Algorithms and Data Structures (BADS/SGDS)

Exam 31 August 2016

Thore Husfeldt, 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 $f(N) \sim g(N)$?

A $f(N) = 3N + 3$ and $g(N) = N + 3$
B $f(N) = 3N$ and $g(N) = N^3 + 3$
C $f(N) = 2N^2 + N$ and $g(N) = 2N^2 + 2N \log N$
D $f(N) = 2N^2 + N$ and $g(N) = N^2 + 2N$

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

A $f(N) = N^3$ and $g(N) = N \log N$
B $f(N) = (N + 1) \cdot (N + 1)$ and $g(N) = N^2 + 1$
C $f(N) = (\log N) \cdot (\log N)$ and $g(N) = 1699 + \log N$
D $f(N) = N^5$ and $g(N) = 5N^4 + N$

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

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

Base revision c5a2536. . ., Mon Jan 30 13:44:50 2017 +0100, Riko Jacob (@ITU Air 13).

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

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

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

A $O(\log N)$ B $O(N)$ C $O(N \log N)$ 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*N; i = i+1) A[i] = 0;
else          for (int i = 0; i <   N; i = i+1) A[i] = i*i*i;
```

A linear in N B linearithmic in N C quadratic in N 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 $T(0) = T(1) = 0$ in all cases.)

```
static int r(int N)
{
    if (N > 1) return r(N-1) + r(N-2);
    else       return 1;
}
```

A $T(N) = T(N-1) + N + 1$
B $T(N) = T(N) + T(N-1) + 3$
C $T(N) = T(N) + N + 2$
D $T(N) = T(N-1) + T(N-2) + 3$

(g) (*1 pt.*) Assume I have a function `f(int K)` that runs in amortised constant time, and logarithmic 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.)

A Linearithmic in N.
B Linear in N.
C Quadratic in N.
D Impossible to say from the information given.

2. Class M. The next few questions all concern the class defined in fig. 1.

(a) (*1 pt.*) What is the result of executing the `main` method in `M`?

A Depends on the implementation of `MinPQ`.
B `Alice`
C `Bob`
D Nothing.

(b) (*1 pt.*) Class *M* behaves like which well-known data structure?

A (LIFO) Stack. B (FIFO) Queue. C Priority queue. D Union–Find.

```
1  import edu.princeton.cs.algs4.*;
2
3  public class M
4  {
5          public class PQObject implements Comparable<PQObject>
6          {
7                  String item;
8                  Integer priority;
9                  PQObject(String x, int priority)
10                 {
11                         this.item= x;
12                         this.priority= priority;
13                 }
14                 public int compareTo(PQObject that) {
15                         return this.priority.compareTo(that.priority);
16                         }
17         }
18
19         MinPQ<PQObject> pq = new MinPQ<PQObject>();
20         int counter;
21
22         public void insert(String x)
23         {
24                 pq.insert(new PQObject(x,counter));
25                 counter+= 1;
26         }
27
28         public String remove()
29         {
30                 return pq.delMin().item;
31         }
32
33         public static void main(String[] args)
34         {
35                 M m = new M();
36                 m.insert("Alice");
37                 m.insert("Bob");
38                 StdOut.println(m.remove());
39         }
40 }
```

Figure 1: Class `M` (for Mystery). The method names `insert` and `remove` are purposefully vague.

(c) (*1 pt.*) In the remaining questions, we assume that `MinPQ` is implemented as an array-based binary heap, with dynamic resizing of the underlying array. What is the worst-case running time of `insert` after N operations? (Choose the smallest correct estimate.)

A $O(\log N)$. B $O(N)$. C $O(N \log N)$. D $O(1)$.

(d) (*1 pt.*) What is the worst-case running time of `remove` after N operations? (Choose the smallest correct estimate.)

A $O(\log N)$. B $O(N)$. C $O(N \log N)$. D $O(1)$.

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

```
M m = new M();
for (int i = 0; i < N; i++) m.insert("Bob");
for (int i = 0; i < N; i++) m.remove();
```

A $O(\log N)$. B $O(N)$. C $O(N \log N)$. D $O(N^2)$.

(f) (*1 pt.*) Assume I changed the first line of the body of the `insert` method to

```
pq.insert(new PQObject(x,-counter));
```

(So the second argument to the constructor now uses the *negation* of the original value.) What happens?

A Now all operations take constant worst-case time.

B Now `insert` takes linear worst-case time.

C Now `remove` takes constant worst-case time.

D The data structure is now a stack.

(g) (*1 pt.*) Add a method

```
public int size()
```

that returns the total number of elements in the data structure. *Don't change any other methods in class* `M` *and don't introduce new instance variables to* `M`.

(h) (*2 pt.*) Add a method

