UNIVERSITI TEKNOLOGI MALAYSIA

SEMESTER I 2014/2015
FINAL EXAM

CODE OF SUBJECT	**: SCSJ2013**
NAME OF SUBJECT	**: DATA STRUCTURES AND ALGORITHMS**
YEAR / COURSE	**2**
TIME	**: 2 HOURS**
DATE	**:**
VENUE	**:**

INSTRUCTIONS TO THE STUDENTS:

This test book consists of **TWO** (2) parts. The layout is as follows:

PART A – 3 Questions

PART B – 2 Questions

Answer **ALL Questions** in PART A (3 questions) and select **ONLY ONE** (1) Question from Part B. Altogether you need to **answer 4 questions**.

ANSWER ALL QUESTIONS IN THE TEST BOOKLET.

Name	
Identity card (or matric) Number	
Name of Lecturer	
Subject Code and Section	

This examination book consists of 11 printed pages EXCLUDING this page.

PART A
~~THIS PART CONSISTS OF THREE (3) QUESTIONS. ANSWER ALL QESTIONS IN~~ THIS PART.

Question 1 **[12 marks]**

Given **DATA** array of integers as in Figure 1, answer question a) and b).

index	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
	250	270	111	135	98	100	125	170	190

Figure 1: **DATA** array

a) Name the searching technique suitable to be performed on the data array in Figure 1. Give the reason to justify your answer.

[2 marks]

b) What should be done to the **DATA** array in Figure 1 to reduce the complexity time of the searching process? Explain your answer.

[2 marks]

Given **NUM** array of integers as in Figure 2, answer question c), d) and e).

index	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]
	1	3	5	7	9	11	13	17	19

Figure 2: **NUM** array

c) Perform binary search for searching **key 20** on the **NUM** array in Figure 2. Show the tracing of your search using variables **left**, **right**, **middle, NUM[middle]** and **found** as shown in the table format below.

[3 marks]

left	right	middle	NUM[middle]	found

d) Perform binary search for searching **key 5** on the **`NUM`** array in Figure 2. Show the tracing of your search using variables **`left`**, **`right`**, **`middle, NUM[middle]`** and **`found`** as shown in the table format below.

[3 marks]

left	right	middle	NUM[middle]	found

e) Discuss the differences between the number of steps and the complexity time when searching for **key equals to 19** on **`NUM`** array in Figure 2 using binary search and sequential search technique.

[2 marks]

Question 2 **[19 marks]**

a) A queue can be implemented using circular array or circular linked list data structure. Give one advantage of queue using circular array and one advantage of queue using circular linked list. What kind of problem that both implementation can solve.

[2 marks]

b) Give the differences between the implementation of the queue implementation array and queue implementation circular array in the following operations:

i) Queue declaration

ii) **`isEmpty()`** function.

iii) **`isFull()`** function.

[3 marks]

c) Figure 3 shows the current content of a circular queue.

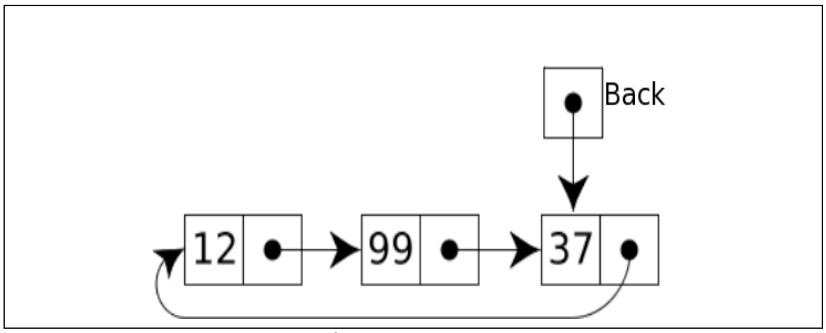

Figure 3: The current queue content

Redraw the queue in Figure 4 that shows the current content of the queue after the sequence of instructions in Figure 4 is executed.

[3 marks]

```
Queue -> enQ(9);
Queue -> enQ(4);
Queue -> deQ();
```

Figure 4 : Queue instructions

Answer the following questions, which deal with operations on queue implementation array. Figure 5 shows Queue 1 with maximum content of 4 items. Currently, there are two items already in the queue after **`enQueue (A)`** and **`enQueue (B)`** operations have been executed.

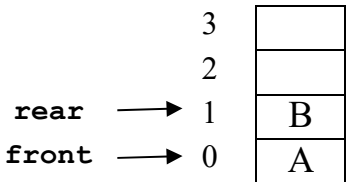

Figure 5: Queue 1

d) Perform the following tasks to Queue 1:

i. Redraw Queue 1 after **`enQueue(G)`** followed by **`enQueue(F)`.** Label the correct location of the **`front`** and **`rear`**.

ii. Redraw the queue in question (d- i) after **`deQueue()`** and **`enQueue(G)`**. Label the correct location of the **`front`** and **`rear`**.

[3 marks]

e) Figure 6, shows a circular queue called Queue 2, which fits maximum of 8 items. The current content of Queue 2 are as shown in the figure.

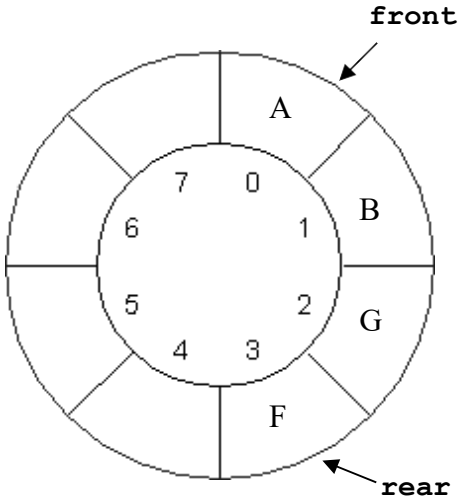

Figure 6: Queue 2

i. Redraw Queue 2 after **E, J, M, H** are **enQueue** in sequence. Label the correct location of the **`front`** and **`rear`**.

[2 marks]

ii. Redraw updated queue in question (i) after **3 deQueue** operations are performed . Label the correct location of the **`front`** and **`rear`**.

[2 marks]

iii. List all the deQueue items in question (ii) in FIFO sequence.

[1.5 marks]

iv. Redraw updated queue in (ii) after **R and Q** are **enQueue** in sequence. Label the correct location of the **`front`** and **`rear`**.

[1 marks]

v. After a series of **enQueue** and **deQueue** operations, compare the efficiency of memory usage for Queue1 and Queue2.

[1.5 marks]

Question 3 **[19 marks]**

a) Discuss the advantage of full tree and complete tree in terms of complexity related to insertion, deletion and searching operations. [2 marks]

b) Discuss the drawbacks of Binary Search Tree in terms of complexity related to insertion, deletion and searching operations. [2 marks]

c) Draw the expression tree for the following arithmetic expression. [3 marks]

a * b – (c * 11 / d) + (e + f * 5)

d) Give the inorder, preorder and postorder traversal of the tree in Figure 7. [6 marks]

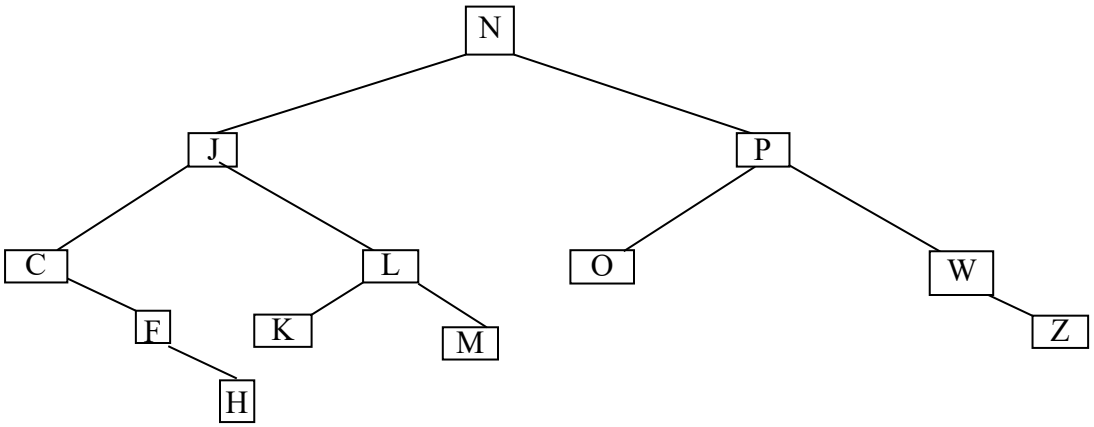

Figure 7: Binary Search Tree of char value

Given the following binary search tree in Figure 8, answer question 5 and 6.

