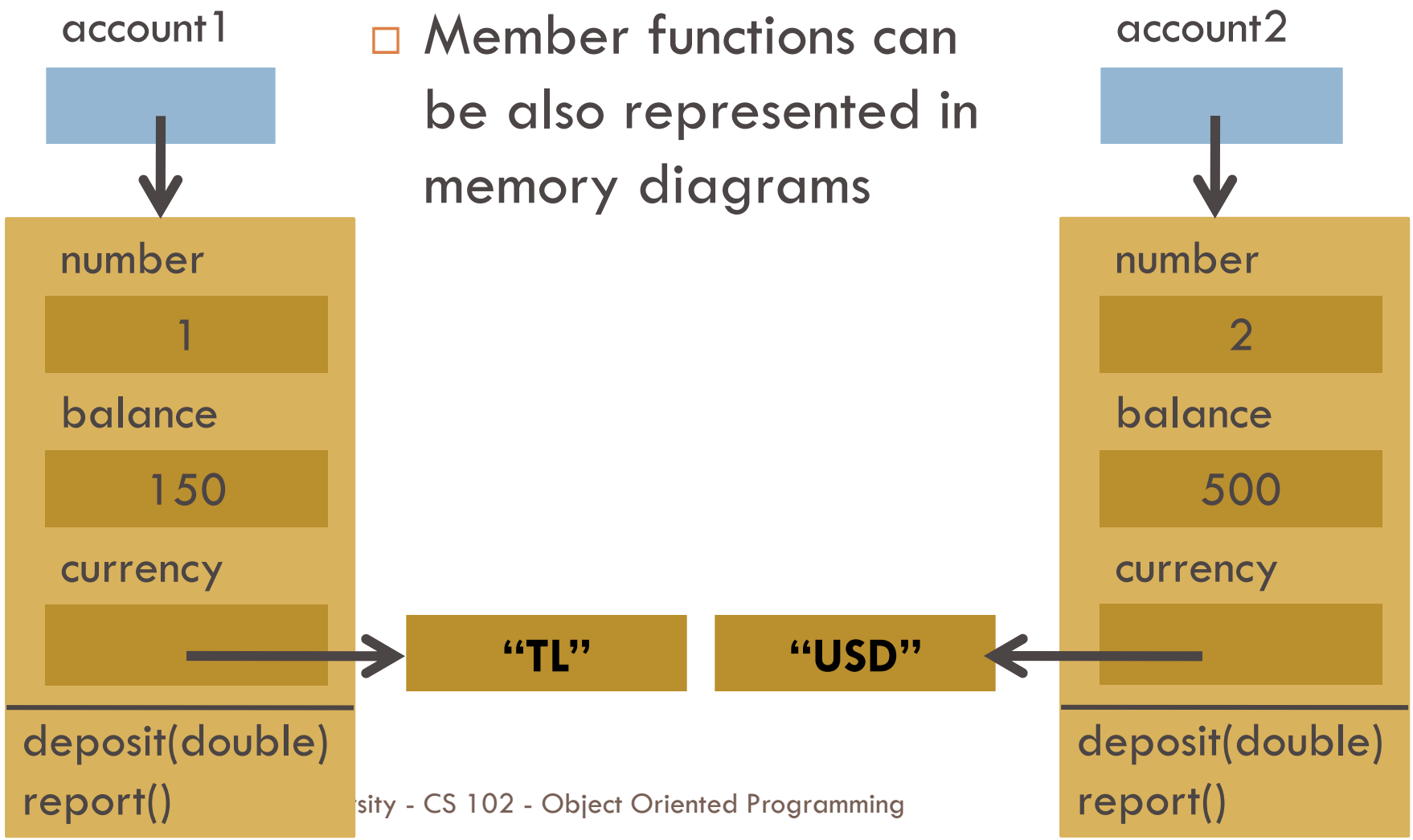
```
    public void report() {
        System.out.println("Account " + number
                + " has " + balance
                + " " + currency + ".");
    }
}
```

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();
}
```

Member Functions

- Member functions can be also represented in memory diagrams



Exercise: What is the output?

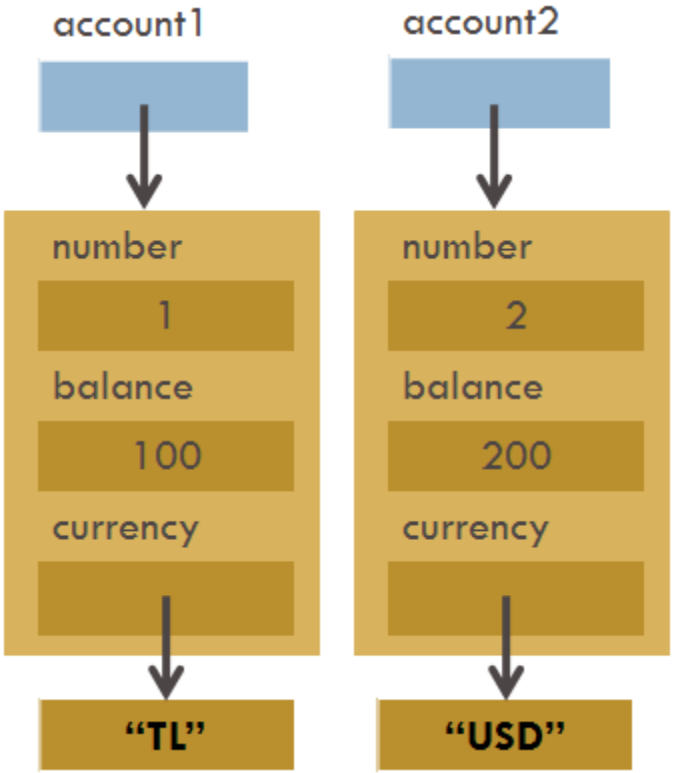

