Philadelphia University **Student Name:**

Faculty of Engineering **Student Number:**

Dept. of Computer Engineering

Final exam, First Semester: 2010/2011

Course Title:	**Algorithms and Data Structures**	**Date:**	26/01/2011
Course No:	630231	**Time Allowed:**	2 hours
Lecturer:	Dr. Qadri Hamarsheh	**No. Of Pages:**	4

Information for candidates

1. *This exam paper contains 8 questions totaling 50 marks.*
2. *The marks for parts of question are shown in round brackets.*

Advices to candidates

1. *You should attempt all sub questions.*
2. *You should write your answers clearly.*

Basic notions: *The aims of the questions in this part are to evaluate the required minimal student knowledge and skills. Answers in the pass category represent the minimum understanding of basic concepts: Time complexity of algorithms, Building new data structures like Array List, Linked list, Stack, Queue and Binary Trees, using these data structures with different applications.*

Question 1 **Multiple Choice** ***(10 marks)***

Identify the choice that best completes the statement or answers the question.

1. The three fundamental stages a program goes through are: development, use, and ____.
   a. `implementation` c. `analysis`
   b. `maintenance` d. `requirements gathering`
2. In a(n) ____ copy, two or more pointers of the same type point to the same memory.
   a. `indirect` c. `deep`
   b. `direct` d. `shallow`
3. In an array list the time complexity of the `remove` function is identical to the time complexity of the ____ function.
   a. `insert` c. `isFull`
   b. `isEmpty` d. `maxListSize`
4. If the data needs to be processed in a First In First Out (FIFO) manner, we typically use a(n) ____.
   a. `stack` c. `map`
   b. `queue` d. `hash table`
5. Because initially the list is empty, the pointer `first` must be initialized to _____.
   a. `NULL` c. `NIL`
   b. `EMPTY` d. `NOP`
6. A recursive function executes ____ its iterative counterpart.
   a. `more slowly than` c. `at the same speed as`
   b. `more quickly than` d. `proportionately to`
7. Because all the elements of a stack are of the same type, you can use a(n) ____ to implement a stack.
   a. `struct` c. `record`
   b. `array` d. `class`
8. To speed up item insertion and deletion in a data set, use ____.
   a. `arrays` c. `classes`
   b. `linked lists` d. `ADTs`

9. The height of a perfectly balanced binary tree with n nodes is ____.

a. $\log n$ c. $n \log n$

b. n^2 d. $\log_2 n$

10. The ____ provides class templates to process lists (contiguous or linked), stacks, and queues.

a. STL c. CTL

b. ITL d. ADL

Question 2 ***(3 marks)***

Describe the running time of the following code in **Big O** notation:

a)	`for (int j = 0; j < n; ++j)`
	`{`
	`for (int i = 0; i < j; ++i)`
	`a = a + b;`
	`for (int i = 0; i < n; ++i)`
	`{`
	`c = b + c;`
	`cin >> b;`
	`}}`	

b)	`for (int i = 0; i < 999999; ++i) {`
	`for (int j = 0; j < n; ++j) {`
	`a[i]= a[i] + b + c;`
	`}`
	`for (int k = 0; k < n; ++k) {`
	`cout << i;`
	`}}`	

c)	`for (int i = 0; i < 999999; ++i) {`
	`for (int j = 0; j < n; ++j) {`
	`a[i]= a[i] + b + c;`
	`}`
	`for (int k = 0; k < n; ++k) {`
	`cout << i;`
	`}}`	

Question 3 ***(7 marks)***

a) Give one reason in favor of implementing a stack using an array, and one reason in favor of implementing a stack using dynamic memory allocation via a linked list. **(2 marks)**

b) How do you delete a node from a binary tree that has two child nodes? **(1 mark)**

c) What is the maximum number of comparisons a sequential search and a binary search would need to determine that a particular item is not in a sorted list of three items? 1,023 items? 65,535 items? (fill the table) **(2 marks)**

items	Sequential Search algorithm	Binary Search algorithm
3		
1,023		
65,535		

d) Consider a binary search tree that is built by inserting the sequence:

4 1 9 6 7 2 5 0 8

into an initially empty binary tree.

