Datastructures Lab Task II/IV B.Tech

Batch-1: 09-November-2021 @ 9.30 to 12

Batch-2: 11-November-2021 @ 8.40 to 11.10

Task-1

We have discussed linear search algorithm in the class. Implement the same algorithm using any of the languages like C/Python. Write all the possible testcases. Write down your observations.

Sample input

Enter number of elements 8

Enter 8 elements

44,16,18,164,47,10,0,-68

Enter element to search 10

Sample output

10 is present at location 5

Task-2

Suppose you are implementing Linear search algorithm using any of the languages like C/Python. In the given array if there is a possibility of multiple occurances of some elements. In such case how to identify the location of the element. Design and implement the solution for the same.

Sample input

Enter number of elements 6

Enter 6 elements

44,16,18,16,47,16

Enter element to search 16

Sample output

16 is present at location 1

16 is present at location 3

16 is present at location 5

Task-3

Rithick gets a lottery ticket and checks each number in the list to see whether he has won the lottery or not. Since there are many numbers,he finds it tedious to check each ticket number manually. So he decides to write a code to check whether he has won the lottery or not. Help Rithick write a code to find his lottery ticket number from the given ticket numbers.

Input Format:

First line of the input consists of n, that corresponds to total number of lottery tickets.

Next n lines consists of Integers, that corresponds to the given lottery ticket numbers.

Last line consists of an Integer 'l', which corresponds to Rithick's lottery ticket number.

Output Format:

Output consists of string "Congratulations! You have won the lottery" or "Sorry your ticket number is not there. Better luck next time", according to the search result.

Sample Input and Output:

[All text in bold corresponds to input and the rest corresponds to output]
Enter the total number of tickets:
5
Enter the ticket number 1:
4521
Enter the ticket number 2:
3589
Enter the ticket number 3:
147852
Enter the ticket number 4:
2365
Enter the ticket number 5:
8965
The ticket numbers are:
4521 3589 147852 2365 8965
Enter the ticket number to be searched:
8965
The ticket number 8965 is found at position 5
Congratulations!You have won the lottery

Datastructures Lab Task II/IV B.Tech

Batch-1: 16-November-2021 @ 9.30 to 12

Batch-2: 18-November-2021 @ 8.40 to 11.10

Task-1

We have discussed Binary search algorithm in the class. Implement the same algorithm using any of the languages like C/Python. Write all the possible testcases. Write down your observations.

Sample input

Enter number of elements 8

Enter 8 elements

44,16,18,164,47,10,0,-68

Enter element to search 10

Sample output

10 is present at location 5

Task-2

Once there lived a king and a queen. They were very rich and led a happy life.

The King's brother Humayun was jealous of him and he wanted to become the King. So in order to become the King, Humayun carries the Queen away and keeps her in a jail which are numbered in sorted sequence.

Humayun informs the King that he took away the Queen and if the King agrees him to become the King, he would leave the Queen. The King has a special power of flying. He flies and reaches the place where Humayun has hidden the Queen and Humayun has given clue to King about the jail number of the Queen. Luckily the King finds the key of the jail where the Queen is locked.But Humayun might have tried to fool the King by saying wrong jail number of Queen. The King already knew how many jails were there. He reaches the centre jail and checks whether the jail number and the key number are equal. He also carries a small machine with him to find the next jail to be visited by him. The machine initially contains the low(low=1) and the high value(high=total number of jails), and calculates the mid value (the jail which is to be visited). If the mid value and key value are equal, the King opens the lock and takes the Queen away with him. If the key number is greater than the mid value, the King increases the mid value by 1 in the machine and assigns it to the low value. If the key number is less than the mid value, the King decreases the mid value by 1 in the machine and assigns it to the high value. The King does this until he finds that the key value and the mid value to be equal.

Sample Input and Output1:
Enter the total number of jails in Humayun's Place:
5
Enter the jail number 1

6
Enter the jail number 2
89
Enter the jail number 3
91
Enter the jail number 4
105
Enter the jail number 5
200
Enter the clue given by Humayun:
105
Hurray!The King rescued the Queen

Sample Input and Output 2:
Enter the total number of jails in Humayun's Place:
6
Enter the jail number 1
56
Enter the jail number 2
68
Enter the jail number 3
156
Enter the jail number 4
196
Enter the jail number 5
204
Enter the jail number 6
895
Enter the clue given by Humayun:
4
The King has been fooled by Humayun

Task-3

Given an Array A of N of integers.You have to find number of subarrays having Primicity less than or equal to K

Primicity of sub-array is defined an number of primes in that subarray.

Input:

First line contains N and K

Second line contains N integers.

