Algorithms and Data Structures (BADS/SGDS)

Exam 29 May 2015

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. Do not hand in pages 1–6. (You are welcome to mark pages 1–6 any way you want, including checking boxes. But you can not hand these pages in. They won't count.) Instead, mark you answers to questions 1a to 3h on pages 7 and 8. 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.) Four questions need to be answered on separate sheet(s) of paper anyway. For the love of all that is Good and Holy, write legibly.

Exam questions

1. Analysis of algorithms

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

A $f(N) = N + 2N + 3N$ and $g(N) = 6N$

B $f(N) = (N+1) + (N+2) + (N+3)$ and $g(N) = N + 6$

C $f(N) = N + N^2 + N^3$ and $g(N) = N^6$

D $f(N) = \log N + \log 2N + \log 3N$ and $g(N) = \log 6N$

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

A $f(N) = N + N + N$ and $g(N) = N$

B $f(N) = (N+1) \cdot (N+1) \cdot (N+1)$ and $g(N) = N + 1$

C $f(N) = (\log N) \cdot (\log N) \cdot (\log N)$ and $g(N) = \log N$

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

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

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

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

(d) (*1 pt.*) How many stars are printed when I call $f(N)$? (Choose the smallest correct estimate.)

```
static void f(int K)
{  for (int i = 0; i < K; i = i+1) g(K); }

static void g(int R)
{  for (int i = 0; i < R; i = 2*i) StdOut.print("*"); }
```

Choose the smallest correct estimate.

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

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

```
if (N < 1000)
  for (int i = 0; i < N*N*N; i = i+1) A[i] = j;
else if (N < 10000)
  for (int i = 0; i < N; i = i+1) A[i] = j;
else
  for (int i = 0; i < N*N; i = i+1) A[i] = i;
```

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

(f) (*1 pt.*) Find a recurrence relation for the number of arithmitic operations (additions and subtractions) performed by the following recursive method:

```
static int r(int N)
{
    if (N > 2) return r(N-2) + N;
    if (N == 0) return 1;
}
```

(Choose the smallest correct estimate.)

$\boxed{\text{A}}$ $T(N) = T(N-2) + N$ $\boxed{\text{B}}$ $T(N) = T(N-1) + T(N-2)$

$\boxed{\text{C}}$ $T(N) = T(N-1) + 1$ $\boxed{\text{D}}$ $T(N) = T(N-2) + 2$

(g) (*1 pt.*) Consider the following piece of code:

```
class A
{
  int k = 0;

  void f(int N)
  {
    if (k < 0)
    {
      k = N;
      for (int i = 0; i<N; ++i) StdOut.print("*");
    }
```

```
      k = k-1;
      StdOut.print("*");
    }
  }
```

How many stars are printed by the operations `A a = new A(); a.f(N);`?

$\boxed{\text{A}}$ ~ 1. $\boxed{\text{B}}$ $\sim N$. $\boxed{\text{C}}$ $\sim N/2$. $\boxed{\text{D}}$ $\sim \log N$.

(h) (*1 pt.*) Consider class *A* from the previous question. How many stars are printed *in total* by the sequence of operations `A a = new A(); for (int i=0; i<N; ++i) a.f(N);` ? Choose the smallest correct estimate.

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

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

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

$\boxed{\text{A}}$ (LIFO) Stack. $\boxed{\text{B}}$ (FIFO) Queue. $\boxed{\text{C}}$ Priority queue. $\boxed{\text{D}}$ Union–Find.

(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 you draw the entire data structure. Make sure to include all instance variables.)

```
Z z = new Z(4);
z.update(0,1);
z.update(0,3);
```

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

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

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

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

(e) (*1 pt.*) What is the *total* running time of the following code:

```
Z z = new Z(N);
for (int i = 0; i < N; i++) z.update(0,i);
```

(Choose the smallest correct estimate.)

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

(f) (*1 pt.*) What is the *total* running time of the following code:

```
Z z = new Z(N);
for (int i = 0; i < N-1; i = i+2) z.update(i,i+1);
```

(Choose the smallest correct estimate.)

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

(g) (*2 pt.*) An element *i* is *isolated* if we have `query(i,j)==false` for all $j \neq i$. Extend class Z with a method

```
        void makeIsolated(i)
```

with the effect of isolating *i*. Write complete and correct Java. *Don't change any other methods in class Z.* State the running time; faster is better.

```
public class Z
{
  int[] next, prev;

  Z(int N)
  {
    prev = new int[N];
    next = new int[N];
    for (int i = 0; i<N; ++i)
    // put element i in a list of its own
      {
        next[i] = i;
        prev[i] = i;
      }
  }

  int first(int i)
  // return first element of list containing i
  {
    while (i != prev[i]) i = prev[i];
    return i;
  }

  int last(int i)
  // return last element of list containing i
  {
    while (i != next[i]) i = next[i];
    return i;
  }


  void update(int i, int j)
  {
    int f = first(j);
    int l = last(i);
    next[l] = f;
    prev[f] = l;
  }

  boolean query(int i, int j)
  {
    return last(i) == last(j);
  }
}
```

Figure 1: Class Z. The method names `update` and `query` are purposefully vague. Access modifiers (`public`, `private`, etc.) are omitted for readability.

(h) (*2 pt.*) Extend class Z with a method `boolean existsIsolated()` that checks if there is an isolated element in the data structure. Explain your solution in prose, pseudocode, drawing, or code, as you see fit. Be brief. You may change other parts of class Z. State the running time; faster is better.

(i) (*2 pt.*) Suggest how to modify class Z so as to support `update` in constant worst-case time. Explain your solution in prose, pseudocode, drawing, or code, as you see fit. Be brief.

