Family Name:	
Other Names:	
Signature:	
Student Number:	

This PAPER is NOT to be retained by the STUDENT

The University Of New South Wales

C0MP1921/2011/2091 SAMPLE Final Exam

Data Structures and Algorithms/Data Organisation/Computing 2

November 2006

Time allowed: **3 hrs**
Total number of questions: **17**
Total number of marks: **105**

You must hand in this entire exam paper and ALL your answer booklets. Otherwise you will get zero marks for the exam and a possible charge of academic misconduct.

Ensure that you fill in all of the details on the front of this pink paper, and the 5 answer booklets, and then SIGN everything. Mark one booklet PART B, one PART C, one PART D, one PART E, and one WORKING ONLY. The working only booklet will not be marked.

Do not use red pen or pencil in this exam. Unless advised otherwise you may not use any language features or library functions not covered in class. Answers which do not comply with the course style guide or which introduce potential security vulnerabilities will be penalised.

There are two marks for following the examination instructions.

No examination materials permitted.
Calculators may **not** be used.
Questions are **not** worth equal marks.
Answer **all** questions.

Examiner's Use Only:

Inst	Part A	Part B	Part C	Part D	Part E	**Total**

Part A: Short Answer Questions

Answer these questions in the spaces provided on **this pink question paper**. DO NOT answer these questions in an **answer booklet**! Do all your working in the *Working Only* booklet, your working is not marked in Part A.

Write your answers clearly. Keep your answers neat and very brief. Messy or long answers will not be marked. For multiple-choice questions, circle one answer only.

Question 1

(2 marks)
Given the C implementations of sorting algorithms discussed in lectures, which of the following is an nlogn sorting algorithm in the worst case?

[A] Insertion Sort
[B] Quicksort
[C] Bubble Sort with Early Exit
[D] Shell Sort
[E] None of the above

Question 2

(2 marks)
Given the following definition of a stack:

```
struct stack {
    int item;
    struct stack *next;
};

typedef struct stack *Stack;
struct stack *tmp;
```

Which one of the following best "pops" the first element off the top of the stack, returning any used memory to the heap?

```
[A] assert (stack != NULL);
    tmp = stack;
    tmp = tmp->next;
    free(tmp);

[B] tmp = stack;
    stack = stack->next;
    assert (tmp != NULL);
    free(tmp);

[C] assert (stack != NULL);
    stack = stack->next;
    free(stack);

[D] assert (stack != NULL);
    tmp = stack;
    stack = tmp->next;
    free(tmp);

[E] tmp = stack;
    stack = tmp->next;
```

Question 3

(4 marks)
What is an ADT?

What are two advantages of making a type abstract?

1. _____

2. _____

Question 4

(2 marks)
Consider the following C variable definition:

```
char s[] = "hello world";
```

Suppose we wish to make a copy of the string in `s` and have it referred to by the variable `p`. Which of the following C code fragments does this correctly?

[A] `char *p; strcpy(p, s);`

[B] `char *p = NULL; strcpy(p, s);`

[C] `char *p; p = (char *) malloc(strlen(s)); strcpy(p, s);`

[D] All of the above

[E] None of the above

Question 5

(2 marks)
A programmer is wondering which sort algorithm to use in their program and has just asked you about the speed difference between a sort function which uses the insertion sort algorithm and one which uses the quicksort algorithm. Which of the following can you say? (if more than one is correct select the most informative of them).

[A] A quicksort function will always be faster than an insertion sort function

[B] An insertion sort function will always be faster than a quicksort function

[C] A quicksort function will usually be faster than an insertion sort function

[D] An insertion sort function will usually be faster than a quicksort function

[E] None of the above

Part B: Sorting and Analysis of Algorithms

Answer this part in your Part B answer booklet. Start each question on a new page.

Make your answers as clear and easy to understand as possible. Provide type definitions and brief comments where necessary. Confusing or illegible solutions will lose marks.

If you do not wish your answer for a question to be marked, clearly record 1 mark for that question on the front of your Part D answer booklet. If you do this your answer for that question will **not** be marked.

Question 6

(5 marks)

Consider this C function for shell-sort that takes an array of integers `sequence` and an integer `numElements` giving the number of elements in the array `sequence`. (The numbers on the left-hand side are line numbers.)

```
1   void shellSort(int sequence[], int numElements) {
2
3       int i, j, temp;
4
... NOTE BY RICHARD - INCOMPLETE QUESTION FOR SAMPLE EXAM
```

This code fragment is valid C and executes without error. What intervals are currently being used in the shell sort? Suppose you wish to use the intervals 9,4,1 instead. State which line(s) would need to be altered and give the C code that should replace these lines.

Part C: Abstract Data Types

Answer this part in your Part C answer booklet. Start each question on a new page.

Make your answers as clear and easy to understand as possible. Provide type definitions and brief comments where necessary. Confusing or illegible solutions will lose marks.

If you do not wish your answer for a question to be marked, clearly record 1 mark for that question on the front of your Part C answer booklet. If you do this your answer for that question will **not** be marked.

Question 7

(12 marks)

You wish to write an ADT to represent a stack of ints, and an ADT to represent a queue of ints.

(i) Write the file **Stack.h**.
It should include the following prototype:

```
void pop(Stack stack);
```

(and it will need to include other things too!) Don't forget to include brief comments.

(ii) Give a complete definition and implementation of the type `Stack` in C, stating in which file, or files, the code goes.

(iii) Write the file **Queue.h**.
It should include the following prototype:

```
int head(Queue queue);
```

(and it will need to include other things too!) Don't forget to include brief comments.

(iv) Give a complete definition and implementation of the type `Queue` in C, stating in which file, or files, the code goes.

Question 8

(8 marks)

This question refers to the ADTs you wrote in the previous question. You may #include those files in your answers for this question.

(i) Write a function to reverse a Stack. The function is not part of the Stack ADT - suppose it is defined in some other file. State the #include(s) you would need to place at the top of this file.

(ii) Write a function to reverse a Queue. The function is not part of the Queue ADT - suppose it is defined in some other file. State the #include(s) you would need to place at the top of this file.

Part D: Graphs and Trees

Answer this part in your Part D answer booklet. Start each question on a new page.

Make your answers as clear and easy to understand as possible. Provide type definitions and brief comments where necessary. Confusing or illegible solutions will lose marks.

If you do not wish your answer for a question to be marked, clearly record 1 mark for that question on the front of your Part D answer booklet. If you do this your answer for that question will **not** be marked.

Question 9

(7 marks)

Consider the following graph:

`A SAMPLE GRAPH GOES HERE`

State the edges in the Minimum Spanning Tree which would be found by Prim's Algorithm if it started with Node "A". List the edges in the order in which they would be added by the algorithm.

Part E: Challenge Question

Answer this part in your Part E answer booklet.

Partial solutions for this question (i.e. attempts worth less than 50%) will score no marks in this part. If you do not wish your answer for the question to be marked, record 1 mark for the question on the front of the answer booklet. If you do this your answers for the question will **not** be marked.

Your solutions must be clear, elegant and easy to understand. In this part confusing or difficult to understand solutions will score no marks.

Question 10

(12 marks)

Suppose you wish to write a Treap ADT.

Give an abstract type definition for the type Treap which would go in Treap.h. (The tree and heap values will be ints).

Complete the type implementation for treaps, the first line of which is given below:

```
struct treap ...
```

Implement the interface function:

```
void delete (Treap treap, int key);
```

which, given a treap and an integer tree value which is contained in the treap, deletes that element from the treap.