Output:

Print the Number of sub-arrays having Primicity<= K

Datastructures Lab Task II/IV B.Tech

Batch-1: 23-November-2021 @ 9.30 to 12

Batch-2: 25-November-2021 @ 8.40 to 11.10

Task-1

In a class room the teacher decides to shuffle all the students so that each student collaborates with other students too and she also checks that board is visible to all students. In a bench n number of students can be seated. She calls the students and makes them stand in a line.

She checks first two person and if the first person is taller than the second person he is put in second place and second in first, i.e., their positions are swapped. After swapping, the teacher checks second and third person height and if the second person is taller than third person their positions are swapped. She repeats this till the end. She finds that the last student is the tallest student after completing one iteration.

So she repeats the process for (n-1) times so that all students are sorted. She stops the process in the middle as soon as she find that the students are sorted.

For ex:

Consider 5 number of students can be seated in a bench

Their height order is in

6 4 3 2 5

During first comparison, adjacent numbers 6 and 4 is compared, 6 is greater than 4 hence the order becomes 4 6 3 2 5.

Again the teacher checks 6 and 3. 6 is greater than 3 hence the numbers are swapped and the order becomes 4 3 6 2 5.

She compares 6 and 2 and they get swapped and the order becomes 4 3 2 6 5. She compares 6 and 5 they get swapped and the order becomes 4 3 2 5 6.

She sees that they are not in order she again takes adjacent person heights and she swaps. She keeps doing this until the order becomes 2 3 4 5 6.

Help the teacher write a program to do bubble sorting inorder to sort the height in order.

For ex:

Consider 5 number of students can be seated in a bench

Their height is in order

6 4 3 2 5

During first comparison, adjacent numbers 6 and 4 is compared, 6 is greater than 4 hence the order becomes 4 6 3 2 5.

Again the teacher checks 6 and 3, 6 is greater than 3. Hence the numbers are swapped and the order becomes 4 3 6 2 5.

She compares 6 and 2 and they get swapped and the order becomes 4 3 2 6 5. She compares 6 and 5, they get swapped and the order becomes 4 3 2 5 6.

She sees that they are not in order. So she again compares adjacent person heights and she swaps. She repeats this until the heights get sorted. Finally the students will be seated in this order - 2 3 4 5 6.

Help the teacher write a program to do this task.

Input Format:Input consists of n+1 integers. The first integer corresponds to n, the number of elements in the array. The next n integers correspond to the elements in the array.

Output Format:Refer sample output for formatting specs.

Task-2

Two friends Reema and Rita decide to play cards. But Reema doesn't know to play the cards. So Rita decides to teach her how to play cards. Rita distributes 13 cards to each other which are in unsorted order and teaches her how to make set with the cards without considering the symbols.
Rita first takes two cards and checks which card has the lesser number and the card with the least number is put first and larger at second.
She takes third card and checks it with second card. If the third card number is less than the second card she inserts the third card at the second position and checks newly inserted second card with the first card. If the newly inserted second card number is less than the first card, she inserts the second card at the first place and this goes on till all the cards get sorted.
Rita finally learns how to order the cards.

Write a program to perform insertion sort on an array of n elements.

Input Format:
Input consists of n+1 integers. The first integer corresponds to n, the number of elements in the array. The next n integers correspond to the elements in the array.

Output Format:
Refer sample output for formatting specs

Task-3

Its the first day for the students at school and the students enter the class and get seated at random places without any height order. So the students who are short and sitting back are not able to see the board since they sit behind taller students.

Understanding this difficulty, the teacher decides to make the students sit in height order.

Suppose there are n students in the class. She makes all the students to stand in a line and compares the first student's height with the remaining (n-1) students. If the first student's height is greater than the ith student, then the taller person goes to the ith place and ith student comes to the first place. Again the new first student's height is compared with remaining students and if his height is greater than ith student the first student goes to ith place and ith place student comes to first place and this goes on till the end.

This process continues for all the students. Finally the students are in height order.

Write a program to perform selection sort on an array of n elements.

Input Format:Input consists of n+1 integers. The first integer corresponds to n, the number of elements in the array.The next n integers correspond to the elements in the array.

Output Format:Refer sample output for formatting specs.

Datastructures Lab Task II/IV B.Tech

Topic: External Sorting

Batch-1: 30-November-2021 @ 9.30 to 12

Batch-2: 2-December-2021 @ 8.40 to 11.10

Task-1

We have discussed sorting algorithm in the class. Implement the same algorithm using any of the languages like C/Python. Given a group of unordered elements. Design an algorithm/pseudocode based on divide and conquer to make the elements in to sorted list. Write all your observations. Eloborate the testcases.

