TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI
VICTORIA
UNIVERSITY OF WELLINGTON

Student ID: ....

EXAMINATIONS — 2011

END-OF-YEAR

COMP 261
ALGORITHMS
and
DATA STRUCTURES

Time Allowed: 3 Hours

Instructions: Attempt ALL Questions.

Answer in the appropriate boxes if possible — if you write your answer elsewhere, make it clear where your answer can be found.

The exam will be marked out of 180 marks.

Non-programmable calculators without a full alphabetic key pad are permitted.

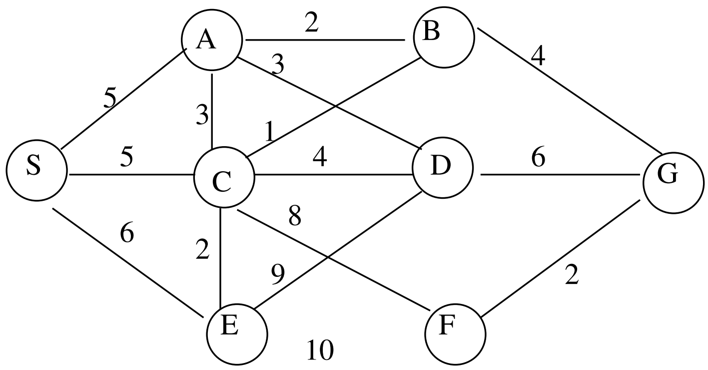
Non-electronic foreign language dictionaries are permitted.

Useful formulas are listed on the last page of the exam.

Questions	**Marks**
1. Shortest Path in Graphs	[15]
2. Minimum Spanning Tree	[13]
3. String Search	[12]
4. Text Processing	[25]
5. Graphics Rendering	[30]
6. File Structures	[33]
7. B-Trees	[32]
8. Hashing	[20]

Question 1. Shortest Path in Graphs [15 marks]

(a) [8 marks] Suppose you are using Dijkstra's algorithm to find the shortest path from **S** to **G** in the graph below. Show the order in which nodes will be *added* to the queue, and the order in which they are *removed* from the queue. In case of a tie, visit the nodes in alphabetic order. When visiting a node, consider the neighbours of the node in alphabetic order.
Hint: Keep track of the queue, along with the priority for each node on the queue.



Nodes Added to Queue:

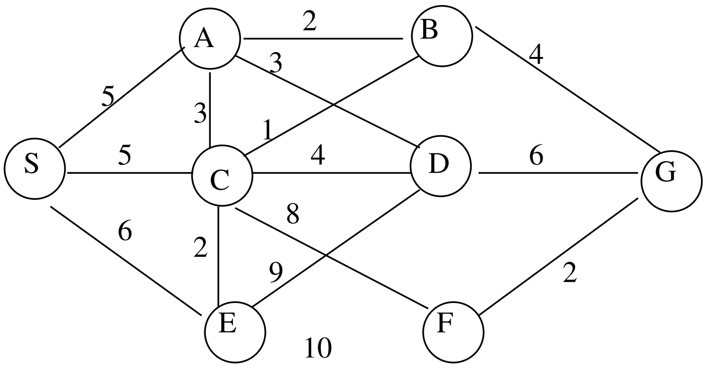
Nodes Removed from Queue:

(b) [7 marks] The A* algorithm for shortest path finding is similar to Dijkstra's algorithm. Explain the key way in which A* differs from Dijkstra's algorithm, and explain why A* is usually better. Give an example to show a special case where A* fails to find the shortest path.

Question 2. Minimum Spanning tree [13 marks]

(a) [8 marks] Suppose you are using Prim's algorithm to find a minimum spanning tree in the graph below, starting from node **S**. Show the order in which *edges* will be added to the tree.

Hint: Keep track of a queue, along with the priority for each edge on the queue. In case of a tie, visit the nodes in alphabetic order. When visiting a node, consider the neighbours of the node in alphabetic order.



Edges added to queue:

Edges added to the tree in this order:

(b) [5 marks] Use an example to explain why Prim's algorithm does not work on a directed graph.

