



INDIVIDUAL ASSIGNMENT

Name : ZHANG ZITENG
Student ID : TP052096
Intake : UC2F1908IS
Lab Group : Group 1

This is an open-notes, open-Internet, and open-book non coding assignment.
The only thing you aren't allowed to do is consult with other people about the assignment (except for *your module lecturer*).

Rules of this assignment:

1. Posting questions on online forums counts as consulting with other people, and is thus prohibited.
2. If you use the information obtained online or from a book, cite your sources. You must rewrite the collected information based on your understanding and using your own sentences.
3. You must include the scanned or photographed hand-drawn diagrams in this individual assignment. Diagrams that drawn by the electronics software are not allowed.
4. You must make sure all the hand-drawn diagrams are neat and easy to understand.

If you do not follow any of the above rules, you will be deducted 5-20% of the total score as a penalty.

The assignment is divided into 4 questions. Each question carries 50 marks, which the total score of this assignment is 200 marks. You have to answer **ALL** of the questions in this word document file. Once you answer all the questions here, convert this word document file become a pdf file and submit to the Moodle system.

To pass this module, you should get more than 50% of the total score.

Details of submission:

You have to submit the softcopy of this assignment to your module lecturer through the online Moodle system.

The submission due date of this assignment is **17 June 2020**, before **5.00p.m.**

Late submission of the assignment will receive 0 points.

Question 1:

Answer ALL of the sub questions below.

Question 1 carries 50 marks.

- a) *Memory management* in operating system is one of the popular application that using the stack concept. Based on your understanding, explain in details how the stack concept working in computer memory. You can include any related diagram in your explanation.

You may click on the grey colour sentence of “*Click here to enter text*” to add your explanation.

[15 marks]

Answer:

In general, the stack automatically allocates variables, stores function parameter values, and local variable values. It is some space used when the function is called. Its address decreases from high to low.

The working principle is that the variable is automatically assigned first, it is fast but the programmer cannot control it. For example, declare a local variable `int b` in the function; the system automatically creates space for `b` on the stack.

Secondly, if there is enough remaining space of the stack, memory will be provided to the system, or there will be an exception and the stack will overflow.

At last, stack is a data structure of a continues area of memory that extends to a lower address. The top address and the maximum capacity of the stack are predetermined by the system. For example, suppose the capacity of the stack is set to 2M at beginning. Overflow will happen if the requested space is bigger than the remaining space on the stack.

(GeeksforGeeks,2014)

b) **Figure 1** shows empty stack 1 and stack 2.

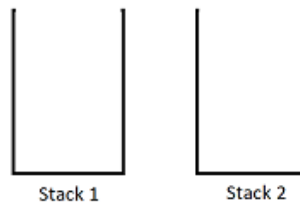



Figure 1. Empty Stack 1 and Empty Stack 2

List the final contents of the stack1 and stack2 after the following operations. **Top()** function is used to reference the top(or the newest) element of the stack.

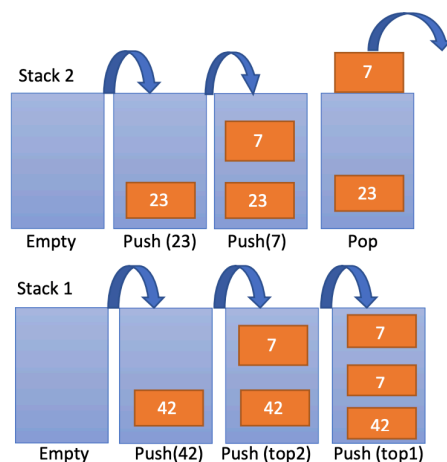
```
stack2.push(23);
stack2.push(7);
stack1.push(42);
int top1 = stack1.top();
int top2 = stack2.top();
stack1.push(top2);
stack2.pop();
stack1.push(top1);
```

You are required to draw the step-by-step diagrams to show the above operations, until to your final contents. Include the correct caption and arrows in each of the diagrams.

All your hand drawing diagrams must paste in the given answer section in *page 3*. You may click on the button  to add more columns based on your needs.