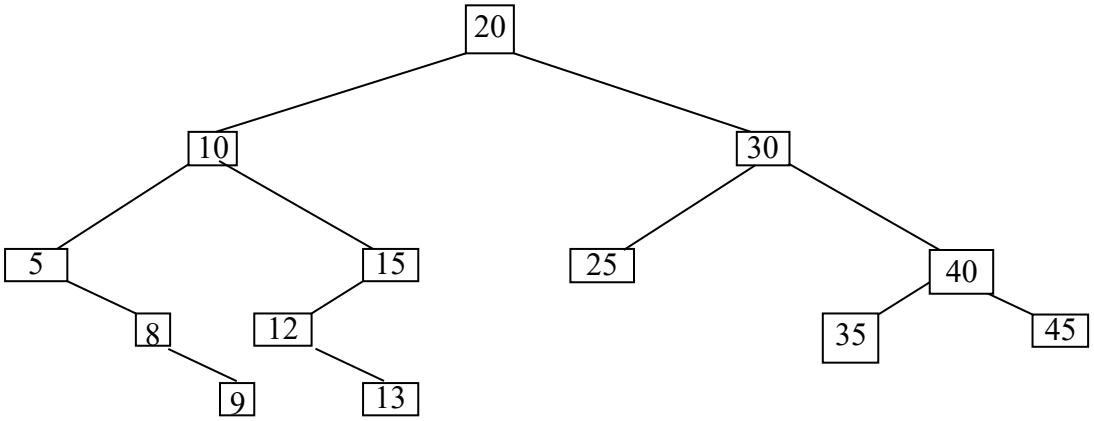

Figure 8: Binary Search Tree of int value

e) Redraw the tree in Figure 8 after nodes with values **18, 16, 19, 33 and 43** are inserted in sequence. [2 marks]

f) Redraw the tree in Figure 8 after the following nodes are deleted **in sequence**. Show the new tree after every deletion.

i) 25

ii) 20

iii) 10

[4 marks]

PART B

THIS PART CONSISTS OF 2 QUESTIONS. SELECT AND ANSWER ONLY ONE (1) QUESTION IN THIS PART.

Question 1 **[20 marks]**

You are given the following code as shown in Figure 9. When the code execute, the result can be represented in Figure 10.

```
struct NODE {
    int item;
    NODE* prev;
    NODE* next;
};

// main function
int main(int argc, char *argv[]) {
    NODE* n1 = new NODE;
    n1->item = 6;
    n1->prev = NULL;
    n1->next = NULL;

    //assume the additional code segment for question a – d is here

}//end main
```

Figure 9 : Linked List Codes

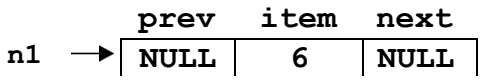

Figure 10: Current execution state with node **n1**

a) Show in a diagram, how nodes **n1, n2** and **n3** are interconnected to each other, when the following code segment is executed.

[3 marks]

```
NODE* n2 = new NODE;
n2->item = 3;
n2->prev = n1;
n2->next = NULL;

NODE* n3 = new NODE;
n3->item = 8;
n3->prev = n2;
n3->next = NULL;
```

b) Show in a diagram, how nodes **n1, n2** and **n3** are interconnected to each other, when the following code segment is executed.