Question 3. String Searching [12 marks]

(a) [7 marks] To search a text for an occurrence of any of a set of words, it is efficient to construct a trie of the words. Draw a trie for the following set of words.
Note: The characters should be attached to the edges in the trie, and all terminal nodes should be indicated clearly.

`cat`	`bat`	`bath`	`bin`	`candy`
`car`	`clean`	`clerk`	`cars`	`hat`

(b) [5 marks] Suppose you are using the KMP algorithm to search for all occurrences of a given string in a text file consisting of 10000 characters with many *a*'s.

Consider each of the following three strings as the input:

(i) `aaaaaaaaab` (ii) `ababcabcde` (iii) `abcdefghij`

Which string is more likely to take the most time? Explain why.

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

Question 4. Text Processing [25 marks]

Consider the following grammar where nonterminals are in uppercase and terminals are enclosed in quotation marks. Assume that tokens will be separated by spaces. Note that nonterminal SMTH uses a regular expression to express the kinds of terminals it accepts.

BAZ ::= `"!"` BAR `"end"` `"world"` | BAR `"end"` `"world"`
BAR ::= `"bca"` FOO | SMTH
FOO ::= SMTH | BAR
SMTH ::= a+b∗c+

(a) [10 marks] For each of the following sentences, state whether it belongs to the language defined by this grammar.

☐	`! bca abc end world`
☐	`bca ac end world`
☐	`! ac end`
☐	`bca ! bca bca abc end world`
☐	`bca ! bca bca abc abc end world`

(b) [5 marks] Can the grammar above be parsed by a predictive, one symbol lookahead, left-to-right (LL(1)) parser? Explain why or why not.

(Question 4 continued on next page)

(Question 4 continued)

(c) Consider a very simple functional programming language defined as follows. An identifier always starts with a character and can be followed by either more characters or digits. An identifier on its own is a variable. An identifier followed by a comma separated list of variables and other function calls is a function call itself. All function calls contains at least one argument.

Here are some sample programs in this language:

```
a

f(g(c(aB,ba),d(a)))
```

(i) [5 marks] Write a grammar for this language that is parseable by an LL(1) (*single character lookahead only*) top down recursive descent parser.

(Question 4 continued on next page)

(Question 4 continued)

(ii) [5 marks] For each of the two examples on the previous page, draw a concrete parse tree derived using your grammar.

SPARE PAGE FOR EXTRA ANSWERS

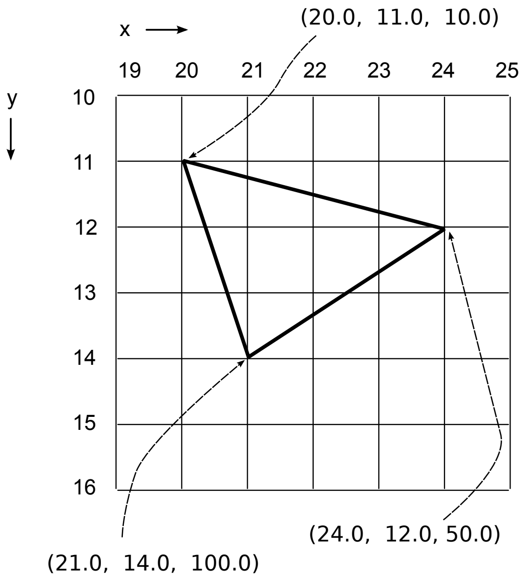
Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

Student ID: ....

Question 5. Graphics Rendering [30 marks]

(a) [10 marks] Show the values in the edge-lists that would be constructed when rendering the polygon shown below. The (x, y, z) coordinates of the vertices are shown.

(Note that the z values are chosen carefully to make the interpolation easy.)



Edge-Lists

	x_{left}	z_{left}	x_{right}	z_{right}
11				
12				
13				
14				

(b) [5 marks] In constructing and using the edge-lists, you had to convert floating point numbers to integers. Explain how this can introduce errors unless you are careful.

(Question 5 continued on next page)

(Question 5 continued)

(c) [10 marks] List all the steps in the 3D Rendering Algorithm described in the lectures that utilised *both* the Z-Buffer and the Edge Lists and explain the purpose of each step. One sentence or so description for each major step would do.