```
public int numempty()
```

that returns the total number of *empty strings* in the data structure. You may change other methods and add new instance variables. Faster is better.

3. Operation of common algorithms and data structures.

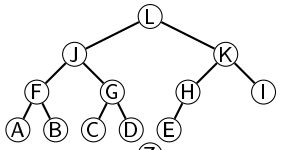
(a) (*1 pt.*) Consider the following sequence of operations on a data structure, where a number i means `insert(`i`)` and "$*$" means `remove()`. The data structure is initially empty.

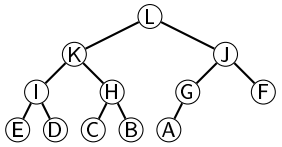
5 $*$ 3 9 7 $*$

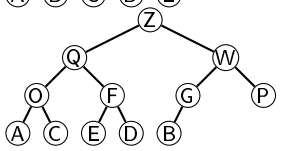
What is the data structure if the removed elements are $5, 7$ in that order?

A Symbol table B Stack C Queue D Heap

(b) (*1 pt.*) Which of the following trees is in search tree order?

A 

B 

C 

D None of them.

(c) (*1 pt.*) Draw two different search trees for the keys `4 6 8 1 2`.

(d) (*1 pt.*) Assume I sort the letters `P I K A C H U` using selection sort. Which one of the following situations *does not* arise at any time during the algorithm?

A `P I K A C H U` B `A C H I K P U` C `A I K P C H U` D `I P K A C H U`

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

key	P	A	N	A	M	A	P	A	P	E	R	S
value	0	1	2	3	4	5	6	7	8	9	10	11

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.

(f) (*1 pt.*) In which sense is insertion sort better than selection sort?

A It has better asympotic worst-case performance

B It is faster on sorted inputs.

C It is in-place while selection sort is not.

D It uses less space.

(g) (*1 pt.*) Insert the letters `P I K A C H U` in that order into a 2–3 tree. Draw the resulting structure in the style of the book.

(h) (*1 pt.*) Sort 4 strings using LSD string sort (algorithm 5.1 in [SW], with $N = 4$, $W = 3$). Give a trace in the style of the book by completing this table:

input	$d = 2$	$d = 1$	$d = 0$
OLE	OLA	...	
OLA	...		
NIS			
PUK			

4. Design of algorithms

(a) (*2 pt.*) You have been collecting Pokemon, writing down each collected individual in a text file. Your list contains many duplicates. (This is because you just mindlessly collected every Pokemon you encountered, not trusting your memory of whether you had seen that particular species already.) The input is a list of N lines, each line containing the name of a Pokemon species.

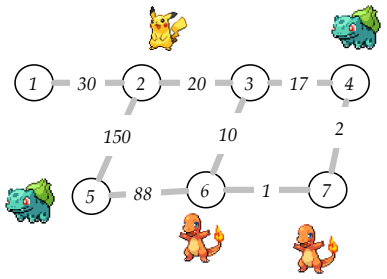
```
Charmander
Pikachu
```

Pokemon are fictional creatures from Japenese popular culture. Unless you're twelve years old, it's not important, just think of them as text strings.

```
Charmander
Bulbasaur
Charmander
Pikachu
...
```

There exist 151 different species of Pokemon in total, and the goal of the game is collecting them all. Design a method that determines if you're finally done with this stupid game, i.e., if your list contains every single of the 151 species of Pokemon at least once. Fun fact: the longest name of a Pokemon is 11 letters ("Fletchinder").

(b) (*3 pt.*) You want to catch a specimen of the species *Charmander*. Luckily, several of them are in the neighbourhood, but you only have 1 hour to catch one.
To fix notation: We model your local neighbourhood as an undirected, connected graph G with vertex set $V = \{1, \ldots, n\}$ and edge set E, we have $m = |E|$. The Pokémon appear only in the graph's vertices, at most one Pokémon per vertex. You live in vertex 1, which may also contain a Pokémon. With each edge $e \in E$ we associate an integer weight $w(e) \geq 0$ that models the amount of time (in minutes) you need to traverse the edge. (Let's make life easy and assume that each edge can be traversed in each direction in the same amount of time—this isn't really true for staircases, but it doesn't matter for the exercise.)



The graph in the example is very simple (it is planar, for example), but in general you can assume nothing about the graph other than being connected.
Input: First, the edges of an undirected graph (endpoints separated by `-`, followed by weight). Then, the Pokémon, one species per line, consisting of species name followed by the vertices where specimen appear.
Output: "True" if it is possible to reach a Charmander with a path of length at most 60 from vertex 1. "False" otherwise.
Here is an example:

Say, because your phone's batteries run out.

Input:

```
1-2 30
2-3 20
2-5 150
3-6 10
3-4 17
4-7 2
5-6 88
6-7 1
Bulbasaur 4 5
Charmander 6 7
Pikachu 2
```

Output:

```
True
```

(The correct answer is True because you can reach the Charmander at node 6 in 60 minutes.)

For both questions (4a and 4b), state the running time of your resulting algorithms in terms of the given parameters; introduce new parameters if you need to. 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). Be short and precise. 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.

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

2g

```
public int size() {




}
```

2h

3c

3e

3g

3h

input	$d = 2$	$d = 1$	$d = 0$
OLE	OLA		
OLA			
NIS			
PUK			

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