[3 marks]

```
n1->next = n2;
n2->next = n3;
```

c) A new node, **n4** is created as follows:

```
NODE* n4 = new NODE;
n4->item = 7;
n4->prev = NULL;
n4->next = NULL;
```

Show in a diagram, how nodes **n1, n2, n3** and **n4** are interconnected to each other, when the code is executed

[3 marks]

d) Write a code segment to delete node n2 and replace it with the node n4

[3 marks]

e) Another code segment is added as follows:

```
NODE * print_ptr;
print_ptr = n1;
```

Write a code segment to display the value of item in all the nodes, using **`print_ptr`**

[3 marks]

f) Bill tracker application called "Bills Reminder" need to be developed. The Bills Reminder has a basic functions that store types of bill payment, bill payment amount and the monthly due date. If linked list is used as a data structure to be used in designing the application, describe how linked list can be used as the data structure that able to store the information of the bill every month as shown in Table 1.

Table 1: Sample of bills, the amount, and due date for one month

Bill	**Amount**	**Due date for every month**
Celcom	40	1
Electricity	30	1
Credit card	900	16
House rent	200	29

[5 marks]

Question 2 **[20 marks]**

a) Consider the following source codes in Program Segment 1 that will be implemented on stack using array. The **size of the stack is 5**.

```
1   Stack s;
2   string a1, a2, a3;
3   :
4   :
5   a1=s.stackTop( );
6   a2= "Luck";
7   s.push(a2);
8   s.push(a1+a2);
9   s.push("Congratulations");
10  a2=s.stackTop( );
11  s.pop();
12  s.pop();
13  s.push("Bye!");
14  a3=s.stackTop( );
15  s.pop();
16  cout<< a1 << a3 <<endl;
17  cout<< a2 <<endl;
```

Program Segment 1: Stack source codes

i) Show the content of the stack after each implementation of the sources codes based on the situation below. Assume the current content of the stack before statement Line 5 is executed as shown in first stack in the Table 2. [3 marks]

Element of array	Content of Stack						
[4]							
[3]							
[2]	**Good**						
[1]	**Bye!**						
[0]	**Thank You**						
	Current content before Line 5	Content After Line 5-8 are executed	Content After Line 9	Content After Line 11	Content After Line 12	Content After Line 13	Content After Line 15

Table 2

ii) What will be displayed by the program in Program Segment 1? [2 mark]

b) Using stack operations, show step-by-step conversion of the following infix expression to postfix notation. Use the format given in the table below.

A – B * (C + D * E) / F

[5 marks]

Infix	Stack	Postfix

c) Evaluate the following postfix expression using stack concept: **3 6 – 10 + 5 ***
You need to use the format given in the table below in order to show the evaluation process. [4 marks]

Postfix	Ch	Opr	Opn1	Opn2	Result	Stack

d) A compiler need to keep track of a return address of next statement to be executed when return from a function call as well as all parameters passed to the calling function. An example of a program (partial codes) is given in Figure 11, in which a main function calls function A. Function A calls function B, which then calls function C. Bellow are steps for the complier when executing the program:

step	i.	the main function calls function A
step	ii.	function A calls function B
step	iii.	function B calls function C
step	iv.	completion of executing function C and returns to function B
step	v.	completion of executing function B and returns to function A
step	vi.	completion of executing function A and returns to main function

address	partial code
1000	int main() {
1004	int a, b, c;
....	.....
1050	function A(a,b,c);
1054	cout << a << b << c << endl;
	.....
1100	}

address	partial code
200	function A(int a, int b, int c) {
204	int d, e;
....	.....
300	function B(a,b,d,e);
304	cout << a << b << c << d << e << endl;
	.....
500	}

address	partial code
100	function B(int a, int b, int c, int d) {
104	int e, f;
....	.....
150	function C(a,d,e,f);
154	cout << a << b << c << d << e << f << endl;
	.....
198	}

address	partial code
800	function C(int a, int b, int c, int d) {
804	int g, h;
....	.....
900	cout << a << b << c << d << g<< h<< endl;
904	.....
	cout << "end of nested call" << endl;
998	}

Figure 11: Partial codes for function call

i) What is the most suitable data structure that the compiler should use in keeping track of returned addresses and parameters when executing the program.

[1 mark]

ii) Draw a diagram to show the content of the data structure for each step of execution (steps i – iv) in illustrating the process. Based on the diagram, provide description of the operation on the data structure.

[5 mark]