Draw the tree constructed by this insertion sequence. **(1 mark)**

e) Give a pre-order traversal of your tree shown above (section d) **(1 mark)**

Familiar and Unfamiliar Problems Solving: *The aim of the questions in this part is to evaluate that the student has some basic knowledge of the key aspects of the lecture material and can attempt to solve familiar and unfamiliar problems of Time complexity of algorithms, Building new data structures like Array List, Linked list, Stack, Queue and Binary Trees, using these data structures with different applications and using STL Library.*

Question 4 ***(5 marks)***

Draw the **UML** diagram of the class **binaryTreeType** that specifies the basic operations to implement a binary tree:

- Determine if binary tree is empty.
- inorderTraversal the binary tree.
- PreorderTraversal the binary tree.
- PostorderTraversal the binary tree.
- Find the height of the binary tree.
- Find the number of nodes in the binary tree.
- Find the number of leaves in the binary tree.
- Default constructor.
- Copy constructor.
- Destructor.
- Overloaded assignment operator.
- Copy the binary tree.
- Destroy the binary tree.

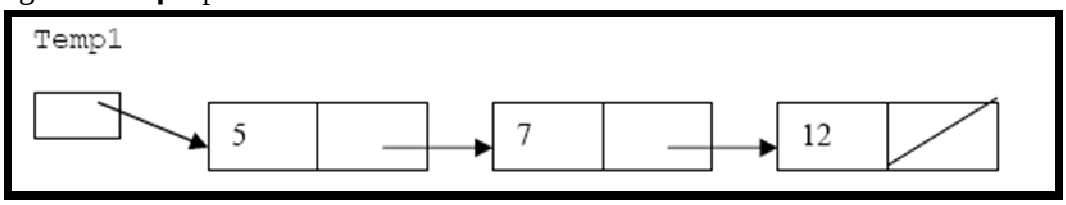
Question 5 ***(6 marks)***

a) Write a nonmember function with the following prototype**:**

```
int compare_linked_lists (LinkedListNode *q, LinkedListNode *r);
```

This function compares two linked lists using recursion algorithm and returns 1 if they are equals and 0 otherwise (assume that the **LinkedListNode** is defined). **(4 marks)**

b) Assuming that **Temp1** points to a list of Nodes that looks like this:



Write code that will insert a new Node containing the value 6 into the list after the Node containing 5 and before the Node containing 7. This new node should be allocated. You do not need to write code that searches for the values 5 or 7. **(2 marks)**

Question 6 ***(5 marks)***

Write a program that uses a stack (STL version) to convert the base-ten representation of a positive integer to base two and print it in the standard output stream.

Question 7 ***(9 marks)***

a) Implement a **queue** ADT in C++. The underlying representation of the queue should be the **Node** class as described below. Your queue should store integers and should handle **errors**. You should implement the **constructor** and the **enqueue** function that inserts value into the queue. **(4 marks)**

```
class Node {
public:
      Node(int value) {val=value; prev = next = NULL;};
      int val;
      Node *prev;
```

```
        Node *next;
    };
    class Queue {
    public:
        Queue(); // constructor
        void enqueue(int value); // insert value into the queue
        int dequeue(); // returns and removes the value at
                       // the head of the queue
        bool isEmpty(); // returns true if the queue contains no elements.
    private:
        Node *head;
        Node *tail;
        int size; };
```

b) Implement a member function **(5 marks)**

`void printQueue (bool chron) const`

`printQueue` should print out all elements in the queue in the direction specified by its parameter. If **`chron`** is true, then print the items out in the order they were originally inserted into the queue. If the **`chron`** is false, then print the items out in reverse order. For example:

```
Q.enqueue (1);
Q.enqueue (2);
Q.enqueue (3);
Q.printQueue(true); // Should print: 1 2 3
Q.printQueue(false); // Should print: 3 2 1
```

Note:When you return from printQueue the original queue should be unchanged!

Question 8 ***(5 marks)***

Write a C++ program that demonstrates a few routines for processing binary sort trees (as shown in the following code). It uses a binary sort tree to store the characters that you enter, after each item is inserted; the contents of the tree are displayed. The number of nodes in the tree is also output and the program ends when the user enters **`'q'`** character.

(Assume that the functions: treeInsert, treeContains, treeList and countNodes are defined)

```
struct TreeNode {
   char item;
   TreeNode *left;
   TreeNode *right;
   TreeNode(char chr = ' ') {
      item = chr;
      left = NULL;
      right = NULL;
   }
};
// Add the item to the binary sort tree.
void treeInsert(TreeNode *&root, char newItem) ;
// Return true if item is one of the items in the binary.
bool treeContains( TreeNode *root, char item ) ;
// Print the items in the tree in postorder.
void treeList(TreeNode *node) ;
// Count the nodes in the binary tree. Return the answer.
int countNodes(TreeNode *node) ;
```

GOOD LUCK