[15 marks]

Answer:



The final content:

Stack 1: {7,7,42}

Stack 2: {23}.

c) Given the following infix expression:

$$A+B-F*(C*D-E)$$

Apply stack concept and transform the above expression into its postfix and prefix form by using the sample table style in **Table 1** and **Table 2** below.

Infix Expression : A+B-F*(C*D-E)				
Token	Action	Result	Stack	Notes
A	Add A to the result	A		
+	Push + to the stack	A	+	
B	Add B to the result	AB	+	
-	Push - to the stack	AB+	-	-same priority as +. Thus, pop + and push -
Postfix Expression:				

Table 1. Sample table to transform from Infix expression to Postfix expression.

Infix Expression : A+B-F*(C*D-E)				
Reverse Infix Expression :)E-D*C(*F-B+A				
Reverse brackets: (E-D*C)*F-B+A				
Token	Action	Result	Stack	Notes
Prefix Expression (Reversed result):				

Table 2. Sample table to transform from Infix expression to Prefix expression

You may click on the grey colour sentence of “*Click here to enter text*” to add your hand drawing pictures / tables.

[20 marks]

Your answer:

Infix Expression: A+B-F*(C*D-E)				
Token	Action	Result	Stack	Notes
A	Add A to result	A		
+	Push + to stack	A	+	
B	Add B to result	AB	+	
-	Push - to stack	AB+	-	-same priority as +, thus, pop + and push -
F	Add F to result	AB+F	-	
*	Push * to stack	AB+F	*-	
(Push (to stack	AB+F	(*-	
C	Add C to result	AB+FC	(*-	
*	Push * to stack	AB+FC	*(*-	
D	Add D to result	AB+FCD	*(*-	
-	Pop * from stack and add to result	AB+FCD*	(*-	-has lower precedence than *
	Push - to stack	AB+FCD*	-(*-	
E	Add E to result	AB+FCD*E	-(*-	
)	Pop -from stack and add to result	AB+FCD*E-	(*-	Do process until (is popped from stack
	Pop (from stack	AB+FCD*E-	*-	
	Pop * from stack and add to result	AB+FCD*E-*	-	Given expression is iterated, do the process until stack is not empty, it will give final result.
	Pop – from stack and add to result	AB+FCD*E-*-		
Postfix Expression: AB+FCD*E-*-				

Infix Expression: $A+B-F*(C*D-E)$				
Reverse Infix Expression: $E-D*C (*F-B+A)$				
Reverse brackets: $(E-D*C) *F-B+A$				
Token	Action	Result	Stack	Notes
(Push (to stack		(
E	Add E to the result	E	(
-	Push - to stack	E	(-	
D	Add D to the result	ED	(-	
*	Push * to stack	ED	(-*	
C	Add C to the result	EDC	(-*	
)	Pop * from stack and add to result	EDC*	(-	Do process until (is popped from stack
	Pop - from stack and add to result	EDC*-	(
	Pop (from stack	EDC*-		
*	Push * to stack	EDC*-	*	
F	Add F to the result	EDC*-F	*	
-	Pop * from stack and add to result	EDC*-F*		- has lower precedence than *
	Push - to stack	EDC*-F*	-	
B	Add B to the result	EDC*-F*B	-	
	Pop - from stack and add to result	EDC*-f*B-		
+	Push + to stack	EDC*-F*B-	+	
A	Add a to the result	EDC*-F*B-A	+	
	Pop + from stack and add to result	EDC*-F*B-A+		Given expression is iterated, do the process till stack is not empty. It will give the final result.
Prefix Expression (Reversed result): $+A-B*F-*CDE$				

Question 2:

Answer **ALL** of the sub questions below.

Question 2 carries **50 marks**.

a) Problem Scenario:

Shopping list (Item/Priority);

- Eggs (4)
- Bread (2)
- Milk (6)
- Water (3)
- Meat (1)
- Detergent (5)

Figure 1 Shopping list

Assume that you have a shopping list as shown in Figure 2, and your mum is telling you to grab them in 15 minutes. She gives you also priorities, so you need to grab them first. You gotta rush!