(Question 5 continued on next page)

(Question 5 continued)

(d) [5 marks] Explain what *affine* transformation means and why they are preferred in computer graphics.

Question 6. File Structures [33 marks]

Suppose a file contains 65, 536 fixed length records describing individual patients. Each record has the following fields:

PatientID: (length = 5 characters),

Name: (length = 30 characters),

Illness: (length = 100 characters),

Prescription: (length = 15 characters).

Assume that the file blocks are stored contiguously and that the block size for the file is 600 characters.

(a) [2 marks] Calculate the record size L in characters. Show your working.

(b) [3 marks] Calculate the blocking factor f and the number of file blocks b. Assume an unspanned file organisation. Show your working.

(c) [5 marks] Calculate the *worst case* number of block accesses needed to perform a binary search for a random record in the file given its PatientID. Assume the file is ordered by PatientID. Show your working.

(Question 6 continued on next page)

(Question 6 continued)

(d) [5 marks] Explain the differences between primary file organisation and secondary file organisation.

(Question 6 continued on next page)

(Question 6 continued)

(e) [10 marks] For each of the following file structures, discuss their advantages and disadvantages. You can do it by:

- explaining the efficiency of the different file operations (insertion, deletion, search, sequential access) with different structures, and
- giving examples of when it is appropriate to use each kind of file.

(i) Heap file:

(ii) Sequential file.

(iii) Hash file.

(Question 6 continued on next page)

(Question 6 continued)

(f) [4 marks] Describe in detail (with pictures) the sort and merge stages involved in the Sort-Merge algorithm.

(g) [4 marks] What problems would arise if you try to do it "in place"?

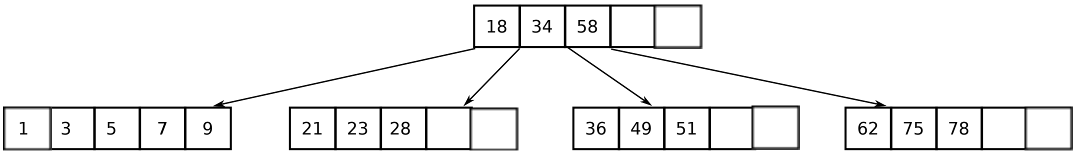
Question 7. B-Trees [32 marks]

(a) [10 marks] Draw an example of a 2-3 Tree with at least 12 integer values in it, *and* state and explain the 2-3 Tree properties that make them distinct from binary trees or general trees.

(Question 7 continued on next page)

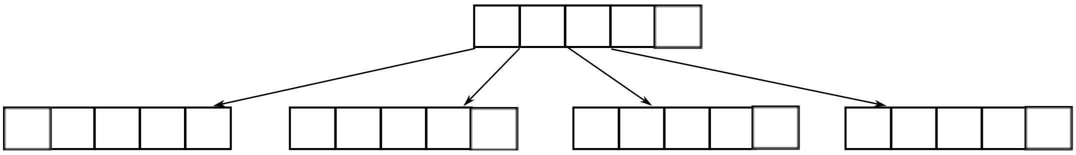
(Question 7 continued)

(b) [12 marks] Consider the B-tree of order 7 illustrated below.

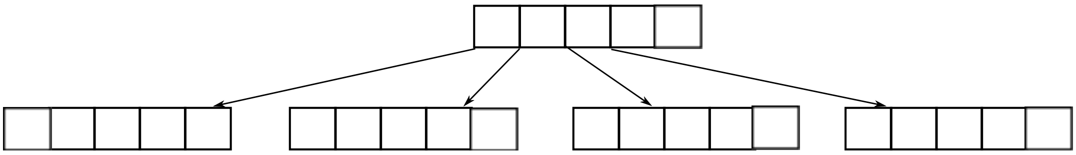


Update the B-tree by successively deleting the key values 62, 34, 21, 18. In your answer, show the B-tree after each deletion and briefly describe what you have done.
Note, the empty trees below are to save you time; you may modify their structure if you choose.

