```java
package oop1_dat1_u;

//import java.util.ArrayList;
//import java.util.Iterator;

/**
 *
 * @author Felix Rohrer <felix.rohrer@stud.hslu.ch>
 */
public class Bank
{

    private String name;
    //private ArrayList<Konto> kontoList;
    private MyLinkedList<Konto> kontoList;

    public Bank()
    {
        this("Bank");
    }

    public Bank(String newName)
    {
        name = newName;
        //kontoList = new ArrayList<Konto>();
        kontoList = new MyLinkedList<Konto>();
        System.out.println("Bank: '" + newName + "' wurde erstellt.");
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args)
    {
        /*
         Konto myKonto = new Konto();
        myKonto.print();
        myKonto.payIn(100.0);
        myKonto.print();
        myKonto.payOut(50.0);
        myKonto.print();

        Konto myKonto2 = new Konto(10000.0, 5.5);
        myKonto2.print();
        myKonto2.payOut(1234.5);
        myKonto2.print();

        Spar mySpar = new Spar(1000.0, 7.5, 200.0);
        mySpar.print();
        mySpar.payOut(200.0);
        mySpar.print();
         */

        Bank myBank = new Bank("Meine Bank");
        myBank.openKonto();
        myBank.openKonto(20000.0, 2.5);
        myBank.openKontoRnd();
        myBank.openKontoRnd();
        myBank.openKontoRnd();
        myBank.openSpar();
        myBank.openSpar(100000.0, 5.0, 1000.0);
        myBank.openSparRnd();

        System.out.println("Anzahl Konto: " + myBank.noOfAccounts());
        myBank.printAccounts();
        myBank.printFund();
    }

    public void openKonto()
    {
        //kontoList.add(new Konto());
        kontoList.insert(new Konto());
    }

    public void openKontoRnd()
    {
        openKonto(Math.random()*1000, Math.random()*2);
    }

    public void openKonto(double newSaldo, double newRate)
    {
        kontoList.insert(new Konto(newSaldo, newRate));
    }

    public void openSpar()
    {
        kontoList.insert(new Spar());
    }
    public void openSparRnd()
    {
        openSpar(Math.random()*5000, Math.random()*5, Math.random()*1000);
    }

    public void openSpar(double newSaldo, double newRate, double newMaxOut)
    {
        kontoList.insert(new Spar(newSaldo, newRate, newMaxOut));
    }

    public int noOfAccounts()
    {
        return kontoList.size();
    }

    public void printAccounts()
    {
        /*
        Iterator<Konto> itr;
        itr = kontoList.iterator();
        while (itr.hasNext()) {
            itr.next().print();
        }
         */
        MyListNode actualNode = kontoList.getHead();
        System.out.println("----------");
        while (actualNode != null) {
            Konto k = (Konto)actualNode.getData();
            k.print();
            actualNode = actualNode.getNext();
        }
    }

    public void printFund()
    {
        /*
         int sumFund = 0;
        Iterator<Konto> itr;
        itr = kontoList.iterator();
        while (itr.hasNext()) {
            sumFund += itr.next().getSaldo();
        }
        System.out.println("Summe aller Flüssige Mittel: " + sumFund);
        */
        MyListNode actualNode = kontoList.getHead();
        double sumFund = 0.0;
        while (actualNode != null) {
            Konto k = (Konto)actualNode.getData();
            sumFund += k.getSaldo();
            actualNode = actualNode.getNext();
        }
        //System.out.println("Summe aller flüssigen Mittel: " + sumFund);
        System.out.format("Summe aller flüssigen Mittel: %.2f%n", sumFund);
    }
}
```

```java
package oop1_dat1_u;

/**
 *
 * @author Felix Rohrer <felix.rohrer@stud.hslu.ch>
 */
public class Konto
{

    private static int count = 1;
    private int no;
    private double saldo;
    private double rate;

    public Konto()
    {
        no = count;
        count ++;
        saldo = 0.0;
        rate = 0.0;
    }

    public Konto(double newSaldo, double newRate)
    {
        this();
        saldo = newSaldo;
        rate = newRate;
    }

    public void payIn(double value)
    {
        if (value > 0.0) {
            saldo += value;
        }
    }

    public void payOut(double value)
    {
        if ((value > 0.0) && (value <= saldo)) {
            saldo -= value;
        }
    }

    public void print()
    {
        // System.out.println("Konto: " + no + ", Saldo: " + saldo + ", Zins: " + rate + "%");
        System.out.format("Konto: %d, Saldo: %.2f, Zins: %.2f%%n", no, saldo, rate);
    }

    protected double getSaldo()
    {
        return saldo;
    }

    protected double getRate()
    {
        return rate;
    }
    protected int getKontoNummer()
    {
        return no;
    }
}
```

```java
package oop1_dat1_u;

/**
 *
 * @author Felix Rohrer <felix.rohrer@stud.hslu.ch>
 */
public class MyLinkedList<T>
{

    private MyListNode<T> head; // Kopf bzw. Anfang der Liste
    private int numOfElements; // Anzahl Elemente in dieser Liste

    public MyLinkedList()
    {
        head = null;
        numOfElements = 0;
    }

    /**
     * Prüft, ob Liste leer ist.
