

INTRODUCTION TO PROGRAMMING - CONCEPTS AND TOOLS  
IT UNIVERSITY OF COPENHAGEN

Examiner: Noah Torp-Smith & Jens Chr. Godskesen, IT University of Copenhagen  
Exam Form: Written Exam  
Date: June 18, 2004  
Time: 9:00 – 13:00

WRITTEN EXAM, JUNE 18, 2004

Please read the following instructions carefully before starting answering the questions.

- The exam is a written exam. You have four (4) hours to answer the questions.
- The exam is graded on the Danish 13 scale.
- The question set is available in English only. You are allowed to answer the questions in either *English* or *Danish*.
- The exam is an open book exam, which for this exam means that in particular the following aids may be used:
  - The text book *Java by Dissection*.
  - Additional course notes, like copies of slides, the notes by Peter Sestoft on searching and sorting, and on software testing.
  - Handwritten notes.
  - A language dictionary.
- No electronic devices are permitted. Electronic dictionaries are not allowed.
- The exam consists of three main questions. The weight of each question is given (in percentage points) next to the question. All three questions should be answered.
- This question set consists of eight (8) pages.
- When using classes or methods from the book or from the notes by Peter Sestoft it is not necessary to copy them. However, you have to give the precise location of those (for example, source, edition, section number, page number). It is your responsibility that the location can be identified.

(Weight 25%) QUESTION 1

We consider the following fragment of code

```
class S1 {

    String s;

    S1() {s = "Hi";}
    S1(String s) {this.s = s;}

    public String toString() {return "s = " + s;}
}

class S2 extends S1 {

    String st;

    S2() {st = s;}
    S2(String st) {s = "Davs"; this.st = st;}
    S2(String s, String st) {super(s); this.st = st;}

    public String toString() {return super.toString() + ", st = " + st;}
}

class S3 extends S2 {

    String st;
    String str;

    S3(String s, String st, String str) {
        super(s,st);
        this.str = str;
    }

    public String toString() {
        return("s = " + s + ", st = " + st + ", str = " + str);
    }
}
```

QUESTION 1.1

What does this test program print?

```
class Test {
    public static void main (String[] args) {

        S1 g1 = new S1("Aloha");
        S2 g2 = new S2();
        S2 g3 = new S2("Jalla");
        S2 g4 = new S2("Hej","Guten Tag");
        S1 g5 = new S2("Hello","Bon jour");

        System.out.println(g1.toString());
        System.out.println(g2.toString());
        System.out.println(g3.toString());
        System.out.println(g4.toString());
        System.out.println(g5.toString());
    }
}
```

### QUESTION 1.2

State the scope of the following identifiers:

1. The variable `s` declared in the first line of class `S1`;
2. The variable `st` declared in the first line of class `S2`;
3. The variable `st` declared in the first line of class `S3`;
4. The variable `st` declared as the second parameter of the constructor `S3(String, String, String)`.

### QUESTION 1.3

The following constructor for the class `S2` is suggested:

```
S2(S1 s, String st) {this = s; this.b = b;}
```

Argue whether this is correct Java code.

### QUESTION 1.4

In class `S3`, the method `toString` was declared `public`. Argue whether it could have been declared `private`.

### QUESTION 1.5

Here is another test program:

```
class Test {  
    public static void main (String[] args) {  
  
        S3 five = new S3("Goodbye","Cheers","Farewell");  
        System.out.println(five.toString());  
  
    }  
}
```

State the output of this program and give a reason for your answer.

(Weight 40%) **QUESTION 2**

This question is about complexity, searching and sorting. We look at some of the algorithms from Peter Sestoft's notes *Searching and Sorting in Java*. Specifically, consider the following methods for searching and sorting.

```
static void linsearch(int x, int[] arr, int n) {
    int i = 0; boolean found = false;
    while (!found && i < n) {
        if (arr[i] != x) i++;
        else found = true;
    }
}

static void binsearch(int x, int[] arr, int n) {
    int a = 0, b = n-1, i; boolean found = false;
    while (!found && a <= b) {
        i = (a+b) / 2;
        if (x < arr[i]) b = i-1;
        else if (arr[i] < x) a = i+1;
        else found = true;
    }
}

static void swap(int[] arr, int s, int t) {
    int tmp = arr[s]; arr[s] = arr[t]; arr[t] = tmp;
}

static void qsort(int[] arr, int a, int b) {
    if (a < b) {
        int i = a, j = b;
        int x = arr[(i+j) / 2];
        do {
            while (arr[i] < x) i++;
            while (arr[j] > x) j--;
            if (i <= j) {
                swap(arr, i, j);
                i++; j--;
            }
        } while (i <= j);
        qsort(arr, a, j);
        qsort(arr, i, b);
    }
}

static void quicksort(int[] arr) {
    qsort(arr, 0, arr.length - 1);
}
```

**QUESTION 2.1**

(This question is independent of the code fragments above). Assume you are given two programs that solves a given task. The first program has a running time of  $O(n^2)$  and the other has a running time of  $O(\lg n)$ . You have an input to the problem of size 100000. Can you say which of the programs will execute fastest on your input?

## QUESTION 2.2

When is `linsearch` preferable over `binsearch`?

Illustrate the execution of `binsearch(42, arr, length(arr)-1)` where `arr` is the following array.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
2	6	6	7	12	13	17	17	22	27	30	31	42	47	49	50	76	100	101	102

Show the values of `i`, `arr[i]`, and `found` after each iteration of the `while` loop.

## QUESTION 2.3

Consider the execution of `quicksort(arr)`, where `arr` is the following array.

0	1	2	3	4	5	6
137	157	131	163	151	127	149

In this execution, `qsort` will be called several times. For each of these, show the array and underline the value of `x` after completion of the `do-while` loop.

## QUESTION 2.4

Consider the statement

```
int x = arr[(i+j) / 2];
```

in `qsort`. Suppose we change that line to

```
int x = arr[a];
```

1. Will the algorithm still sort the array correctly?
2. What is the time complexity of the resulting algorithm?

Justify your answers.

## QUESTION 2.5

In this question, we consider the sorting routine *merge sort*. Just like `quicksort`, it works by *divide and conquer*, so it splits the input array in two, sorts the two sub-arrays recursively, and *merges* the two sorted arrays in one, sorted array. Here is the program, with `merge` undefined.

```
static int[] mergeSort(int[] arr) {
    if(arr.length < 2) return arr;

    int k = arr.length / 2;
    int[] A1 = new int[k];
    int[] B1 = new int[arr.length - k];

    for(int i = 0; i < k; i++)
        A1[i] = arr[i];
    for(int i = k; i < arr.length; i++)
        B1[i-k] = arr[i];

    int[] A = mergeSort(A1);
    int[] B = mergeSort(B1);

    return merge(A,B);
}
```

First, we copy the first and last half of the array into A1 and A2, respectively. Then we sort the two sub-arrays recursively, and finally, we combine the two sub-solutions using the method `merge`. `merge` has the signature

```
int[] merge(int[], int[]),
```

and it takes two sorted arrays as input and returns a sorted array which contains exactly the elements from the two input arrays. The `merge` algorithm is illustrated by an example. Here are two sorted arrays that are to be collapsed into one using `merge`.

0	1	2	3	4	5						
23	29	31	37	41	43						
<hr/>											
						0	1	2	3	4	
						20	30	40	50	60	

First the algorithm compares 23 and 20, and 20 is picked as the first element of the resulting array, since it is smaller. Then, 23 and 30 are compared and 23 is picked, and subsequently, 29 and 30 are compared and 29 is picked. The algorithm continues until all elements have been picked, and the resulting array is this:

0	1	2	3	4	5	6	7	8	9	10	
20	23	29	30	31	37	40	41	43	50	60	

Give an implementation of `merge`.

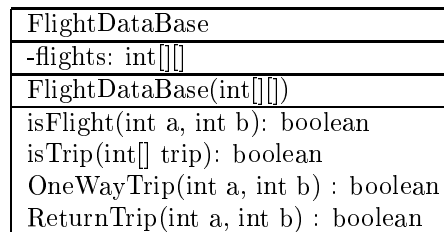
(Weight 35%) **QUESTION 3**

An airline company has flights between airports. To have an overview of the connections the company maintains a small database, i.e. a two dimensional array. For simplicity the destinations are identified by integers starting with zero (0). For example, a particular array with airports 0, . . . ,5 could look as follows:

	0	1	2	3	4	5
0	<i>false</i>	<i>true</i>	<i>true</i>	<i>true</i>	<i>false</i>	<i>true</i>
1	<i>false</i>	<i>false</i>	<i>true</i>	<i>true</i>	<i>false</i>	<i>true</i>
2	<i>true</i>	<i>true</i>	<i>false</i>	<i>false</i>	<i>false</i>	<i>true</i>
3	<i>true</i>	<i>false</i>	<i>false</i>	<i>false</i>	<i>true</i>	<i>false</i>
4	<i>false</i>	<i>false</i>	<i>true</i>	<i>true</i>	<i>false</i>	<i>false</i>
5	<i>false</i>	<i>true</i>	<i>false</i>	<i>false</i>	<i>true</i>	<i>false</i>

where a *true* at row *i* and column *j* denotes a flight from *i* to *j*, dually a *false* at row *i* and column *j* means that there is no flight from *i* to *j*. Clearly, there are no flights from an airport to itself (which explains the values on the main diagonal). It may be that there are only flights in one direction between two destinations.

In this question we will implement a class `FlightDataBase` representing a route database with the following UML class diagram:



The instance variable `flights` contains the two-dimensional array of flights described above.

**QUESTION 3.1**

Show all the code needed to create and initialize an example flight database object with flights of your own choice.

**QUESTION 3.2**

Show the code of the method `boolean isFlight(int a, int b)` which returns *true* if there is a flight from *a* to *b* and *false* otherwise.

**QUESTION 3.3**

Declare a new method (not present in the UML diagram above) that may be used for updating the individual flights in the database `flights`.

**QUESTION 3.4**

Show the code of the method `boolean isTrip(int[] t)` which returns *true* if it is possible to make the trip *t* and *false* otherwise. The trip *t* is a sequence of destinations, for example, 1, 2, 5, 1, showing a trip traveling from 1 to 2 to 5 and back to 1. The trip is represented as a 1-dimensional array of integers.

**QUESTION 3.5**

Show the code of the method `boolean returnTrip(int a, int b)` which returns *true* if it is possible to make a trip from *a* to *b* and a trip returning from *b* to *a*, and *false* otherwise. To ease your solution you should make use of the method `OneWayTrip` (see Question 3.6).

### QUESTION 3.6

Implement the method `boolean oneWayTrip(int a, int b)` which returns *true* if it is possible to make a trip from `a` to `b` and *false* otherwise. You may assume that by default there is always a one way trip from a destination to itself.