Question:

- i. Discuss the theory of **Quick Sort** algorithm in detail. You can include any related diagram in your explanation.
- ii. Based on the given scenario, by implement the **Quick Sort** algorithm, discuss how the algorithm help in reordering the items in **Figure 2** by priority, by comparing them on first element which is eggs. You can include any related diagram in your explanation.

You may click on the grey colour sentence of “*Click here to enter text*” to add your explanation.

[30 marks]

Answer:

i.

The basic logic of quick sorting is to first take an element from the sequence as a pivot. There are many types of selection methods for this element, you can choose the first, last, random or median. The logic of different methods is similar, the difference lies in the different algorithms. Partition this element to the left and right. During the partitioning process, place all the elements that are larger than this element on the right, and those that are less than or equal to it on the left. Then repeat the partitions for the left and right sections until each section has only one number.

ii.

Assume the shopping list is a group of element as follow:					
Egg	Bread	Milk	Water	Meat	Detergent
4	2	6	3	1	5
First Element is Egg (4).					
First sort:					
Bread (2)	Water (3)	Meat (1)	Egg (4)	Milk (6)	Detergent(5)
<= egg (4)				> egg (4)	
Group 1				Group 2	
Assume the first element for group 1 is bread (2).				Assume the first element for group 2 is Milk (6).	
First sort for group 1:				First sort for group 2:	
Meat (1)	Bread (2)	Water (3)		Detergent(5)	Milk (6)
<= Bread (2)		> Bread (2)		< Milk (6)	
So the final result by using quick sort is:					
Meat (1)	Bread (2)	Water(3)	Egg(4)	Detergent(5)	Milk(6)

The first element is Egg(4). So set Egg(4) as pivot. All the element smaller or equal 4 go to left, all the element bigger than 4 go to right. Then we divide the elements into 2 groups. Which is group 1(smaller or equal to 4) and group 2(bigger than 4). Then repeat the last step which is to set an element for each group as their pivot. And sort again. Finally, we can get the result: Meat(1) -> bread(2) -> water(3) -> egg(4) -> detergent(5) -> milk(6)

b) Problem Scenario:

Queue and Stack are both different data structures with different properties and also different use cases. However, it is able to implement the Queue data structure by only using the help of the Stacks, and vice versa.

Question:

Discuss how the Queue data structure can be implemented by using the help of Stack. You must include any related diagram in your discussion.

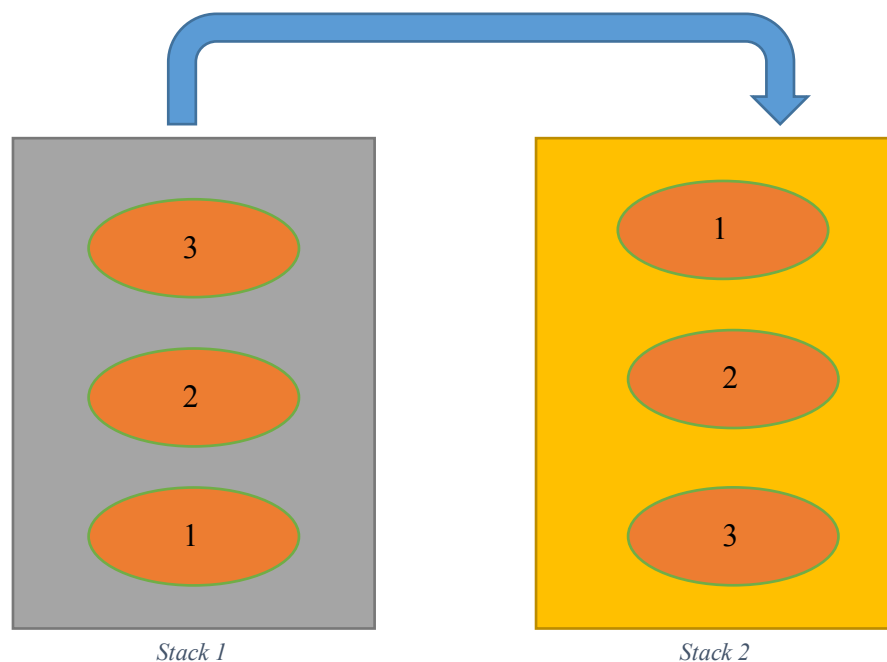
You may click on the grey colour sentence of “*Click here to enter text*” to add your explanation.

