Embedded System Practice Lab7

2016311821 한승하

<Exercise 1>

```
class BankAccount {
        double balance;
        public BankAccount(double balance){
                if(0>balance){
                        IllegalArgumentException exception = new IllegalArgumentException("Unavailable balance.");
                        throw exception;
                }
                this.balance = balance;
                System.out.printf("You have %.1f of balance\n",balance);
        }
        public void withdraw (double amount){
                if(0>amount){
                        IllegalArgumentException exception = new IllegalArgumentException("Amount out of bound.");
                        throw exception;
                }
                else if(amount > balance){
                        IllegalArgumentException exception = new IllegalArgumentException("Amount exceeds balance.");
                        throw exception;
                }
                balance = balance - amount;
                System.out.printf("Withdrwing %.1f amout.\nYou have %.1f of balance left\n",amount,balance);
        }
}

public class Main{
        public static void main(String[] args){
                try{
                        BankAccount acct = new BankAccount(-10);
                } catch (IllegalArgumentException ex){
                        ex.printStackTrace();
                }//Negative Balance
                BankAccount acct = new BankAccount(100); //Normal Balance
                try{
                        acct.withdraw(200);
                }catch (IllegalArgumentException ex){
                        ex.printStackTrace();
                }//withdraw amout exceeding balance
                try{
                        acct.withdraw(-200);
                }catch (IllegalArgumentException ex){
                        ex.printStackTrace();
                }//withdraw negative amount
                try{
                        acct.withdraw(50);
                }catch (IllegalArgumentException ex){
                        ex.printStackTrace();
                }//Normal withdraw
        }
}
```

위는 Exercise1의 소스코드입니다.

Initial Balance가 음수일 경우, withdraw의 금액이 잘못되었을 경우를 각각 if문을 사용하여 exception을 날려주었고, catch로 처리할 수 있도록 하였습니다.

아래는 해당 코드를 실행했을 때의 결과입니다.

```
han@han:~/Java/lab2$ java Main
java.lang.IllegalArgumentException: Unavailable balance.
        at BankAccount.<init>(Main.java:5)
        at Main.main(Main.java:28)
You have 100.0 of balance
java.lang.IllegalArgumentException: Amount exceeds balance.
        at BankAccount.withdraw(Main.java:17)
        at Main.main(Main.java:34)
java.lang.IllegalArgumentException: Amount out of bound.
        at BankAccount.withdraw(Main.java:13)
        at Main.main(Main.java:39)
Withdrwing 50.0 amout.
You have 50.0 of balance left
```

<Source Code>

```
class BankAccount {
	double balance;
	public BankAccount(double balance){
		if(0>balance){
			IllegalArgumentException exception = new IllegalArgumentException("Unavailable balance.");
			throw exception;
		}
		this.balance = balance;
		System.out.printf("You have %.1f of balance₩n",balance);
	}
	public void withdraw (double amount){
		if(0>amount){
			IllegalArgumentException exception = new IllegalArgumentException("Amount out of bound.");
			throw exception;
		}
		else if(amount > balance){
			IllegalArgumentException exception = new IllegalArgumentException("Amount exceeds balance.");
			throw exception;
		}
		balance = balance - amount;
		System.out.printf("Withdrwing %.1f amout.₩nYou have %.1f of balance left₩n",amount,balance);
	}
}

public class Main{
	public static void main(String[] args){
		try{
			BankAccount acct = new BankAccount(-10);
		} catch (IllegalArgumentException ex){
			ex.printStackTrace();
		}//Negative Balance
		BankAccount acct = new BankAccount(100); //Normal Balance
		try{
			acct.withdraw(200);
```

```
                }catch (IllegalArgumentException ex){
                        ex.printStackTrace();
                }//withdraw amout exceeding balance
                try{
                        acct.withdraw(-200);
                }catch (IllegalArgumentException ex){
                        ex.printStackTrace();
                }//withdraw negative amount
                try{
                        acct.withdraw(50);
                }catch (IllegalArgumentException ex){
                        ex.printStackTrace();
                }//Normal withdraw
        }
}
```

