



ASSIGNMENT 10 – SOLUTION

GENERAL INFORMATION

This assignment is made public on Friday, April 11, 2003. The assignment is due on

Friday, April 25, 1 PM.

Hand in your assignment to the teaching assistant running your lab session.

The first page of your (written) assignment has to contain at least the following information:

- the course name (Introduction to Programming - Concepts and Tools)
- your name and your student number
- name and student number of the fellow student if you submit in pairs
- assignment number

Please staple your assignment!

You will get back the graded assignment one week after submission deadline.

QUESTIONS

1. (*Abstract Data Types*)

SOLUTION

Your answers to this question will be quite individual, so no solution is provided.

2. (*Linked Lists*)

SOLUTION

```
//IntList.java - a linked list of integers
```

```
/* The following methods are added:
```

```
* boolean moveForward()
* int howMany()
* void moveToTail()
* append(IntList)
* IntListElement find(int)
*/
```

```
class IntList {
```

```
    IntListElement head;
    IntListElement current;//previous;
```

```
    /* Note that there is no constructor. */
```

```
    /* New instance methods */
```

```
    boolean moveForward(){
        boolean moved = false;
        if(current == null){
```

```

        if(head != null){
            current = head;
            moved = true;
        }
    }
    else{
        if(current.next != null){
            current = current.next;
            moved = true;
        }
    }
    return moved;
}

int howMany(){
    int howMany = 0;

    IntListElement iterator = head;
    while(iterator != null){
        howMany++;
        iterator = iterator.next;
    }
    return howMany;
}

void moveToTail(){
    if(head == null)
        return;
    else{
        current = head;
        while(current.next != null){
            current = current.next;
        }
        return;
    }
}

IntListElement find(int key){
    IntListElement r = null;
    boolean found = false;

    IntListElement iterator = head;

    while(!found && iterator != null){
        if(iterator.data == key){
            found = true;
            r = iterator;
        }
        iterator = iterator.next;
    }
    return r;
}

/* Instance methods: */

int current(){return current.data;}

void moveToHead(){current = null;}

```

```

boolean hasNext(){
    if(current != null)
        return current.next != null;
    else
        return head != null;
}

int next(){
    if (current == null)
        current = head;
    else
        current = current.next;
    return current.data;
}

void insert(int value){
    IntListElement nle = new IntListElement(value);

    if(current != null){ // current points to a non-empty listmember
        IntListElement tmp = current.next;
        current.next = nle;
        nle.next = tmp;
        current = nle;
    }
    else if(head != null){// one element list
        current = nle;
        nle.next = head;
        head = current;
    }
    else {// empty list
        head = current = nle;
    }
    return;
}

public String toString(){
    String r = "";
    IntListElement cell = head;
    while(cell != null){
        r = r + cell.toString() + "\n";
        cell = cell.next;
    }
    return r;
}
}

```

A possible program testing some of the functions is the following:

```

class TestIntList{

    public static void main(String[] args){

        IntList il = new IntList();

        System.out.println("Empty list " + il.howMany());

        il.insert(1);
        il.insert(3);
    }
}

```

```
il.insert(5);

System.out.println("1,3,5 added");
System.out.println("Number of elements: " + il.howMany());

System.out.println("Current value: " + il.current());

System.out.println("Move to head");
il.moveToHead();
System.out.println("Next value: " + il.next());

System.out.println("Move to tail.");
il.moveToTail();
System.out.println("Current value: " + il.current());
System.out.println("Advance using moveForward ");
il.moveForward();
System.out.println("Current value: " + il.current());
}
}
```

3. (*Abstract Data Type Revolver*)

SOLUTION

```
/* Class IntRevolver is a cyclic data structure
 * where each cell holds an integer number.
 * The individual elements of this 'revolver'
 * contain the integer number and a pointer to
 * the next cell.
 *
 * For simplicity we access the cells of the
 * list elements directly (data being the integer
 * number, next being the pointer to the next
 * list element) instead of using methods.
 *
 * Carsten Butz, November 2002
 */
class IntRevolver{

    private IntListElement current = null;

    int getCurrent(){
        if(current == null)
            return 0;
        else
            return current.data;
    }

    void insert(int d){
        IntListElement nile = new IntListElement(d);

        if(current == null){
            /* revolver must be empty */
            current = nile;
            current.next = current;
        }
        else{
            /* revolver contains at least one element */
            nile.next = current.next;
            current.next = nile;
        }
        return;
    }

    IntListElement remove(){
        /* returnElement contains the pointer that will be returned */
        IntListElement returnElement = null;

        if(current != null){
            /* revolver contains at least one element */
            returnElement = current.next;

            if(current.next == current){
                /* revolver contains exactly one element */
                current = null;
            }
            else{
                /* revolver contains more than one element */

```

```

        current.next = returnElement.next;
    }
}
return returnElement;
}

void next(){
    current = current.next;
    return;
}

int count(){
    int howMany = 0;

    if(current != null){
        /* There is at least one element.
         * Use start as a marker, use current
         * to run through all cells until you
         * are back to start.
         */
        IntListElement start = current;
        do{
            current = current.next;
            howMany++;
        }while(start != current);

    }
    return howMany;
}

IntListElement find(int key){
    IntListElement r = null;
    boolean found = false;

    IntListElement start = current;
    do{
        if(key == current.data){
            r = current;
            found = true;
        }
        current = current.next;
    }while(start != current && !found);

    return r;
}

public String toString(){
    String s = "";

    if(current!=null){
        IntListElement start = current;

        do{
            s = s + current.data + "\n";
            /* Use the following line if you want to
             * see the contents of your revolver printed
             * on one line.
             */
            //s = s + current.data + "\t";
        }
    }
}

```

```
        current = current.next;
    }while(start != current);
}
return s;
}
}
```