[20 marks]

Answer:

The main feature of the stack is first in / last out. The main feature of the queue is first in / first out. So using the stack to realize the queue can also be understood as the process of converting first in last out into first in first out.

For example, a group of numbers from low to high are 1 2 3. If the stack is used, the data extraction order is 3 2 1, and the queue extraction order is 1 2 3. However, if you add another stack. Put the array of stack 1 into stack 2 in turn, then the order of stack 2 from low to high will become 3 2 1 as you see in the following stack 2 image. The data extraction order of stack 2 becomes 1 2 3. This indirectly realizes the first in first out which is queue data structure.



Question 3:

Answer ALL of the sub questions below.

Question 3 carries 50 marks.

a) Determine each of the following statements is true or false. Justify each of your answers with a proper discussion. You can include any related diagram in your discussion.

- i. *Full binary tree is also called as complete binary tree. Both are the same.*
- ii. *Binary Tree is not equivalent to Binary Search Tree. There are differences between them.*
- iii. *AVL tree is better than BST.*
- iv. *We can convert an infix expression to become a postfix or a prefix expression using a Binary Expression Tree.*

You may click on the grey colour sentence of “*Click here to enter text*” to add your explanation.

[20 marks]

Answer:

i. False

Every node except leaves must have two children for the full binary tree.

Differently, each level other than the last should be completely filled and all of its nodes must be as far left as possible for the complete binary tree.

ii. True

Binary tree is a non linear data structure. Every node except leaves must have two children.

Binary search tree, all values in its left subtree must be smaller than the node, and all the values in its right subtree must be bigger than the node.

iii. True

AVL tree can access data more efficiently than BST. AVL is a balanced BST, the node is 1 bigger than its Left Sub Tree and 1 smaller than the Right Sub Tree. Thus, the height of AVL tree when rotation is always $O(\log n)$.

iv. True

They are all kind of data structure. However, expression tree is a binary tree, its internal node corresponds to operator and each leaf node corresponds to operand which makes it convertible. (GeeksforGeeks,2015)

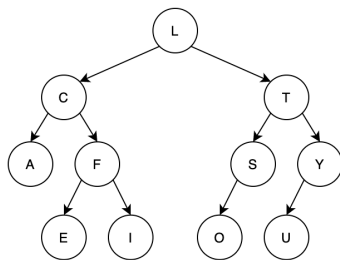
- b) Imagine starting with an empty binary search tree (BST) whose nodes have characters for keys, and adding the letters in the word “**F A C E T I O U S L Y**” to the tree in that order.
- Show the resulting tree.
 - Evaluate the “*post-order traversal*” answer of this **BST**.
 - If you search for the letter ‘**R**’ in the **BST** in question 3b (i), to which letters will you compare **R**, and in what order? Discuss your answer.
 - Draw the binary search tree after the node **T** is deleted. Clarify your answer with a proper justification.

You may click on the grey colour sentence of “*Click here to enter text*” to add your explanation.

[30 marks]

Answer:

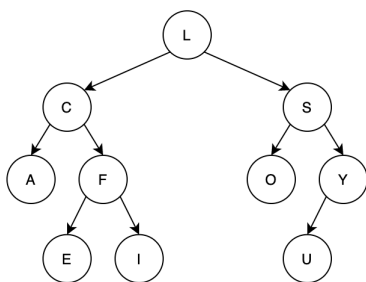
- i. adding the letter in the word FACETIOUSLY



- ii.
Post order traversal: A E I F C L O S U Y T L

- iii.
To search letter R, firstly it will go right of the node L, because R is greater than L. Then because R smaller than T, so go to the left of subnode T. Next, because R is smaller than S, it will go to the left of S. Lastly, because O is the leaf of S, so compare R with O.

- iv.



The node T to be delete has nonempty left and right subtrees. Then delete node T, change to S, S change to O. Now S is the node instead of T.

Question 4:

Answer ALL of the sub questions below.