<Example2>

```java
import java.util.*;
import java.util.ArrayList;

class Bank{
	ArrayList<Account> Accounts = new ArrayList<Account>();
	public Bank(double balance){
		addAccount(balance);
	}
	public void addAccount(double initialBalance){
		Accounts.add(new Account(initialBalance));
		System.out.printf("Account[%d] with %.1f balance added\n",(Accounts.size()-1),initialBalance);
	}
	public void deposit(int account, double amount){
		Account temp = (Account)Accounts.get(account);
		temp.deposit(amount);
		Accounts.set(account,temp);
		System.out.printf("Depoist %.1f amount to Account[%d], Balance: %.1f\n",amount,account,temp.getBalance());
	}
	public void withdraw(int account, double amount){
		Account temp = (Account)Accounts.get(account);
		temp.withdraw(amount);
		Accounts.set(account,temp);
		System.out.printf("Withdraw %.1f amount to Account[%d], Balance: %.1f\n",amount,account,temp.getBalance());
	}
	public double getBalance(int account){
		Account temp = (Account)Accounts.get(account);
		return temp.getBalance();
	}
}

class Account{
	double balance;
	public Account(double initialBalance){
		this.balance = initialBalance;
	}
	public void deposit(double amount){
		this.balance += amount;
	}
	public void withdraw(double amount){
		this.balance -= amount;
	}
	public double getBalance(){
		return this.balance;
	}
}

public class Main{
	public static void main(String[] args){
		Bank accounts = new Bank(1000);
		accounts.addAccount(500);
		accounts.addAccount(300);
		accounts.addAccount(0);
		accounts.deposit(3,200);
		System.out.printf("Account[%d] has balance of %.1f\n",3,accounts.getBalance(3));
		accounts.withdraw(2,100);
		System.out.printf("Account[%d] has balance of %.1f\n",2,accounts.getBalance(2));
	}
}
```

위는 Example2의 Source Code입니다. Arraylist를 이용하여 각 account가 새로운 index에 추가될 수 있게 하였고, 각 함수들에 대한 test를 진행하였습니다. 아래는 실행 결과입니다.

```
han@han:~/Java/lab2-1$ java Main
Account[0] with 1000.0 balance added
Account[1] with 500.0 balance added
Account[2] with 300.0 balance added
Account[3] with 0.0 balance added
Depoist 200.0 amount to Account[3], Balance: 200.0
Account[3] has balance of 200.0
Withdraw 100.0 amount to Account[2], Balance: 200.0
Account[2] has balance of 200.0
```

<Source Code>

```
import java.util.*;
import java.util.ArrayList;

class Bank{
	ArrayList<Account> Accounts = new ArrayList<Account>();
	public Bank(double balance){
		addAccount(balance);
	}
	public void addAccount(double initialBalance){
		Accounts.add(new Account(initialBalance));
		System.out.printf("Account[%d] with %.1f balance added₩n",(Accounts.size()-
1),initialBalance);
	}
	public void deposit(int account, double amount){
		Account temp = (Account)Accounts.get(account);
		temp.deposit(amount);
		Accounts.set(account,temp);
		System.out.printf("Depoist %.1f amount to Account[%d],
Balance: %.1f₩n",amount,account,temp.getBalance());
	}
	public void withdraw(int account, double amount){
		Account temp = (Account)Accounts.get(account);
		temp.withdraw(amount);
		Accounts.set(account,temp);
		System.out.printf("Withdraw %.1f amount to Account[%d],
Balance: %.1f₩n",amount,account,temp.getBalance());
	}
	public double getBalance(int account){
		Account temp = (Account)Accounts.get(account);
		return temp.getBalance();
	}
}

class Account{
	double balance;
	public Account(double initialBalance){
		this.balance = initialBalance;
	}
```

```
	public void deposit(double amount){
		this.balance += amount;
	}
	public void withdraw(double amount){
		this.balance -= amount;
	}
	public double getBalance(){
		return this.balance;
	}
}

public class Main{
	public static void main(String[] args){
		Bank accounts = new Bank(1000);
		accounts.addAccount(500);
		accounts.addAccount(300);
		accounts.addAccount(0);
		accounts.deposit(3,200);
		System.out.printf("Account[%d] has balance of %.1f₩n",3,accounts.getBalance(3));
		accounts.withdraw(2,100);
		System.out.printf("Account[%d] has balance of %.1f₩n",2,accounts.getBalance(2));
	}
}
```