```
public static void main(String[] args) {
    Account account1 = new Account();
    account1.number = 1;
    account1.balance = 100;
    account1.currency = "TL";

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

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

    account1 = account2;

    // Deposit 50TL into account 1
    account1.deposit(50);
    // Deposit 300 USD into account 2
    account2.deposit(300);

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

Exercise: What is the output?


```
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();

    account1 = account2;

    // Deposit 50TL into account 1
    account1.deposit(50);
    // Deposit 300 USD into account 2
    account2.deposit(300);

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

Exercise (Bank Account – version 4)

```
62  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();

        account1 = account2;

        // Deposit 50TL into account 1
        account1.deposit(50);
        // Deposit 300 USD into account 2
        account2.deposit(300);

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



Exercise

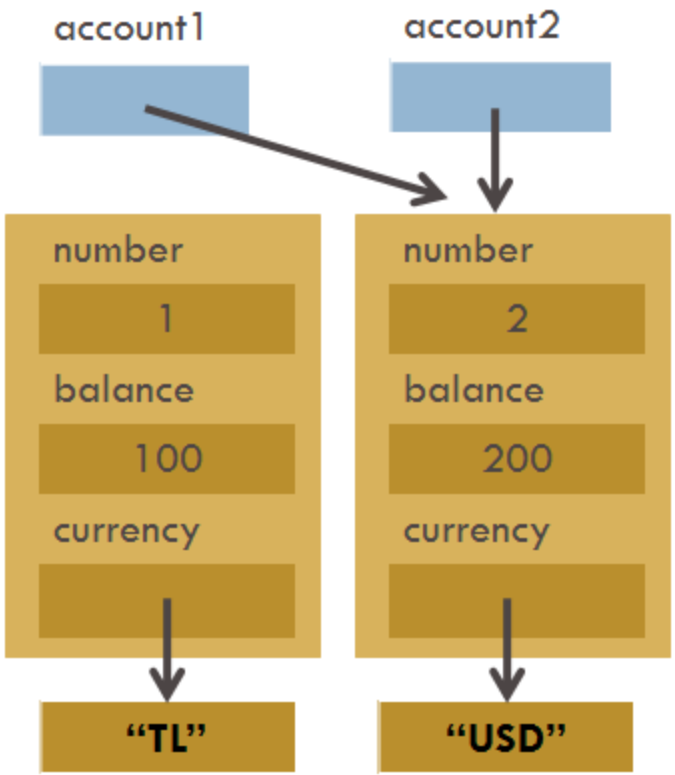
```
63 public static void main(String[] args) {
       Account account1 = new Account();
       account1.number = 1;
       account1.balance = 100;
       account1.currency = "TL";

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

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

       account1 = account2;
```

```
       // Deposit 50TL into account 1
       account1.deposit(50);
       // Deposit 300 USD into account 2
       account2.deposit(300);

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

Exercise

```
64 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();

       account1 = account2;
```

```
       // Deposit 50TL into account 1
       account1.deposit(50);
       // Deposit 300 USD into account 2
       account2.deposit(300);

