

Duration: **50 minutes**  
Aids Allowed: **NONE** (in particular, no calculator)

Student Number: \_\_\_\_\_

Last (Family) Name(s): \_\_\_\_\_

First (Given) Name(s): \_\_\_\_\_

---

*Do **not** turn this page until you have received the signal to start.*  
(In the meantime, please fill out the identification section above,  
and read the instructions below *carefully*.)

---

This test consists of 2 questions on 6 pages (including this one), printed on one side of the paper. *When you receive the signal to start, please make sure that your copy of the test is complete and write your student number at the bottom of every page, where indicated.*

Answer each question directly on the test paper, in the space provided, and use the reverse side of the pages for rough work. If you need more space for one of your solutions, use the reverse side of a page and *indicate clearly the part of your work that should be marked.*

In your answers, you may use without proof any result or theorem covered in lecture, tutorial, or on assignments. You must justify all other facts required for your solution.

If you are unable to answer a question (or part of a question), you will get 20% of the marks for the question (or part of the question) if you state clearly that you do not know how to answer. Note that you will *not* get those marks if your answer contains contradictory statements (such as “I don’t know” followed or preceded by parts of a solution that have not been crossed off).

MARKING GUIDE

Pr 1: \_\_\_\_\_/15

Pr 2: \_\_\_\_\_/23

TOTAL: \_\_\_\_\_/35

*Good Luck!*

**Question 1.** [15 MARKS]

You are given two *sorted* arrays of length  $n$ ,  $A[1..n]$  and  $B[1..n]$ . All the entries in  $A$  are distinct, and  $A$  is sorted in *increasing* order, so that  $A[1] < A[2] < \dots < A[n]$ . Similarly, all the entries in  $B$  are distinct, and  $B$  is sorted in *decreasing* order, so that  $B[1] > B[2] > \dots > B[n]$ . You are given that there is an index  $i$ , such that  $A[i] = B[i]$ . Note that such an index must be unique.

Give an algorithm that given  $A[1..n]$  and  $B[1..n]$  as above finds the index  $i$ . For full marks, your algorithm should run in time  $O(\log n)$ . An algorithm running faster than the trivial  $O(n)$  will receive partial marks.

Justify briefly that your algorithm is correct and runs within the required time bound.

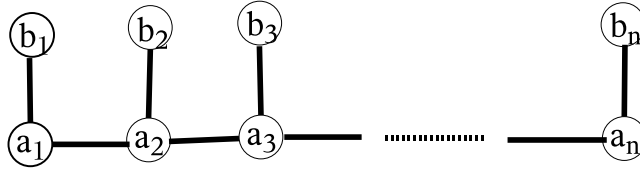
For your reference, the Master Theorem states that a recurrence of the form  $T(n) = aT(n/b) + \Theta(n^d)$  has solution  $\Theta(n^d)$  if  $a < b^d$ ,  $\Theta(n^d \log n)$  if  $a = b^d$ , and  $\Theta(n^{\log_b a})$  if  $a > b^d$ .

**Question 1.** (CONTINUED)

**Question 2.** [23 MARKS]

[20+3 bonus]

Let  $G$  be the graph with  $2n$  vertices and  $2n - 1$  edges presented on the picture below.



Weights  $a_1, a_2, \dots, a_n, b_1, b_2, \dots, b_n$  are assigned to the vertices of  $G$  as on the picture. The goal of the question is to find an independent set of  $G$  that has the maximum total weight.

**Part (a)** [3 MARKS]

Show that the following greedy algorithm does not always return an independent set of  $G$  that has the maximum total weight.

```

S := ∅
while G is not empty:
    pick vi in G with maximum weight
    remove vi and its neighbours from G
    S := S ∪ {vi}
return S
    
```

**Part (b)** [17 MARKS]

Write a dynamic programming algorithm that finds the *weight* of the heaviest possible independent set. You do not have to output the actual independent set. Briefly explain why your algorithm is correct, and state its running time.

**Question 2.** (CONTINUED)

**Part (c)** [3 MARKS]

[BONUS] A friend of yours claims that if  $b_i > a_i$  for all  $i$  (i.e.  $b_1 > a_1, b_2 > a_2, \dots, b_n > a_n$ ), then the greedy algorithm from part (a) solves the problem correctly.

Do you agree? If yes, give a brief proof that this is true, if no give a counterexample.

*Note: the "20%" rule does not apply to the bonus question.*

Total Marks = 35

Student #: \_\_\_\_\_

Page 6 of 6

END OF TERM TEST 2