Note: Use the Logic discussed in the class

Algorithm as below

```
MergeSort(arr[], l,  r)
If r > l
     1. Find the middle point to divide the array into two halves:
             middle m = l+ (r-l)/2
     2. Call mergeSort for first half:
             Call mergeSort(arr, l, m)
     3. Call mergeSort for second half:
             Call mergeSort(arr, m+1, r)
     4. Merge the two halves sorted in step 2 and 3:
             Call merge(arr, l, m, r)
```

Test case-1

Enter number of Elements 12

Apply the process of merge sort and sort the following list of elements:

Mar, May, Nov, Aug, Apr, Jan, Dec, Jul, Feb, Jun, Oct, Sep

Test case-2

Enter number of Elements

6

Enter array of elements

12,45,10,33,55,50

Sorted array is

10,12,33,45,50,55

Task-2

We have discussed sorting algorithm in the class. Implement the same algorithm using any of the languages like C/Python. Follow the following LOC.

Sample Input and Output

Enter number of Elements

6

Enter array of elements

12,45,10,33,55,50

Sorted array is

10,12,33,45,50,55

```
Quicksort(A,l,r) // Sort A[l..r-1]

   if (r - l <= 1)) return; // Base case

   // Partition with respect to pivot, a[l]
   yellow = l+1;
   for (green = l+1; green < r; green++)
      if (A[green] <= A[l])
         swap(A,yellow,green);
         yellow++;

   swap(A,l,yellow-1); // Move pivot into place

   Quicksort(A,l,yellow);  // Recursive calls
   Quicksort(A,yellow+1,r);
```

Data structures Lab Task II/IV B.Tech

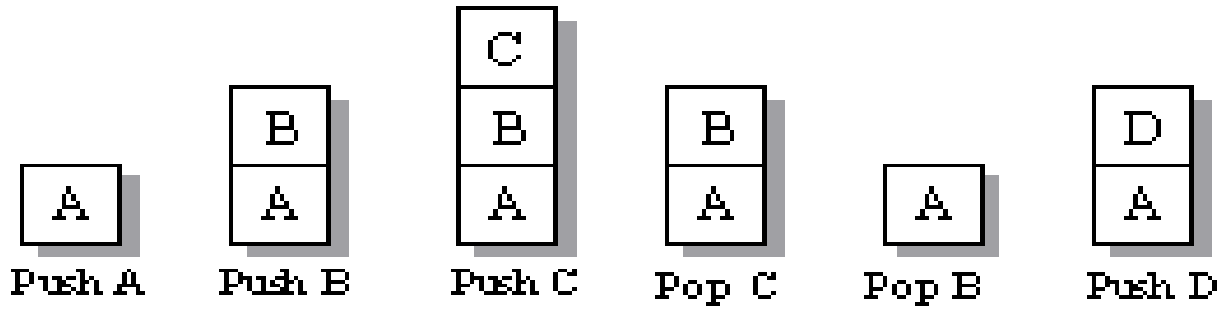
Batch-1: 07-December-2021 @ 9.30 to 12

Batch-2: 09-December-2021 @ 8.40 to 11.10

Task-1

We have discussed stack operations called push () and pop () in the class. Also illustrated with examples. Design and implement menu driven C program with 4 operations. (1) Add element to stack (2) Delete element from stack (3) Traverse elements and (4) Exit. Write all possible examples supported by relevant test cases.

Sample test case is as below:



Task-2

Design and implement C program to check a string is palindrome or not using a stack. A stack is LAST IN FIRST OUT (LIFO) data structure. The element which is inserted last, is accessed first. Insertion and deletion of elements happens only at top of the Stack. The sequence of exit of elements from a stack is reverse of the sequence of their entry in stack.

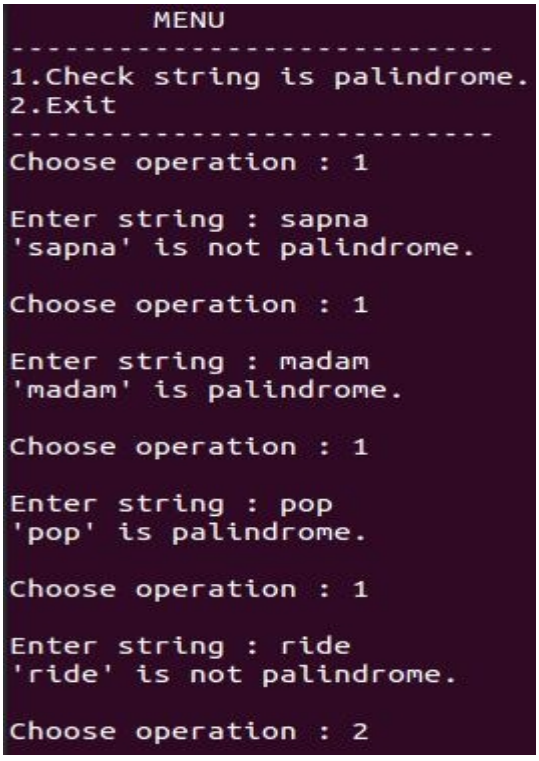
Sequence of Entry.
A --> B --> C -- > D --> E
Sequence of Exit.
E --> D --> C --> B --> A

Algorithm to check palindrome string using stack

- Find the length of the input string using strlen function and store it in a integer variable "length".
- Using a for loop, traverse input string from index 0 to length-1 and push all characters in stack.
- Remove (Pop) characters from stack one by one using a for loop and compare it with corresponding character of input string from beginning(traverse from index 0 to length-1). If we found a mismatch the input string is not a palindrome string otherwise palindrome string.

Write all possible examples supported by relevant test cases.

Sample test case is as below:



Task-3

You have an empty sequence, and you will be given

queries. Each query is one of these three types:

1 x -Push the element x into the stack.
2 -Delete the element present at the top of the stack.
3 -Print the maximum element in the stack.

Function Description

Complete the *getMax* function in the editor below.

getMax has the following parameters:
- *string operations[n]:* operations as strings

Returns
- *int[]:* the answers to each type 3 query

Input Format

The first line of input contains an integer, The next lines each contain an above mentioned query.

Constraints

All queries are valid.

Sample Input

```
STDIN   Function
-----   --------
10      operations[] size n = 10
1 97    operations = ['1 97', '2', '1 20', ....]
2
1 20
2
1 26
1 20
2
3
1 91
3
```

Sample Output

```
26
91
```

Datastructures Lab Task II/IV B.Tech

Batch-1: 14-December-2021 @ 9.30 to 12

Batch-2: 16-December-2021 @ 8.40 to 11.10

Task-1

In a class room we have discussed the stack operations called push() and pop(). Based on these operations design and implement C/Python code to evaluate postfix expression. Hint: Use the following statement in the push function stack[top]=(int)(post[tmp]-48); for ASCII value conversion from string to numbers. For examples please refer http://www.btechsmartclass.com/data_structures/postfix-evaluation.html

Sample Input-1:

Postfix Expression 5 3 + 8 2 - *

Sample Output-1:

Postfix Expression Evaluation Value is 48

Sample Input-2:

Postfix Expression 2 3 4 + * 6 -

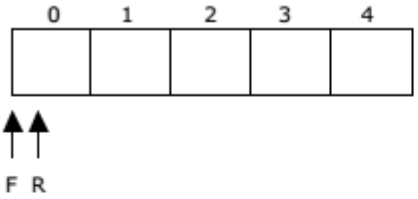
Sample Output-2:

Postfix Expression Evaluation Value is 8

Task-2

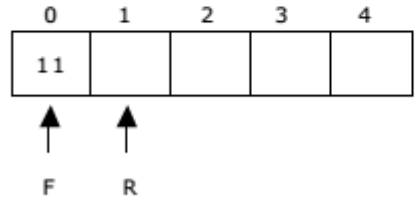
We have discussed Queue operations called insert() and delete() in the class. Also illustrated with examples. Design and implement menu driven C program with 4 operations. (1) Add element to Queue (2) Delete element from Queue (3) Traverse elements and (4) Exit. Write all possible examples supported by relevant test cases.

Use the following Example for Reference



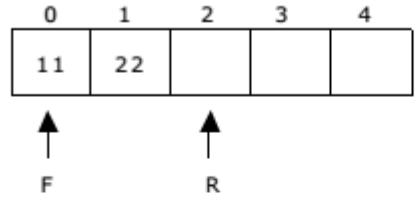
Queue Empty
FRONT = REAR = 0

Now, insert 11 to the queue. Then queue status will be:



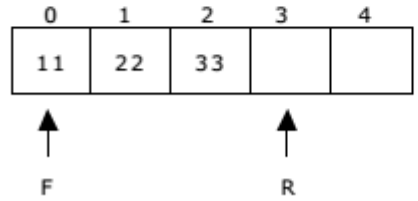
REAR = REAR + 1 = 1
FRONT = 0

Next, insert 22 to the queue. Then the queue status is:



REAR = REAR + 1 = 2
FRONT = 0

Again insert another element 33 to the queue. The status of the queue is:



REAR = REAR + 1 = 3
FRONT = 0

Datastructures Lab Task II/IV B.Tech

Topic: Applications of Stacks&Queues

Batch-1: 21-December-2021 @ 9.30 to 12

Batch-2: 23-December-2021 @ 8.40 to 11.10

Task-1

In a class room we have discussed the stack operations called push() and pop(). Based on these operations design and implement the solution to identify the balanced and un-balanced expressions.

Sample Input-1

The expression like [(a+b)(a-b)]

Sample output-1

is known to be balanced

Sample Input-2

The expression like [(a+b)(a-b] is known to be un-balanced

Sample output-2

is known to be un-balanced

Design algorithm and rules for the same. Use C/Python program for implementation.

Task-2

We have discussed linked list operations called insert() and delete() in the class. Also illustrated with examples. Design and implement menu driven C program with 4 operations. (1) Add element to list (2) Delete element from list (3) Traverse elements and (4) Exit. Write all possible examples supported by relevant test cases.

Data structures Lab Task II/IV B.Tech

Topic: Dynamic Stack and Queue

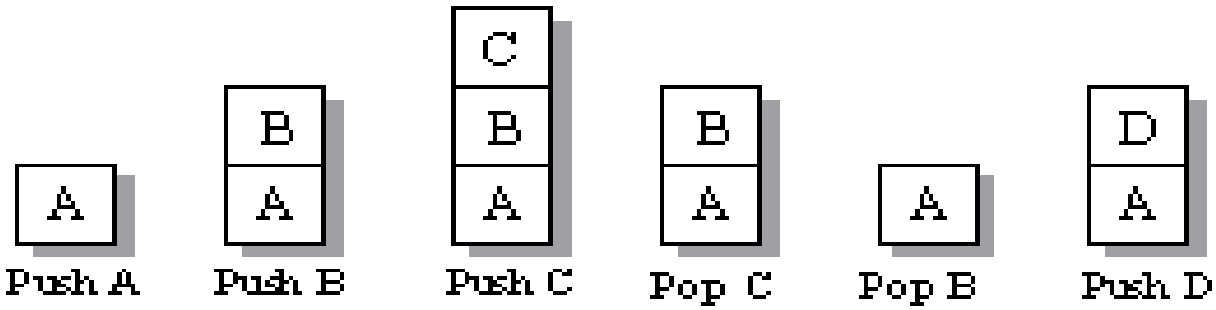
Batch-1: 28-December-2021 @ 9.30 to 12

Batch-2: 30-December-2021 @ 8.40 to 11.10

Task-1

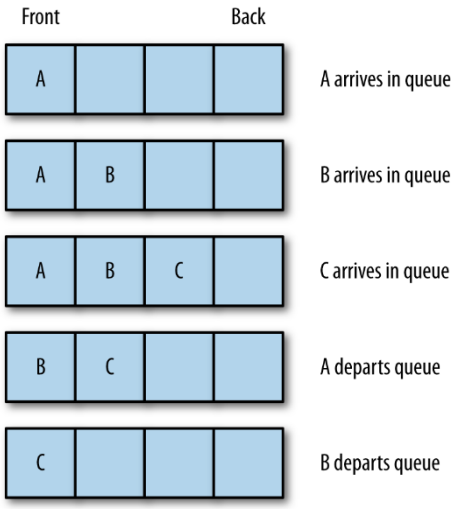
We have discussed stack operations called push () and pop () in the class. Also illustrated with examples. Design and implement menu driven C program with 4 operations. (1) Add element to stack (2) Delete element from stack (3) Traverse elements and (4) Exit. Write all possible examples supported by relevant test cases. (Use Dynamic Implementation for character data)

Sample test case is as below:



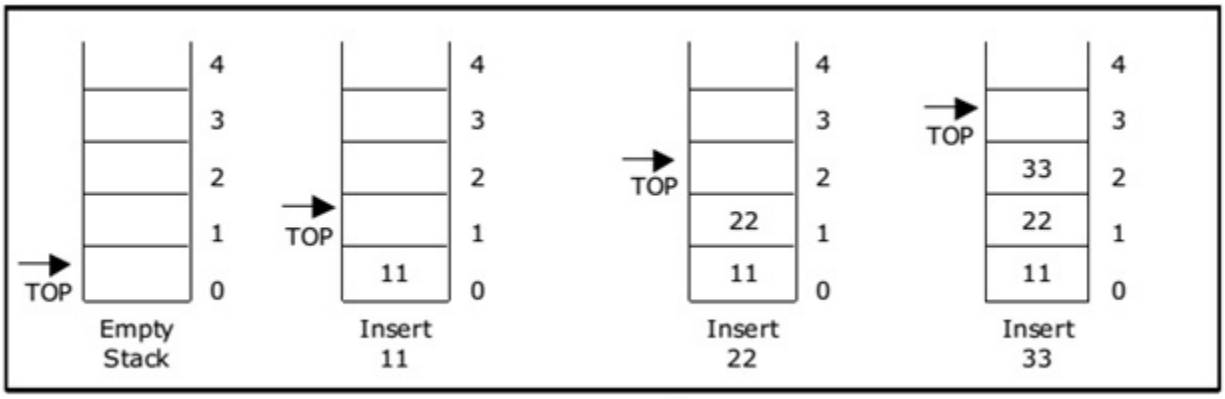
Task-2

We have discussed queue operations called insert () and delete () in the class. Also illustrated with examples. Design and implement menu driven C program with 4 operations. (1) Add element to Queue (2) Delete element from Queue (3) Traverse elements and (4) Exit. Write all possible examples supported by relevant test cases. (Use Dynamic Implementation for character data)



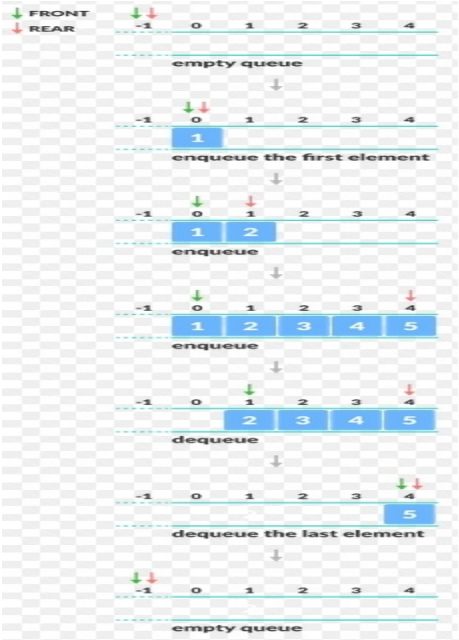
Task-3

We have discussed stack operations called push () and pop () in the class. Also illustrated with examples. Design and implement menu driven C program with 4 operations. (1) Add element to stack (2) Delete element from stack (3) Traverse elements and (4) Exit. Write all possible examples supported by relevant test cases. (Use Dynamic Implementation for integer data)



Task-4

We have discussed queue operations called insert () and delete () in the class. Also illustrated with examples. Design and implement menu driven C program with 4 operations. (1) Add element to Queue (2) Delete element from Queue (3) Traverse elements and (4) Exit. Write all possible examples supported by relevant test cases. (Use Dynamic Implementation for integer data)



Datastructures Lab Task II/IV B.Tech

Topic: Applications of LinkedStacks&Queues

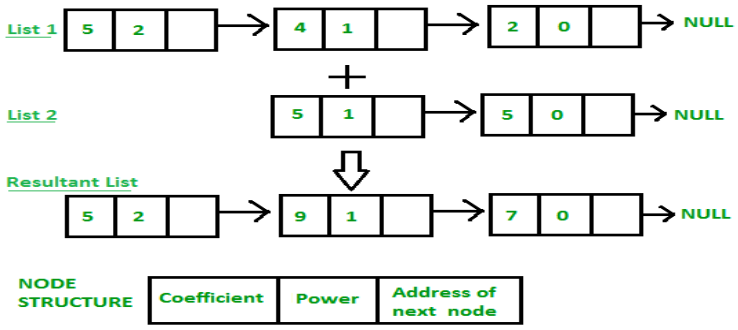
Batch-1: 11-January-2022 @ 9.30 to 12

Batch-2: 13- January -2022 @ 8.40 to 11.10

Task-1

In a class room we have discussed the algorithms to create the polynomial and addition of two polynomials . Based on the same discussion implement a solution.

Input:
1st number = $5x^2 + 4x^1 + 2x^0$
2nd number = $-5x^1 - 5x^0$
Output:
$5x^2-1x^1-3x^0$
Input:
1st number = $5x^3 + 4x^2 + 2x^0$
2nd number = 5x^1 - 5x^0
Output:
$5x^3 + 4x^2 + 5x^1 - 3x^0$



Task-2

A polynomial p(x) is the expression in variable x which is in the form ($ax^n + bx^{n-1} + ... + jx + k$), where a, b, c ...., k fall in the category of real numbers and 'n' is non negative integer, which is called the degree of polynomial. Create a function that takes two sorted linked lists and merges them into a single sorted linked list. Your goal is to return a new linked list that contains the nodes from two lists in sorted order. You may assume the sort order is ascending. For example:

// list1
1 -> 4 -> 10 -> 11

// list2
-1 -> 2 -> 3 -> 6

// merged list
-1 -> 1 -> 2 -> 3 -> 4 -> 6 -> 10 -> 11

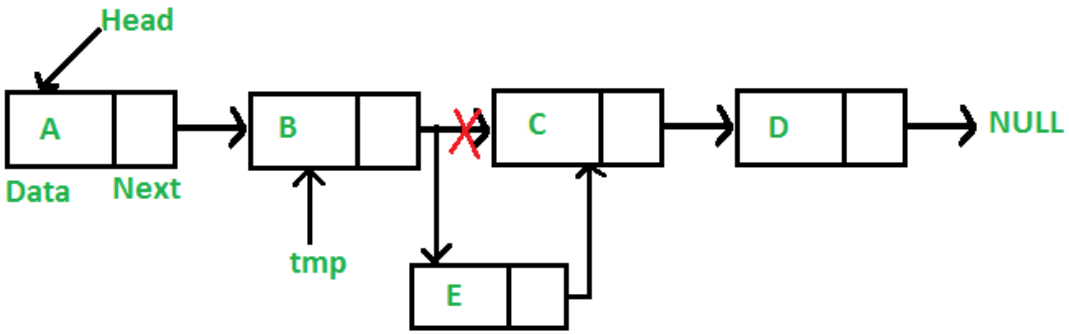
Task-3

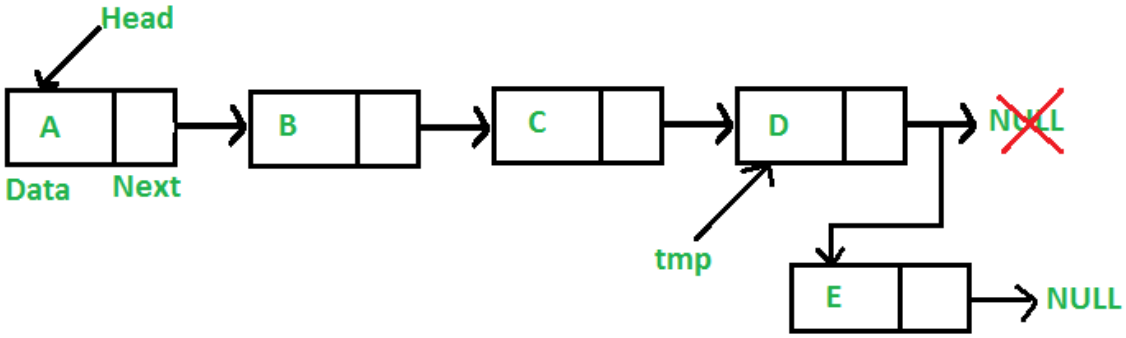
Reverse a linked list: Given pointer to the head node of a linked list, the task is to reverse the linked list. We need to reverse the list by changing the links between nodes.

Input: Head of following linked list
1->2->3->4->NULL
Output: Linked list should be changed to,
4->3->2->1->NULL

Task-4

Perform the following two operation for the case of single linked list





Datastructures Lab Task II/IV B.Tech

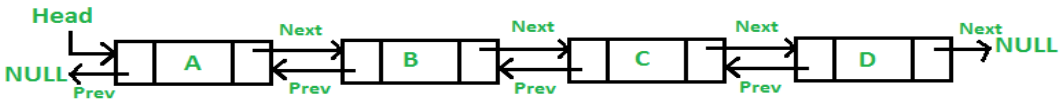
Topic: Types of Linkedlist (Double and Circular)

Batch-1: 25-January-2022 @ 9.30 to 12

Batch-2: 27- January -2022 @ 8.40 to 11.10

Task-1

Design and Implement a program to create double linked list. Use the following pseudocode. Justify your solution with all the possible test cases.



// create node

```
1. Struct node
2. {
3. Struct node *prev;
4. Int data;
5. Struct nodel * next;
6. } *new,*temp,*head;
```

// create double linked list

```
1. do
2. {
3. ….
4. ….
5. ….
6. ….
7. ….
8. ….
9. If(head=NULL)
10. {
11. Head=new;
12. Temp=new;
13. }
14. else
15. {
16. ….
17. ….
18. ….
19. }
20. Printf(“do you need one more term”);
```

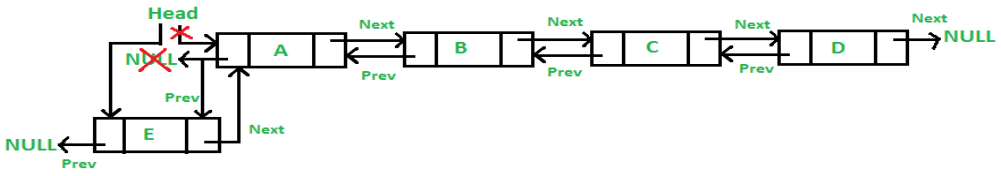
21. Scanf("%c",&ch);
22. }while(ch=='y');

Task-2

Design and Implement a program to insert node in the beginning of the double linked list. Use the following pseudocode. Justify your solution with all the possible test cases. (Use the code designed in task-1 and make extension).

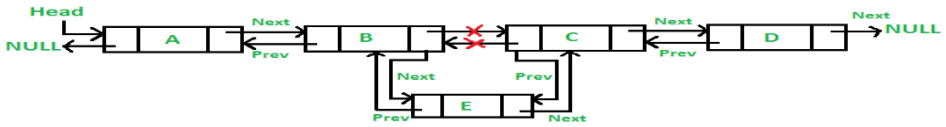
// Insert at begin

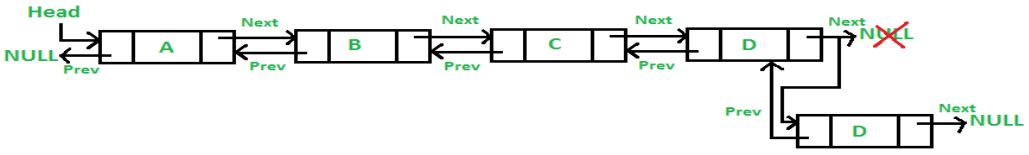
1. new=…….. //allocate memory
2. new-->prev=NULL;
3. …….
4. …….. // read value
5. New→data=value;
6. New→next=head;
7. head=new;



Task-3

Based on the knowledge acquired from Task-1 and Task-2. Perform (i) Insert of node at the end of a double linked list and (ii) Insert of node at the specified position of a double linked list.