3. Operation of common algorithms and data structures.

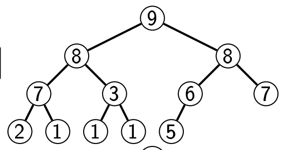
(a) (*1 pt.*) Consider the following sequence of operations on a data structure, where a number i means `insert(`i`)` and "$*$" means `remove()`. The data structure is initially empty.

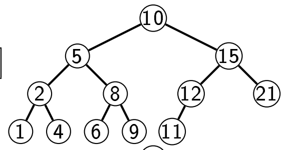
$$1 \quad 12 \quad 5 \quad * \quad 3 \quad 7 \quad * \quad * \quad * \quad 2 \quad 4 \quad 13 \quad * \quad 14 \quad 15 \quad * \quad * \quad *$$

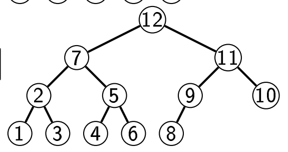
What is the data structure if the removed elements are: $1, 12, 5, 3, 7, 2, 4, 13$, in that order?

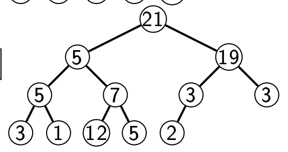
A Symbol table B Stack C Queue D Heap

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

A


B


C


D


(c) (*1 pt.*) Insert the keys `2 4 5 1 3 6` in that order into a heap. Draw the result.

(d) (*1 pt.*) Assume I sort the letters `S E Q U E N C E` using merge sort. Which of the following situations *cannot* arise at any time during the algorithm?

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

(e) (1 *pt.*) Run (i) insertion sort and (ii) selection sort on the 8-letter input `A L G O R I S T`. Stop each algorithm after exactly 3 calls to `exch`. Write the resulting sequences.

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

key	E	X	A	M	K	E	Y	S
value	0	1	2	3	4	5	6	7
hash	4	1	0	1	10	4	2	7

We use the hash function `(key.hashCode() & 0x7fffffff) % 11`. To spare you the calculations, the hash values are given in the table above as well. The elements are inserted from left to right into an initally empty hash table using linear probing. Draw the result.

(g) (*1 pt.*) In which sense is insertion sort better than quick sort?

A It has better asympotic worst-case performance

B It is faster on random input.

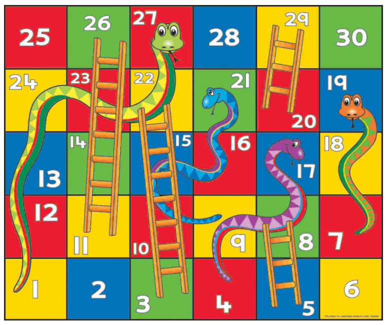
C It allows for a natural recursive implementation

D It uses less space.

(h) (*1 pt.*) Insert the letters `E X A M` 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.)

4. Design of algorithms

We consider the popular children's game *Snakes and Ladders*. It is played with a 6-sided die and a game board like this one:



Squares a numbered $\{1, \ldots, K\}$. In the example above, $K = 30$. You start in square 1. Roll the die and move forward that many steps. If your move ends on the head of a snake, you are moved back to the snake's tail. If your move ends on the bottom of a ladder, you are moved to the top of the ladder. In the example, assume you stand on square 18. If your die came up ⚂, your move would end in square 9. If your die came up ⚁, your move would end in square 29. The goal is to end in square 30.

The game board is described by a text file containing three lines. The first line contains K. The second line contains the position of the S snakes, as a list of pairs (h, t) where h is the snake's head and t is the snake's tail. The third line contains the position of the L ladders, as a list of pairs (b, t) where b is the ladder's bottom and t is the ladder's top.

The example game board would be represented like this, with $K = 30$, $L = S = 4$:

```
30
(17,4), (21,9), (19,7), (27,1)
(11,26), (3,22), (5,8), (20,29)
```

(a) (*3 pt.*) The input is a board description as above, and a sequence D of r integers $D = (d_1, \ldots, d_r)$ representing die rolls ($1 \leq d_i \leq 6$). Design an algorithm that decides if the sequence gets you from square 1 to square K. For instance, in the example board and die roll sequence $D = (6, 6, 5, 2, 2)$ the answer would be "yes." For the die roll sequence $D = (6, 6, 6, 2, 2)$ the answer would be "no." (As far as I can tell, you'd end up in square 4.)

(b) (*3 pt.*) The input is a board description as above. Design an algorithm that returns the shortest sequence of die rolls that gets you to square K. (If there are several shortest sequences, any of them is fine.) For the example board, a correct answer is $D = (2, 6, 2)$.

For both questions, state the running time of your resulting algorithms in terms of the given parameters (K, L, S, r). You are 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.

For simplicity, let's agree that you are allowed to "overshoot" the goal, so from square 26 you win with either of ⚃, ⚄ or ⚅. While we're listing simplifying assumptions: no square contains both a ladder top/bottom and a snake head/tail.

Answers

This is where you mark your answers for 1a to 3h.

Your name:

	1a	1b	1c	1d	1e	1f	1g	1h	2a	2c	2d	2e	2f	3a	3b	3d	3g
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

2b

2g

```
void makeIsolated(int i) {

}
```

2h *On a separate piece of paper.*

2i *On a separate piece of paper.*

3c

3e

Insertion sort:								
Selection sort:								

3f

3h

4a *On a separate piece of paper.*

4b *On a separate piece of paper.*

It is strongly preferred that you fit your answers on these sheets, except for 2h, 2i, 4a, and 4b. If you really must, you can use a separate sheet of paper instead. Please indicate that clearly. Questions 2h, 2i, 4a, and 4b are answered on a separate sheet of paper.