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

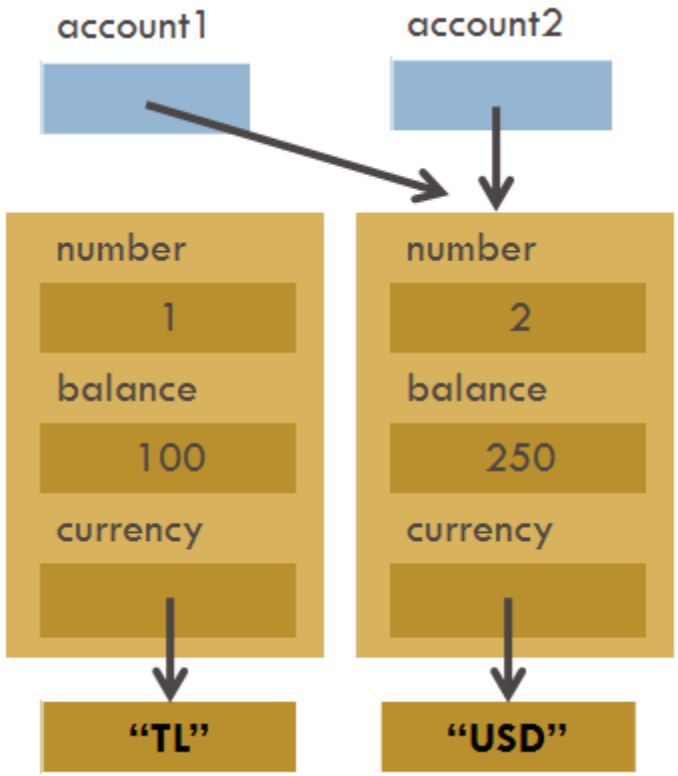


Exercise

```
65  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();

        account1 = account2;

        // Deposit 50TL into account 1
        account1.deposit(50);
        // Deposit 300 USD into account 2
        account2.deposit(300);

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

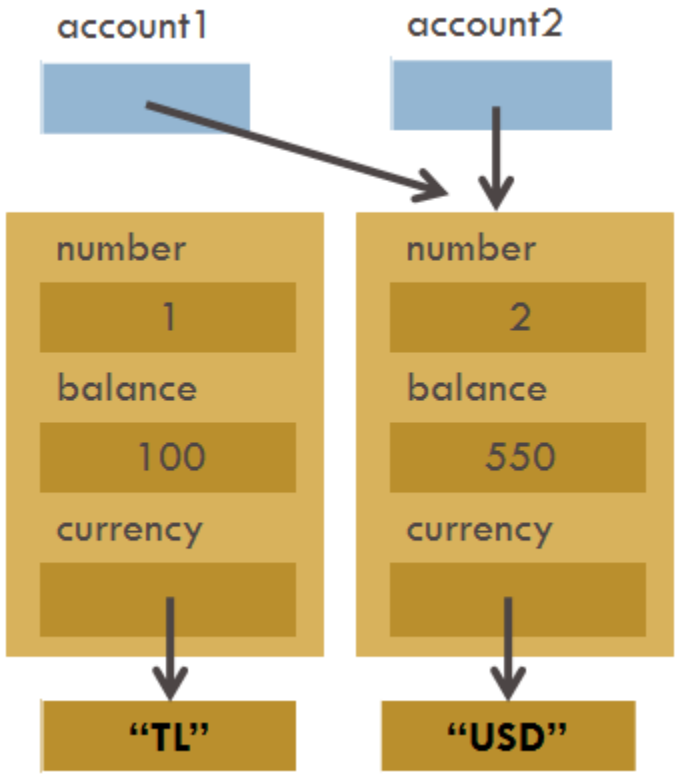
Exercise

```
66 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();

       account1 = account2;

       // Deposit 50TL into account 1
       account1.deposit(50);
       // Deposit 300 USD into account 2
       account2.deposit(300);

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



Exercise

```
67 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();

       account1 = account2;

       // Deposit 50TL into account 1
       account1.deposit(50);
       // Deposit 300 USD into account 2
       account2.deposit(300);

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

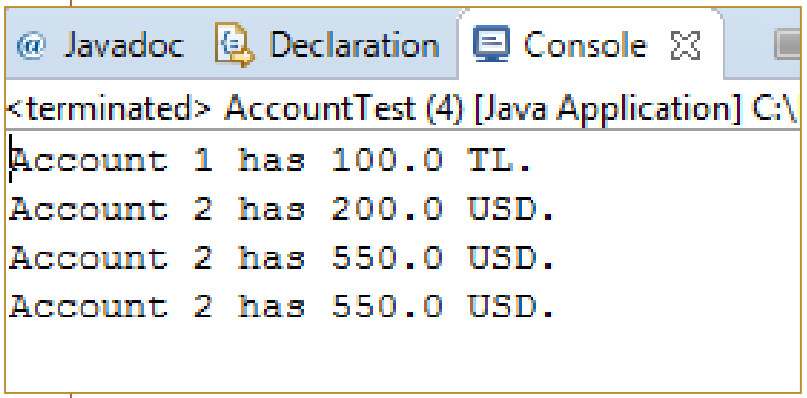


Exercise: What will be the output?

```
68 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();

       account1 = account2;

       // Deposit 50TL into account 1
       account1.deposit(50);
       // Deposit 300 USD into account 2
       account2.deposit(300);

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

Exercise: What will be the output?

```
69  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();

        account1 = account2;

        // Deposit 50TL into account 1
        account1.deposit(50);
        // Deposit 300 USD into account 2
        account2.deposit(300);

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



Exercise


```
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();

    account1 = account2;
    account2 = account1;

    // Deposit 50TL into account 1
    account1.deposit(50);
    // Deposit 300 USD into account 2
    account2.deposit(300);

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

Any Questions ?