PRIFYSGOL CYMRU; UNIVERSITY OF WALES

DEGREE EXAMINATIONS MAY/JUNE 2002

SWANSEA

Computer Science

CS 121 Data Structures

Attempt 2 questions out of 3

Time allowed: 2 hours

Students are permitted to use the dictionaries provided by the University

Students are NOT permitted to use calculators

CS_121
DATA STRUCTURES
(Attempt 2 questions out of 3.)

Question 1

a) Name and describe the operations which could be carried out upon an abstract data type of *stack.* Give the *data types* and *constants* which we would use for a stack and give *exact specifications* for the operations. Also give two *axioms* and explain what they demonstrate.

[8 marks]

b) Given the definition of a function to calculate the Fibonacci numbers :

```
fib(1) = 1
fib(2) = 1
fib(n) = fib(n-1) + fib(n – 2)
```

write a recursive function to compute fib(x). Give a call trace for computing fib(6). Write an iterative function to compute fib and comment on the efficiency of each function. What type of data structures are suitable for recursion and for what type should recursion be avoided?

[9 marks]

c) Show, perhaps with the aid of a diagram, the difference between a linked list and a two-way linked list. Explain when you might prefer a two way linked list and what advantages the extra pointers give for certain operations.

[4 marks]

d) Describe two methods by which memory is allocated at run time. What sort of problems occur and how can these be overcome? Explain how you would store a list of available memory.

[4 marks]

Question 2

a) Give the Big-O notation for the average case for searching in a binary search tree, a linked list and a hash table. Explain in each case how the complexity is derived.

[6 marks]

b) The table below gives a list of names in the order in which they are read from disk. The next two columns give a primary (p) and secondary (s) hash function for each name. (It is not necessary for you to know how the functions are derived).

Name	P(name)	s(name)
Michael	2	2
Paul	5	2
Yvette	2	3
Neil	1	1
Rebecca	6	1
Linda	5	2

Insert these names into a hash table containing 7 entries indexed 0 to 6 in the order given using separate chaining, linear probing and secondary hashing.

[9 marks]

c) What is meant by *covering* a hash table? What are the two methods we can use to ensure that a table is covered? Why do we need to ensure that a table is covered?

[4 marks]

d) How is the *load factor* of a hash table defined and what does it influence?

[2 marks]

e) Describe Brent's algorithm for hash tables.

[4 marks]

Question 3

a) Build a binary search tree with the following ten items read from disk in the given order : Mary, Petra, Timothy, Neil, Rachel, Lucy, Freda, Brendon, Sarah, Lindon.

[2 marks]

b) Using the following type declaration as a template complete the TreeInsert procedure which inserts a node into a binary search tree.

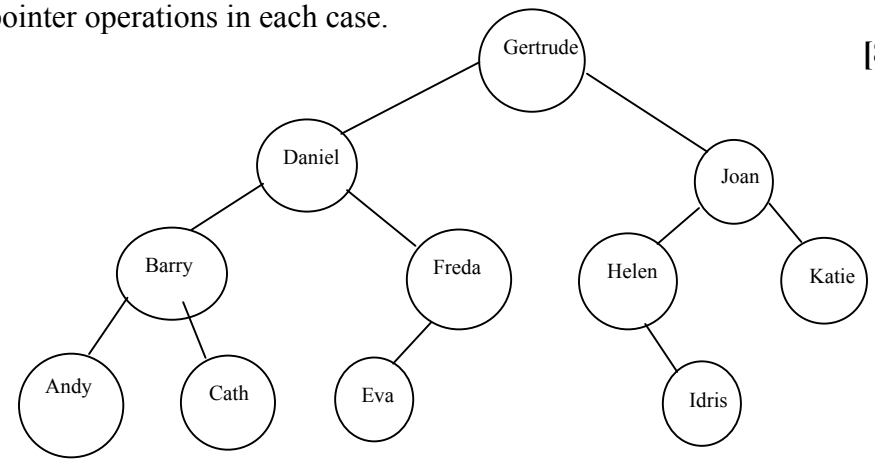
```
NodePtr = ^Node;
Node = Record
  Name : STRING;
   ……….
   ……….
   Left : NODEPTR;
   Right : NODEPTR;
End;

Procedure TreeInsert(var
tree:NODEPTR;NewNode:NODEPTR);
Begin
End;
```

You may assume that a procedure exists which reads the information from disk and sets the left and right pointers to nil.

[6 marks]

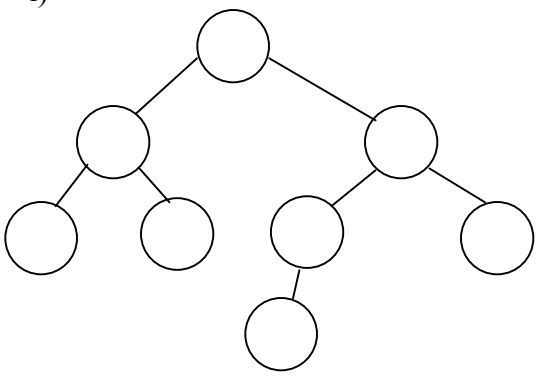
c) There are three different cases for deleting a node from a binary tree. These can be demonstrated in the following tree by the nodes Cath, Freda and Gertrude. Describe each case in full and show the resulting tree after each deletion. (Start with the original tree in each case). Full marks can be obtained by demonstrating the pointer operations in each case.

[8 marks]

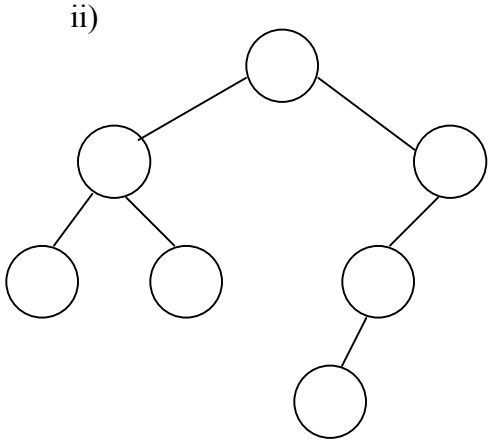


d) What are the benefits of using an AVL tree? Give the balance factor at each node for the two trees below and for each tree explain why it is, or is not, an AVL tree.

i)



ii)



[5 marks]

e) In the following tree A and B are nodes while T_1, T_2 and T_3 are subtrees. Draw the tree after a single right rotation about the root of the tree. Explain why the binary search property remains unaffected by the rotation.

[4 marks]