     */
    public boolean isEmpty()
    {
        return (head == null);
    }

    /**
     * Fügt Objekt d am Anfang in die Liste ein.
     */
    public void insert(T d)
    {
        head = new MyListNode<T>(head, d);
        numOfElements++;
    }

    /**
     * Prüft, ob ein gleiches Objekt wie d bereits in der Liste enthalten ist.
     */
    public boolean isFound(T d)
    {
        MyListNode<T> actualNode = head;
        while ((actualNode != null) &&  ! d.equals(actualNode.getData())) {
            actualNode = actualNode.getNext();
        }
        if (actualNode == null) {
            return false;
        } else {
            return true;
        }
    }

    /**
     * Entfernt aus der Liste das erste Objekt gleich d.
     */
    public void remove(T d)
    {
        MyListNode<T> actualNode = head;
        MyListNode<T> prevNode = null;
        while ((actualNode != null) &&  ! d.equals(actualNode.getData())) {
            prevNode = actualNode;
            actualNode = prevNode.getNext();
        }
        // Liste wieder ordentlich zusammenhängen:
        if (actualNode != null) {
            if (actualNode == head) {
                head = actualNode.getNext();
            } else {
                prevNode.setNext(actualNode.getNext());
            }
            numOfElements--;
        }
    }

    /**
     * Gibt alle Objekte der Reihe nach zeilenweise auf die Konsole aus.
     */
    public void print()
    {
        MyListNode<T> actualNode = head;
        System.out.println("----------");
        while (actualNode != null) {
            System.out.println(actualNode.getData());
            actualNode = actualNode.getNext();
        }
    }

    public MyListNode<T> getHead()
    {
        return head;
    }

    public int size()
    {
        return numOfElements;
    }

    // Test
    public static void main_TestMyLinkedList(String[] args)
    {
        MyLinkedList<Konto> list = new MyLinkedList<Konto>();
        list.insert(new Konto(10000.0, 1.5));
        list.insert(new Konto(500.0, 0.7));
        list.insert(new Spar());
        list.print();
        list.remove(new Konto(500.0, 0.7));
        list.print();
    }
}
```

```
package oop1_dat1_u;

/**
 *
 * @author Felix Rohrer <felix.rohrer@stud.hslu.ch>
 */
public class MyListNode<T> // NEU: Parametrisierte Klasse mit Klassentyp T
{

    private T data; // Daten; jetzt vom vorgegebenen Klassentyp T!
    private MyListNode<T> next; // Referenz zum nächsten ListNode-Objekt

    public MyListNode(MyListNode<T> n, T d)
    {
        next = n;
        data = d;
    }

    public void setData(T d)
    {
        data = d;
    }

    public T getData()
    {
        return data;
    }

    public void setNext(MyListNode<T> n)
    {
        next = n;
    }

    public MyListNode<T> getNext()
    {
        return next;
    }
}
```

```
package oop1_dat1_u;

/**
 *
 * @author Felix Rohrer <felix.rohrer@stud.hslu.ch>
 */
public class Spar extends Konto
{

    private double maxOut;

    public Spar()
    {
        super();
        maxOut = 500.0;
    }

    public Spar(double newSaldo, double newRate, double newMaxOut)
    {
        super(newSaldo, newRate);
        maxOut = newMaxOut;
    }

    @Override
    public void payOut(double value)
    {
        if (value <= maxOut) {
            super.payOut(value);
        }
    }

    @Override
    public void print()
    {
        System.out.println("Spar-Konto: " + super.getKontoNummer() + ", Saldo: " +
super.getSaldo() + ", Zins: " + super.getRate() + "%");
    }
}
```