# UNIVERSITI TEKNOLOGI MALAYSIA

**SEMESTER I 2017/2018**

**LAB EXAM**

**SCSJ2013 - Data Structures and Algorithms**

**Friday, December 8, 2017**

**9 am – 11am (2 hours)**

---

## INSTRUCTIONS TO THE STUDENTS:

Answer all tasks based on the question given.

SWITCH OFF YOUR PHONE AND WEB BROWSER DURING THE EXAM

You are given two tasks, Task 1 and Task 2. Answer all questions.
Total marks for the lab exam is 40 marks and the marks for every task are as indicated in the question.

**Lab Test Materials Download Instructions:**

Download the zip file "lab_ Codes" file into your local drive. Unzip the file and it will contain "linkedlist.cpp", "linkedlist_queue.cpp" , "stack1.cpp" and "stack2.cpp".

For each .cpp source files do the tasks as instructed in the next lab exam activity instructions.

**Lab Test Submission Instructions:**

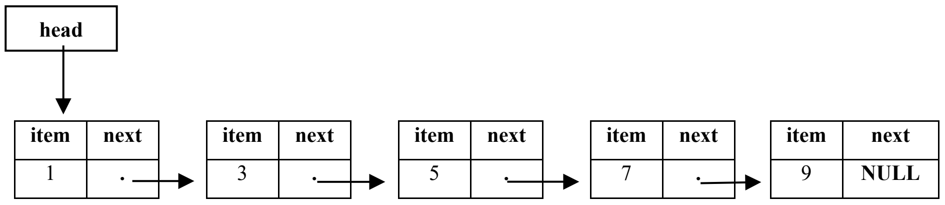
Archive all the C++ source files ("linkedlist.cpp", "linkedlist_queue.cpp" , "stack1.cpp" and "stack2.cpp".) as a zipped file named ""lab_Test_*matricno*.zip". Relpace the *matricno* part of file name with your own matric number. Submit file via UTM's e-learning system.

**Write down your information below.**

| | |
|---|---|
| **Name** | |
| **Identity card (or matric) Number** | |
| **Subject Code and Section** | |

**Task 1 - Linked List and Queue (15 Marks)**

1. Open "linkedlist.cpp" source file. As a guide below is the default linked list structure constructed by the "linkedlist_02.cpp" source file.



a) Inside the "linkedlist_02.cpp" main function find the following comment:

```
// Answer question 1-a after this line [5 Marks]
```

After the comment, add instructions to insert new node with item value 4 in between nodes with item values of 3 and 5.

**[5 Marks]**

b) Still inside the "linkedlist_02.cpp" main function, find the following comment:

```
// Answer question 1-b after this line [3 Marks]
```

After the comment, add instructions to delete node with item value 7.

**[3 Marks]**

2. Open and study how "linkedlist_queue.cpp" source file manipulates Node and Queue classes to handle queue implementation by using linked list. Complete the implementation of deQueue member function of Queue class to correctly extract and delete item from the queue.

**[7 Marks]**

**Task 2 – Stack (25 Marks)**

1. Open "stack1.cpp" and do the following tasks.

   1. In main() function, write source codes for the following:

      Without using push operation, create a stack by inserting p, q, r

      **a.** Insert node pointed by p to Stack : Set top to p **(1 mark)**

      b. Insert node pointed by q to stack, Connect q to top **(2 marks)**

c. Insert node pointed by r to stack, Connect r to top **(2 marks)**

2. In **`popAndPrint(Node *&top)`** function, you need to print item at top and delete the node at top, repeat again, until all nodes in the stack has been deleted. **(5 marks)**

Inside **`while (top != NULL)`** loop write the following source code:

- print the value at top
- using temp pointer, delete node at top
- Make sure your program produce the following output:

```
Items at p : 1
Items at q : 2
Items at r : 3

Items at top stack : 3 and popped
Items at top stack : 2 and popped
Items at top stack : 1 and popped
All items in the stack have been deleted
List is empty!
```

Figure 1: Expected output produced by stack1.cpp

2. Open "stack2.cpp" and do the following tasks regarding palindrome.

Palindrome is a word, phrase, or sequence that reads the same backwards and forward. Sample palindrome words are pop, malam, katak, anna, civic, kayak, level, madam, mom, noon and racecar

In **`main()`** function, write source codes for the following:

a. Write loop to extract char in word1 and push each char in word1 into stack. Your output should be as in the following figure.

```
Push operation succesfull, item=[r] added on top of stack!
Push operation succesfull, item=[a] added on top of stack!
Push operation succesfull, item=[c] added on top of stack!
Push operation succesfull, item=[e] added on top of stack!
Push operation succesfull, item=[c] added on top of stack!
Push operation succesfull, item=[a] added on top of stack!
Push operation succesfull, item=[r] added on top of stack!
```

**(4 marks)**

b. Write loop to pop stack char by char and put each char into word2[].Your output should be as in the following figure.

```
Pop operation successful: r
Pop operation successful: a
Pop operation successful: c
Pop operation successful: e
Pop operation successful: c
Pop operation successful: a
Pop operation successful: r
```

**(3 marks)**

c. Complete **`verifyPalindrome()`** function to check whether a word is a palindrome

- Write loop to compare each character in both array, if not the same break/exit the loop, and set isPalindrome to false **(4 marks)**
- Print the following output according to the format example:

If the word is malam : malam and malam is palindrome

and if the word is cat : cat and tac is not palindrome **(4 marks)**

Make sure your program produce the following output in Figure 2.

```
Push operation succesfull, item=[r] added on top of stack!
Push operation succesfull, item=[a] added on top of stack!
Push operation succesfull, item=[c] added on top of stack!
Push operation succesfull, item=[e] added on top of stack!
Push operation succesfull, item=[c] added on top of stack!
Push operation succesfull, item=[a] added on top of stack!
Push operation succesfull, item=[r] added on top of stack!
Pop operation successful: r
Pop operation successful: a
Pop operation successful: c
Pop operation successful: e
Pop operation successful: c
Pop operation successful: a
Pop operation successful: r
racecar and racecar is palindrome
```

Figure 2: Expected output produced by stack2.cpp

```
List original nodes start from header:
1 3 5 7 9
List nodes after node 4 is added:
1 3 4 5 7 9

List nodes after node 7 is deleted:
1 3 4 5 9

Delete 1
Delete 3
Delete 4
Delete 5
Delete 9
```

Expected Output for linkedlist.cpp

```
enQueue item A
Queue  => A
enQueue item B
Queue  => A B
enQueue item C
Queue  => A B C
deQueue item A
Queue  => B C
deQueue item B
Queue  => C
enQueue item D
Queue  => C D
deQueue item C
Queue  => D
deQueue item D
Queue  =>
Can't deQueue item, queue is empty.
deQueue item
Queue  =>
```

Expected Output for linkedlist_queue.cpp