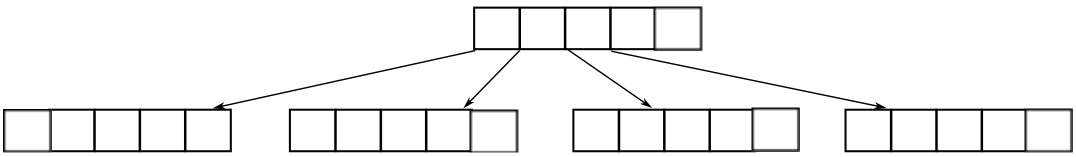
The B-tree after deleting key value 62:



The B-tree after deleting key values 62 and 34:



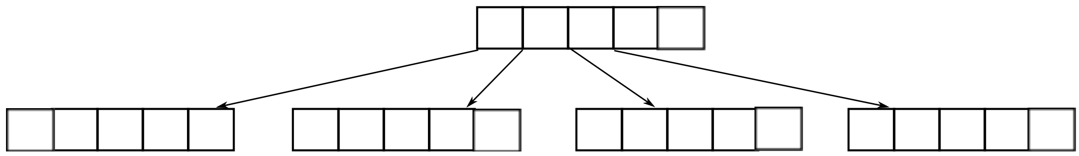
The B-tree after deleting key values 62, 34, and 21:



(Question 7 continued on next page)

(Question 7 continued)

The *B*-tree after deleting key values 62, 34, 21, and 18:



(Question 7 continued on next page)

(Question 7 continued)

(c) [10 marks] Imagine a *B*-tree that has order 5 and starts only a root node. Assume that to begin with the root node contains the values: 5, 10, and 15 (the root node doesn't have any children to begin with). Draw a *B*-tree after the following values are inserted in this order: 1, 2, 3, 4.

The *B*-tree after adding key value 1:

The *B*-tree after adding key values 1 and 2:

The *B*-tree after adding key values 1, 2, and 3:

The *B*-tree after adding key value 1, 2, 3, and 4:

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

Question 8. Hashing [20 marks]

(a) [5 marks] What is a *secondary index* and what can it be used for?

(b) [5 marks] What is the difference between static hashing and dynamic hashing?

(Question 8 continued on next page)

(Question 8 continued)

(c) [10 marks] Describe how extendible hash files work.

****

Student ID: . . . .

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

Useful Formulas

You may tear off this page if you wish. You do not need to hand it in.

File Performance Formulas

- blocking factor: $f = \lfloor \frac{B}{L} \rfloor$
- number of blocks: $b = \left\lceil \frac{r}{f} \right\rceil$
- external sort-merge: $N = 2b \cdot (1 + \lceil (\log_{n-1} b) - 1 \rceil) = 2b \cdot (1 + \left\lceil (\frac{\log_{10} b}{\log_{10}(n-1)} - 1 \right\rceil$
  (where n is the number of buffers)

B-tree (worst case)

- height: $h = 1 + \lfloor \log_{m+1} \frac{r+1}{2} \rfloor = 1 + \left\lfloor \log_{10} \frac{\frac{r+1}{2}}{\log_{10}(m+1)} \right\rfloor$
- number of leaves: $N_{leaves} = 2(m+1)^{h-2} \leq N_{leaves} \leq (2m+1)^{h-1}$

B^+-tree (worst case)

- height: $h = 2 + \lfloor \log_{m+1} \frac{r}{2m} \rfloor = 2 + \left\lfloor \frac{\log_2 \frac{r}{2m}}{\log_2(m+1)} \right\rfloor$
- number of leaves: $N_{leaves} = \lceil \frac{r}{m} \rceil$

Index-Sequential File with a B-tree

- number of sequence sets: $s = \left\lceil \frac{r}{f} \right\rceil \leq s \leq \left\lceil \frac{2r}{f} \right\rceil$

Logs to base 2

n	1	2	4	8	16	32	64	128	256	512	1,024	4096	16384	65536	1,048,576
$\log_2 n$	0	1	2	3	4	5	6	7	8	9	10	12	14	16	20

Logs to base 10

n	5	10	50	100	500	1000	5000	10,000	10^6	5×10^6	10×10^6	50×10^6
$\log_{10} n$	0.7	1	1.7	2	2.7	3	3.7	4	6	6.7	7	7.7