Listing 1: Uebung2_Gyakorlas/src/AccountsTest.java

```java
import javax.security.auth.login.AccountException;

public class AccountsTest {
    public static void main(String[] args) {
        testGiroAccount();
        testSavingsAccount();
        testCreditAccount();
    }


    public static void testGiroAccount() {
        System.out.println("Giro account test...");
        GiroAccount giroAccount = new GiroAccount(1000);
        try {
            giroAccount.deposit(500);
            giroAccount.withdraw(1500);
            System.out.println("Giro account is correct and ready to use.");
        } catch (AccountException ex) {
            System.out.println("Giro account failed: " + ex.getMessage() );
        }

    }

    public static void testSavingsAccount() {
        System.out.println("Saving account test...");
        SavingsAccount savingsAccount = new SavingsAccount();
        try {
            savingsAccount.deposit(1000);
            savingsAccount.withdraw(500);
            savingsAccount.withdraw(600);
            System.out.println("Saving account is correct and ready to use.");
        } catch (AccountException ex) {
            System.out.println("Saving account failed: " + ex.getMessage());
        }
    }

    public static void testCreditAccount() {
        System.out.println("Credit account test...");
        CreditAccount creditAccount = new CreditAccount(-5000);
        try {
            creditAccount.deposit(5000);
            creditAccount.deposit(100);
            creditAccount.withdraw(100);
            System.out.println("Credit account is correct and ready to use.");
        } catch (AccountException ex) {
            System.out.println("Credit account failed: " + ex.getMessage());
        }
    }

    }
```

Listing 2: Uebung2_Gyakorlas/src/GiroAccount.java

```java
import javax.security.auth.login.AccountException;

public class GiroAccount extends Account {
    private int overdraftLimit;

    public GiroAccount(int overdraftLimit) {
        super();
        this.overdraftLimit = overdraftLimit;
    }

    @Override
    public void withdraw(int amount) throws AccountException {
        if (amount > balance + overdraftLimit) {
            throw new AccountException("Overdraft limit exceeded. Transaction 
                failed.");
        } else {
            super.withdraw(amount);

        }
    }
}
```

Listing 3: Uebung2_Gyakorlas/src/Account.java

```java
import javax.security.auth.login.AccountException;

public abstract class Account {
    private static int nextId = 123;
    private final int accountNumber;
    protected int balance;

    public Account() {
        this.accountNumber = nextId;
        nextId += 1;
        this.balance = 0;
    }

    public int getAccountNumber() {
        return accountNumber;
    }

    public int getBalance() {
        return balance;
    }

    public void deposit(int amount) {
        balance += amount;
    }

    public void withdraw(int amount) throws AccountException {
        if(amount > balance) {
            throw new AccountException("Insufficient funds");
        } else {
            balance -= amount;
        }
    }

    public String toString() {
        return "Account Number: " + accountNumber + ", Balance: " + balance;
    }
}
```

Listing 4: Uebung2_Gyakorlas/src/CreditAccount.java

```java
import javax.security.auth.login.AccountException;

public class CreditAccount extends Account{
    public CreditAccount(int currentBalance) {
        super();
        this.balance = currentBalance;
    }

    @Override
    public void deposit(int amount) {
        if (amount > -1 * balance ) {
                balance = 0;
            } else {
                super.deposit(amount);
            }
        }

    @Override
    public void withdraw(int amount) throws AccountException {
        throw new AccountException("Cannot withdraw money from a credit acount. 
            Transaction failed");

    }


}
```

Listing 5: Uebung2_Gyakorlas/src/SavingsAccount.java

```java
import javax.security.auth.login.AccountException;

public class SavingsAccount extends Account {

    @Override
    public void withdraw(int amount) throws AccountException {
        if (amount > balance) {
            throw new AccountException("Cannot overdraft on a savings account. 
                Transaction failed.");
        } else {
            super.withdraw(amount);
        }
    }
}
```