Question 4 carries 50 marks.

a) Look at the below statements, and answer the questions:

*“An example of the **minimum spanning tree** is a cable company wanting to lay line to multiple neighbourhoods; by minimizing the amount of cable laid, the cable company will save money. There are a few **popular algorithms** for finding this minimum distance, such as **Kruskal’s algorithm** and **Prim’s algorithm**”.*

- i. Discuss in details on the minimum spanning tree.
- ii. You have to choose one of the popular algorithms from the statements above, and discuss how it works by using the cable company example.

You may click on the grey colour sentence of “Click here to enter text” to add your explanation.

[30 marks]

Answer:

i.

Spanning tree is a acyclic graph that include connected edges and vertex as a collection. A graph may include more than one such spanning trees. The value of cumulative edge weights is the smallest, it it a minimum spanning tree. which means a acyclic graph that touches every vertex by using the least path.

ii.

I want to choose Kruskal's algorithm to discuss how it works in this case.

First step is to sort all the lines among multiple nerighbourhoods in non-decreasing order of their weight.

Second step is to pick the smallest cable laid. Check if it forms a cycle with the spanning tree formed so far. If the cycle is not formed, this line will be included. Otherwise discard it.

Repeat step 2 until the number of lines among each nerighbourhoods equals to the number of all the nerighbourhoods minuse one (V-1). That is the final Minimum Spanning Tree(MST).

- b) World Air Alliance (WAA) is an alliance of several airlines that operate flights from their hubs to different destinations. The airlines and destinations in WAA are listed below.

Airlines	Hubs	Destinations	Duration (hours)	Fare (RM)
Tony Airlines	KUL	DXB	7	1925.06
	KUL	IST	11	3025.00
	KUL	JFK	19.3	2137.30
Longitude Airways	DXB	ICN	8.5	1500.00
	DXB	KUL	8	2900.50
	DXB	IST	4.8	1012.50
	DXB	LHR	7.7	3500.45
Tiger Air	ICN	KUL	6.1	1350.00
	ICN	LHR	12	3516.00
Jojo Wings	LHR	JFK	7.4	2257.50
	LHR	DXB	7	2107.00
	LHR	IST	4	1204.52
Monster Airlines	IST	LHR	6.6	2312.50

Based on the above information:

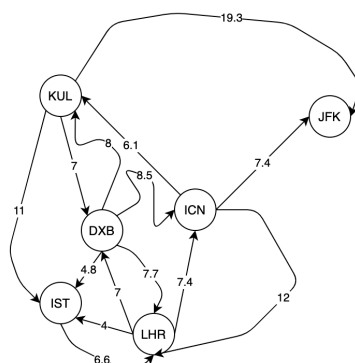
- Draw the directed graph based on the flight duration.
- Produce the adjacency matrix based on the flight fares.
- Find the shortest flight duration from KUL to all other possible destinations using appropriate algorithm. Show your works.

You may click on the grey colour sentence of “*Click here to enter text*” to add your explanation.

[20 marks]

Answer:

i.



ii.

Base on binary

	KUL	DXB	ICN	LHR	IST	JFK
KUL	0	1	0	0	1	1
DXB	1	0	1	1	1	0
ICN	1	0	0	1	0	0
LHR	0	1	0	0	1	1
IST	0	0	0	1	0	0
JFK	0	0	0	0	0	0

Base on flight fares:

	KUL	DXB	ICN	LHR	IST	JFK
KUL	0	1925.06	0	0	3025.00	2137.30
DXB	2900.50	0	1500.00	3500.45	1012.50	0
ICN	1350.00	0	0	3516.00	0	0
LHR	0	2107.00	0	0	1204.52	2257.50
IST	0	0	0	2312.50	0	0
JFK	0	0	0	0	0	0

iii.

The best algorithm to find the shortest distance is Dijkstra's Algorithm.

Firstly, find all the distance as sets from KUL to each one.

KUL->DXB {7,24.6}

KUL->IST {11,11.8,18.7}

KUL->LHR {14.7,17.6,27.5}

KUL->ICN {15.5,21.4,25}

KUL->JFK {19.3,22.9,29.5,32.4}

The set from shortest to longest is

{KUL,DXB,IST,LHR,ICN,JFK}

The shortest path tree is as follow:

