Algorithms and Data Structures (BADS/SGDS)

Exam 29 May 2017

Thore Husfeldt and Riko Jacob, ITU

Instructions

What to bring. You can bring any written aid you want. This includes the course book and a dictionary. In fact, these two things are the only aids that make sense, so I recommend you bring them and only them. But if you want to bring other books, notes, print-out of code, old exams, or today's newspaper you can do so. (It won't help.)

Answering multiple-choice questions. In the multiple-choice questions, there is one and only one correct answer. However, to demonstrate partial knowledge, you are allowed to check 2 or more boxes, but this earns you less than full points for that question.

number of checked boxes	0	1	2	3	4
points if correct answer checked		1	0.5	0.21	0
points if correct answer not checked	0	−0.33	−0.5	−0.62	

In particular, the best thing is to only check the correct answer, and the worst thing is to check all answers but the correct one. If you don't check anything (or check *all* boxes) your score is 0. Also, if you check boxes at random, your expected score is 0. (Just to make sure: a question that is not multiple-choice cannot give you negative points.)

Where to write. Mark you answers to questions 1–3 on pages 8 and 9. If you really have to, you may use separate sheets of paper for these questions instead, but please be clear about it (cross out everything and write "see separate paper, page 1" or something like that.) Question 4 is answered on separate sheet(s) of paper anyway. For the love of all that is Good and Holy, write legibly. Hand in pages 8 and 9, and any separate sheet(s) of paper. Do not hand in pages 1–7, they will not be read.

Exam questions

1. Analysis of algorithms

(a) (*1 pt.*) Which pair of functions satisfy $f(N) \sim g(N)$?

$\boxed{\text{A}}$ $f(N) = 2N$ and $g(N) = N + N + 2$ $\boxed{\text{B}}$ $f(N) = 2N + 10$ and $g(N) = N^2 + 10$

$\boxed{\text{C}}$ $f(N) = 2N^2 + N$ and $g(N) = 2N^2 + N^2$ $\boxed{\text{D}}$ $f(N) = 2N^2 + N$ and $g(N) = N^2 + N^3$

(b) (*1 pt.*) Which pair of functions satisfy $f(N) = O(g(N))$?

$\boxed{\text{A}}$ $f(N) = N + N + N$ and $g(N) = N/\log N$ $\boxed{\text{B}}$ $f(N) = (N+1)\cdot(N+1)$ and $g(N) = N + 1$

$\boxed{\text{C}}$ $f(N) = (\log N)\cdot(\log N)$ and $g(N) = N \log N$ $\boxed{\text{D}}$ $f(N) = N^5 + 5N$ and $g(N) = 4N^4$

(c) (*1 pt.*) How many stars are printed?

```
for (int i = 1 ; i < N; i = i*2) StdOut.print("**");
```

Base revision 3eec736. . ., Sun Aug 9 23:15:45 2020 +0200, Riko Jacob @X260.

$\boxed{A}$ $\sim 2\log_2 N$ $\boxed{B}$ $\sim N$ $\boxed{C}$ $\sim N\log N$ $\boxed{D}$ $\sim \frac{1}{2}N^2$

(d) (*1 pt.*) How many stars are printed? (Choose the smallest correct estimate.)

```
for (int i = 1; i < N/2; i = i+1)
     for (int j = 0; j < i; j = j+1)
          StdOut.print("**");
```

$\boxed{A}$ $O(\log N)$ $\boxed{B}$ $O(N)$ $\boxed{C}$ $O(N\log N)$ $\boxed{D}$ $O(N^2)$

(e) (*1 pt.*) What is the asymptotic running time of the following piece of code? (Choose the smallest correct estimate.)

```
if (N < 1000) for (int i = 0; i <   N; i = i+1) A[i] = i;
else          for (int i = 0; i < N*N; i = i+1) A[i] = 2*i*i;
```

$\boxed{A}$ linear in N $\boxed{B}$ linearithmic in N $\boxed{C}$ quadratic in N $\boxed{D}$ cubic N

(f) (*1 pt.*) Find a recurrence relation for the number of arithmitic operations (additions and subtractions) performed by the following recursive method. (Choose the smallest correct estimate. The base case is $T(0) = 0$ in all cases.)

```
static int r(int N)
{
    if (N > 0) return r(N-1) + N + 1;
    else       return 1;
}
```

$\boxed{A}$ $T(N) = T(N-1) + N + 1$ $\boxed{B}$ $T(N) = T(N) + T(N-1)$
$\boxed{C}$ $T(N) = T(N-1) + 3$ $\boxed{D}$ $T(N) = T(N-1) - N - 1$

(g) (*1 pt.*) Assume I have a function `f(int K)` that runs in amortised constant time, but linear worst case time (in K). What is the running time of

```
for (int i = 0; i<N; i=i+1) f(N);
```

(Choose the smallest correct estimate.)

$\boxed{A}$ Linear in N.

$\boxed{B}$ Linearithmic in N.

$\boxed{C}$ Quadratic in N.

$\boxed{D}$ Impossible to say from the information given.

2. Class M. The next few questions all concern the class defined in fig. 1. We let N denote the number of elements in data structure.

(a) (*1 pt.*) What is the output of the following code:

```
M m = new M();

m.insert(true, 5);
m.insert(false, 3);
m.insert(true, 7);
m.insert(false, 5);
StdOut.print(m.getNextMarked());
StdOut.print(m.getNextMarked());
```

```java
import edu.princeton.cs.algs4.*;

public class M
{
    class Node
    {
        boolean marked;
        int val;
        Node next;
    }

    Node first;
    Node cur; // if non-null, points to a marked node

    public void insert(boolean marked, int val)
    {
        Node N = new Node();
        N.marked = marked;
        N.val = val;
        N.next = first;
        first = N;
        cur = N;
        advance();
    }

    private void advance()
    {
        while (cur != null && !cur.marked) cur = cur.next;
    }

    public int getNextMarked()
    {
        if (cur == null) throw new RuntimeException("No more marked values");
        int val = cur.val;
        cur = cur.next;
        advance();
        return val;
    }
}
```

Figure 1: Class `M` (for Mystery).

(b) (*1 pt.*) Draw the data structure after the following operations: (This means "*at the end* of the operations," not "*after each* operation," so you need to draw only a single picture. Make sure to include all instance variables.)

```
M m = new M();

m.insert(true, 5);
m.insert(false, 3);
m.insert(true, 7);
m.insert(false, 5);
StdOut.print(m.getNextMarked());
```

(c) (*1 pt.*) What is the worst-case running time of `insert`? (Choose the smallest correct estimate.)

$\boxed{\text{A}}$ $O(1)$. $\boxed{\text{B}}$ $O(\log N)$. $\boxed{\text{C}}$ $O(N)$. $\boxed{\text{D}}$ $O(N^2)$.

(d) (*1C pt.*) What is the worst-case running time of `getNextMarked()`? (Choose the smallest correct estimate.)

$\boxed{\text{A}}$ $O(1)$. $\boxed{\text{B}}$ $O(\log N)$. $\boxed{\text{C}}$ $O(N)$. $\boxed{\text{D}}$ $O(N^2)$.

(e) (*1 pt.*) What is the total worst-case running time of N consecutive calls to `getNextMarked()`? (Choose the smallest correct estimate.)

$\boxed{\text{A}}$ $O(1)$. $\boxed{\text{B}}$ $O(\log N)$. $\boxed{\text{C}}$ $O(N)$. $\boxed{\text{D}}$ $O(N^2)$.

(f) (*1 pt.*) What is the total worst-case running time of N calls each to `insert()` and `getNextMarked()`, in any order and with any arguments? (Choose the smallest correct estimate.)

$\boxed{\text{A}}$ $O(1)$. $\boxed{\text{B}}$ $O(\log N)$. $\boxed{\text{C}}$ $O(N)$. $\boxed{\text{D}}$ $O(N^2)$.

(g) (*1 pt.*) Write a method `int pop()` that removes and returns the most recently inserted element (or raises a sensible exception), just like in a Stack data structure. In particular, the intended functionality is that the following operations

```
M m = new M();
m.insert(true, 5);
m.insert(true, 3);
StdOut.println(m.pop());
StdOut.println(m.getNextMarked());
```

print `3` and then `5`. *Don't change any other methods in* `M` *and don't introduce new instance variables to* `M`.

(h) (*1 pt.*) Write a method `hasTwoNextMarked()` that returns `true` if there "are more than one marked nodes left," i.e., if the next two calls to `getNextMarked()` would not raise an exception. *Don't change any other methods in* `M` *and don't introduce new instance variables to* `M`.

(i) (*1 pt.*) Can `hasTwoNextMarked()` (defined in the previous question) be implemented so as to run in constant time, possibly by modifying other parts of `M`? If yes, explain how. If no, give an argument.

3. Operation of common algorithms and data structures.

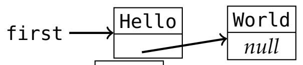
(a) (*1 pt.*) Consider the following sequence of operations on a data structure, where a number i means `insert(`i`)` and "$*$" means `remove()`. The data structure is initially empty.

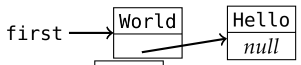
$$5 \quad 3 \quad 8 \quad * \quad *$$

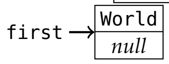
What is the data structure if the removed elements are $5, 8$ in that order?

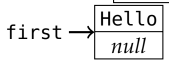
$\boxed{\text{A}}$ Stack $\boxed{\text{B}}$ Queue $\boxed{\text{C}}$ Heap $\boxed{\text{D}}$ Neither of those

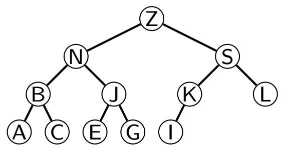
(b) (*1 pt.*) Consider a Stack implementation based on a linked list. Beginning with an empty data structure, assume that I pushed the strings `Hello` and then `World` in that order, and then popped once. How does the data structure look after these operations?

A 

B 

C 

D 

(c) (*1 pt.*) Consider the following tree:  Which statement is true?

A The tree is in heap order, but not in search tree order.

B The tree is not in heap order, but in search tree order.

C The tree is both heap order and search tree order.

D The tree is in neither heap order nor search tree order.

(d) (*1 pt.*) Insert the keys `1 3 2 5 2 4` in that order into a binary search tree (without any rebalancing, using algorithm 3.3 in [SW]). Draw the result.

(e) (*1 pt.*) Assume I sort the letters `S E Q U E N C E` using merge sort. Which one of the following situations *can* arise at some time during the algorithm?

A `C E Q U E N S E` B `C S E Q U E N E` C `S E Q U E C N E` D `E S Q U E N C E`

(f) (*1 pt.*) Consider the key–value pairs

key	V	O	V	S	E
value	0	1	2	3	4

We use the hash function `(key.hashCode() & 0x7fffffff) % 7`. To spare you the calculations, the hash values are here:

key	F, M, T	G, N, U	A, H, O, V	B, I, P, W	C, J, Q, X	D, K, R, Y	E, L, S, Z
hash	0	1	2	3	4	5	6

The keys are inserted from left to right into an initally empty hash table of size 7 using separate chaining. Draw the result.

(g) (*1 pt.*) In which sense is a balanced binary search tree better than an unbalanced binary search tree?

A It has better asympotic worst-case performance.

B It is easier to implement deletion.

C It uses less code.

D It uses less space.

(h) (*1 pt.*) Insert the letters `T R U M P` in that order into a red–black BST as defined in the textbook (algorithm 3.4). Draw the resulting structure in the style of the book, use a fat edge to represent red links. (Your answer will be photocopied in black and white, so don't use fancy colours.)

(i) (*1 pt.*) Sort 4 strings using LSD string sort (algorithm 5.1 in [SW], with $N = 4$, $W = 4$). Give a trace in the style of the book by completing this table:

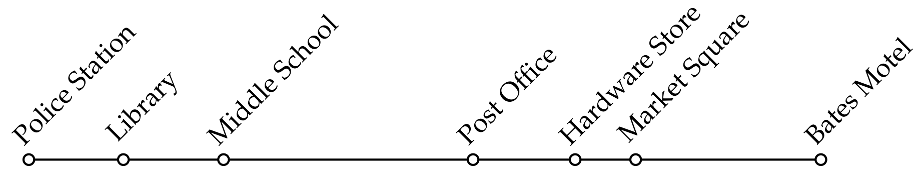
input	$d = 3$	$d = 2$	$\cdots$
BOBA	BOBA	...	
SURE	...		
BIKE			
IDAX			

4. Design of algorithms

Every episode of the tv show *Stranger Things* follows the same storyline: Sheriff Grimes learns about a monster in his home town of Castle Rock, then runs to that monster at full speed and without detours, fights it, and kills it. There are a bunch of monsters in Castle Rock, so we need to help Sheriff Grimes picking one. The constraints are: (1) The monster has to die before the episode is over. (2) The monster has to die as late as possible, to make the episode as exciting as possible. (The remaining time in the episode is filled with a boring side-plot involving awkward teenagers.)

Your task is to write an algorithm that returns the best monster for Sheriff Grimes to kill. He starts the episode at the Police Station.

(a) (*3 pt.*) We assume that Castle Rock just consists of one long street, Main Street, with various *sites* along it.



The input has three parts. The first part gives the number n of sites, the number k of monsters, the length l of the episode (in seconds), as space-separated integers on a single line. Then follow n lines, giving the sites as they appear along Main Street, with the distance to the preceding site. The distance is given in seconds, assuming a Sheriff running at full speed. Then follow k lines in arbitrary order, each giving the name, location, and *killtime* for a monster. The killtime is the time (in second) that it takes for Sheriff Grimes to kill that monster.

```
7 5 3020
Police Station, 0
Library, 500
Middle School, 531
Post Office, 1324
Hardware Store, 542
Market Square, 329
Bates Motel, 981
centuries old vampire, Post Office, 1243
Cthulhu, Library, 873
a bunch of zombies, Market Square, 326
demon from another dimension, Middle School, 1643
Cronenberg, Market Square, 4817
```

In our example, Sheriff Grimes could run to the Middle School (1031 seconds) and kill the demon there, for a total time of 2674 seconds.

(b) (*2 pt.*) We look at the same problem but with a more exciting city map. Now the sites are joined by *streets*.

The input has three parts. The first part gives the number n of sites, the number s of streets, the number k of monsters, and the length l of the episode (in seconds), as space-separated integers on a single line. Then follow s lines, one for each street. Such a line gives the sites at both ends of the street, separated by `--`, followed by a single integer, the length of the street in seconds, assuming a Sheriff running at full speed. Then follow k lines as in the first part of the exercise.

```
6 5 3 20
Police Station -- Jail 30
Police Station -- Hardware Store 10
Police Station -- Bates Motel 10
Police Station -- Market Square 5
Market Square -- Library 5
Cthulhu, Jail, 5
Cronenberg, Market Square, 17
E.T.,  Library, 9
```

For both questions, explain your algorithm briefly and clearly, and state the running time of your algorithm in terms of the given parameters. You are strongly encouraged to make use of existing algorithms, models, or data structures from the book, but please be precise in your references (for example, use page numbers or full class names of constructions in the book). Each question can be perfectly answered on half a page of text. (Even less, in fact.) If you find yourself writing much more than one page, you're using the wrong level of detail. However, it is a very good idea to include a drawing of a concrete (small) example. You don't need to write code. (However, some people have an easier time expressing themselves clearly by writing code. In that case, go ahead.) You are evaluated on correctness and efficiency of your solutions, and clarity of explanation.

This is a toy exercise, the "story" is purely window dressing for entertainment. You cannot (nor do you need to) use any *domain knowledge*, like "no TV episode is longer than 5 hours" or "fighting zombies typically lasts half an hour" or assumptions about the street map of Castle Rock.

Answers

This is where you mark your answers for questions 1–3. It is strongly preferred that you fit your answers on these sheets, except for question 4. If you really must, you can use a separate sheet of paper instead. Please indicate that clearly.

Your name:

	1a	1b	1c	1d	1e	1f	1g	2c	2d	2e	2f	3a	3b	3c	3e	3g
A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
B	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B	B
C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C	C
D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D

2a

2b

2g

```
public int pop() {

}
```

2h

```
public boolean hasTwoNextMarked() {

}
```

2i

3d

3f

3h

3i

input	$d = 3$	$d = 2$
BOBA	BOBA	
SURE		
BIKE		
IDAX		

4a,4b *on a separate piece of paper.*