Task-4

Design and Implement a program to (i) create circular linked list node (ii) create circular linked list (iii) Insert elements at different positions. Use the knowledge acquired in Task-1,2 and 3. Justify your solution with all the possible test cases.

Datastructures Lab Task II/IV B.Tech

Topic: Applications of LinkedStacks&Queues

Batch-1: 01-February-2022 @ 9.30 to 12

Batch-2: 03- February -2022 @ 8.40 to 11.10

Task-1

In a class room we have discussed the algorithms for different traversal techniques like (i) Inorder (ii) preorder and (iii) postorder. Based on the same discussion implement a solution. Use the following LOC and complete the implimenation part. Also run all the possible testcases.

```
1.  struct node
2.  {
3.  int num;
4.  struct node *left;
5.  struct node *right;
6.  };
7.  void main()
8.  {
9.  struct node *v1,*v2,*v3,*v4,*v5,*v6,*v7;
10. v1=create();
11. v2=create();
12. ----------
13. ----------
14. ----------
15. ----------
16. ----------
17. v1->left=v2;
18. v1->right=v3;
19. v2->left=v4;
20. v2->right=v5;
21. v3->left=v6;
22. v3->right=v7;
23. printf("\nInorder:");
24. inorder(v1);
25. printf("\nPreorder:");
26. preorder(v1);
27. printf("\nPostorder:");
28. postorder(v1);
29. getch();
30. }
31. struct node *create()
32. {
33. struct node *temp;
34. temp=(struct node *)malloc(sizeof(struct node));
35. temp->left=NULL;
36. temp->right=NULL;
37. printf("Enter value:");
38. scanf("%d",&temp->num);
39. return temp;
```

```
40. }
41. void inorder(struct node *T)
42. {
43. if(T!=NULL)
44. {
45. inorder(T->left);
46.
47. inorder(T->right);
48. }
49. }
50. void preorder(struct node *T)
51. {
52.
53.
54.
55.
56.
57. }
58. }
59. void postorder(struct node *T)
60. {
61. if(T!=NULL)
62. {
63. postorder(T->left);
64. postorder(T->right);
65. printf("%d",T->num);
66. }
67. }
```

Task-2

In a class room we have discussed the algorithms for infix to postfix conversion. Based on the same discussion implement a solution. Also run all the possible testcases.

Testcase-1

Infix Expression : 3+4*5/6

Postfix Expression : 3 4 5 * 6 / +

Testcase-2

Infix Expression : A+(B*C-(D/E^F)*G)*H

Postfix Expression : ABC*DEF^/G*-H*+

Datastructures Lab Task II/IV B.Tech

Topic: Implementation of Different types of Trees

Batch-1: 08-February-2022 @ 9.30 to 12

Batch-2: 10- February -2022 @ 8.40 to 11.10

Task-1

Design, develop and execute a program in C/Python to implement Binary Search tree where each node consist of integers. The program should support following functions and show different testcases.

a. Create a Binary Search tree

b. Insert a new node

c. Delete a node if it is found, otherwise display appropriate message

d. Display the nodes of Binary Search tree

e Validate Binary Search tree

Task-2

AVL tree is a descendant of Binary Search Tree but overcomes its drawback of increasing complexity if the elements are sorted. It monitors the balance factor of the tree to be 0 or 1 or -1. In case it tree becomes unbalanced corresponding rotation techniques are performed to balance the tree. Implement insert operation for the creation of nodes in AVL tree. Check the balance factor. If not satisfied apply rotations.

1. Left Rotation: If the addition of a node to the tree's right makes it imbalance then, in that case, Left Rotation needs to be performed.

2. Right Rotation: If the addition of a node to the left of the tree makes the node imbalance then Right Rotation needs to be performed. In other words, When the number of nodes increases on the left side, then there emerges a need to shift the elements to the right side to balance it thus it is said to be Right Rotation.

3. Left-Right Rotation: This type of rotation is a combination of the above 2 rotations explained. This type of rotation occurs when one element is added to the right subtree of a left tree.

In such a case first, perform left rotation on the subtree followed by a right rotation of the left tree.

4. Right-Left Rotation: This type of rotation is also composed of a sequence of above 2 rotations. This type of rotation is needed when an element is added to the left of the right subtree, and the tree becomes imbalanced. In such a case, we first perform right rotation on the right subtree and then left rotation on the right tree.

Justtify the program with different testcases.