

CSC 148H L0301 Midterm 2003
Duration — 50 minutes
Aids allowed: none

Student Number:

Lab day, time, room:

Last Name: First Name:

*Do **not** turn this page until you have received the signal to start.*
(Please fill out the identification section above,
and read the instructions below.) *Good Luck!*

This midterm consists of 3 questions on 4 pages (including this one). *When you receive the signal to start, please make sure that your copy is complete.* Comments are not required except where indicated, although they may help us mark your answers. They may also get you part marks if you can't figure out how to write the code.

Write your student number at the bottom of pages 2-4 of this test.

If you use any space for rough work, please indicate clearly what you want marked.

1: /11

2: /10

3: / 9

TOTAL: /30

Question 1. [11 MARKS]

Complete the body of the method `reversedList(Node)`, according to its external and internal comments.

```
public interface Stack {
    void push(Object o);
    Object pop();
    boolean isEmpty();
}
public class S implements Stack { /* body not shown */ }
public class Node {
    public Object value;
    public Node link;
    public Node(Object value) { this.value = value; }
}
public class Question1 {
    /** Return a copy of the linked list 'list' in reverse order.
     * @param list the first Node (null if empty) in a linked list.
     * @return     the first Node (null if empty) in the reversed copy of list. */
    public static Node reversedList(Node list) {
        if (list == null) { return null; }
        Stack s = new S(); // this is the only instance of S you may use.
        // Put the values from list into s.

        // Make reversed list from s. You must get values only from s now, not list.
        Node first = new Node(s.pop()); // the first Node in the reversed copy.
        Node last = first;              // the last Node in the reversed copy.

        return first;
    }
}
```

Question 2. [10 MARKS]

```
public class A {
    public static void f(A a) {
        System.out.println("A.f");
        a.g();
        a.h();
    }
    public void g() {
        System.out.println("A.g");
    }
    private void h() {
        System.out.println("A.h");
    }
}
```

```
public class B extends A {
    public static void f(B b) {
        System.out.println("B.f");
        b.g();
    }

    public void g() {
        System.out.println("B.g");
    }
    private void h() {
        System.out.println("B.h");
    }
}
```

```
public class M {
    public static void main(String[] args) {
        B b = new B();
        b.f(b);
        A.f(b);
    }
}
```

Part (a) [5 MARKS] Draw the memory model when line 1 of B's method `f` is first reached:

--	--

Part (b) [5 MARKS] Write the output from running the entire program `M`:

Question 3. [9 MARKS]**Part (a)** [2 MARKS]

Write your student number at the bottom of every page of the midterm (except the front page).

Part (b) [7 MARKS]

Write the body of `countInvalidIntegers(BufferedReader)` in the following class.

You are **not** required to throw an exception if the user of `countInvalidIntegers` violates the precondition.

Use the fact that `Integer.parseInt(String)` throws a `NumberFormatException` if its argument does not represent an integer.

```
import java.io.*;

public class Question3 {

    /**
     * Return the number of lines from 'br' that don't represent an integer.
     * Requires: br != null.
     */
    public static int countInvalidIntegers(BufferedReader br) throws IOException {

        int count = 0; // the number of lines not representing an integer

        return count;

    }
}
```

Total Marks = 30