Listing 6: Uebung2_Gyakorlas/src/Out.java

```java


import java.io.*;

/**
 * Simple output to the console and to files.
 * <p>Copyright (c) 2005 Hanspeter Moessenboeck, University of Linz</p>
 *
 * <p>This class is free software; you can redistribute it and/or modify it
 * under the terms of the GNU General Public License as published by the
 * Free Software Foundation; either version 2, or (at your option) any
 * later version.</p>
 *
 * <p>This class is distributed in the hope that it will be useful, but
 * WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY
 * or FITNESS FOR A PARTICULAR PURPOSE.  See the <a
    href="http://www.gnu.org/copyleft/gpl.html">
 * GNU General Public License</a> for more details.</p>
 * <hr>
 *
 * <p>This class allows printing formatted data either to the console
 * or to a file. It is intended to be used in an introductory
 * programming course when classes, packages and exceptions are unknown
 * at the beginning. To use it, simply copy Out.class into the
 * current directory. </p>
 *
 * <p>All output goes to the current output file, which is initially
 * the console. Opening a file with open() makes it the new current
 * output file. Closing a file with close() switches back to the previous
 * output file.</p>
 */
@SuppressWarnings("unused")
public class Out {

    private static final PrintStream[] stack;

    private static PrintStream out;
    private static int         sp;
    private static boolean     done;

    /**
     * Return true if the previous Out operation was
     * successful, otherwise return false.
     */
    public static boolean done() {
        return done && !out.checkError();
    }

    /**
     * Print the boolean value b either as "true" or "false".
     */
    public static void print(boolean b) {
        out.print(b);
    }

    /**
     * Print the character value c.
     */
    public static void print(char s) {
        out.print(s);
    }

    /**
     * Print the integer value i.
     */
    public static void print(int i) {
        out.print(i);
    }

    /**
     * Print the long value l.
     */
    public static void print(long l) {
        out.print(l);
    }

    /**
     * Print the float value f.
     */
    public static void print(float f) {
        out.print(f);
    }

    /**
     * Print the double value d.
     */
    public static void print(double d) {
        out.print(d);
    }

    /**
     * Print the character array a.
     */
    public static void print(char[] a) {
```

```java
        out.print(a);
    }

    /**
     * Print the String s.
     */
    public static void print(String s) {
        out.print(s);
    }

    /**
     * Print the Object o as resulting from String.valueOf(o).
     */
    public static void print(Object o) {
        out.print(o);
    }

    /**
     * Terminate the current line by writing a line separator string.
     * On Windows this is the character sequence '\r' and '\n'
     */
    public static void println() {
        out.println();
    }

    /**
     * Print the boolean value b and terminate the line.
     */
    public static void println(boolean b) {
        out.println(b);
    }

    /**
     * Print the character value c and terminate the line.
     */
    public static void println(char s) {
        out.println(s);
    }

    /**
     * Print the integer value i and terminate the line.
     */
    public static void println(int i) {
        out.println(i);
    }

    /**
     * Print the long value l and terminate the line.
     */
    public static void println(long l) {
        out.println(l);
    }

    /**
     * Print the float value f and terminate the line.
     */
    public static void println(float f) {
        out.println(f);
    }

    /**
     * Print the double value d and terminate the line.
     */
    public static void println(double d) {
        out.println(d);
    }

    /**
     * Print the character array a and terminate the line.
     */
    public static void println(char[] a) {
        out.println(a);
    }

    /**
     * Print the String s and terminate the line.
     */
    public static void println(String s) {
        out.println(s);
    }

    /**
     * Print the Object o as resulting from String.valueOf(o)
     * and terminate the line.
     */
    public static void println(Object o) {
        out.println(o);
    }

    /**
     * Open the file with the name fn as the current output file.
     * All subsequent output goes to this file until it is closed.
     * The old output file will be restored when the new output file is closed.
     */
    public static void open(String fn) {
        try {
            PrintStream s = new PrintStream(new FileOutputStream(fn));
            stack[sp++] = out;
            out = s;
        } catch (Exception e) {
            done = false;
        }
    }

    /**
     * Close the current output file.
     * The previous output file is restored and becomes the current output file.
     */
    public static void close() {
        out.flush();
        out.close();
        if (sp > 0) {
            out = stack[--sp];
        }
    }

    static { // initializer
        done = true;
        out = System.out;
        stack = new PrintStream[8];
        sp = 0;
    }

}
```

Listing 7: Uebung2_Gyakorlas/src/AccountException.java

```
public class AccountException extends Exception {
    public AccountException(String message) {
        super(message);
    }
}
```