Linked List

1. Assume that struct node contains data and next pointer and head points to the beginning of a linked list. What does the following function do:

void foo(struct node\* head)

{

if(head == NULL)

return;

foo(head->next);

printf("%d ", head->data);

}

1. Prints all nodes of linked lists
2. Prints all nodes of linked list in reverse order
3. Prints alternate nodes of Linked List
4. Prints alternate nodes in reverse order

1. Which of the following statements is FALSE regarding Linked List and array data structures.

A It is easy to insert and delete elements in Linked List

B. The size of array has to be pre-decided, linked lists can change their size any time.

C. Accessing any element in the Linked List is faster than that of an array.

D. Linked List can be easily traversed for processing from beginning to end

1. Which of the following statements is FALSE regarding Linked List and array data structures.

A. Arrays have better cache locality (consecutive memory locations) that can make them better in terms of performance.

B It is easy to insert and delete elements in an array.

C Random access is faster in array compared to a linked list.

D Arrays can be easily traversed for processing from beginning to end

1. The following function takes reference to head of a doubly linked list as parameter and reverses the dll. Assume that a dnode of doubly linked list has previous pointer as prev and next pointer as next. Fill in the missing four statements in the while loop to complete the function.

/\* Link list dnode \*/

struct dnode

{

int data;

struct dnode\* prev;

struct dnode\* next;

};

void Reversedll (struct dnode \*\*head\_ref)

{

struct dnode \*temp = NULL;

struct dnode \*current = \*head\_ref;

while (current != NULL)

{

temp = current->prev;

current->prev = current->next;

current->next = temp;

current = current->prev;

}

if(temp != NULL )

\*head\_ref = temp->prev;

}

1. The following C function takes a simply-linked list as input argument. It modifies the list by moving the last element to the front of the list and returns the modified list. Some part of the code is left blank. Choose the correct alternative to replace the blank line.

typedef struct node

{

int value;

struct node \*next;

}Node;

Node \*move\_to\_front(Node \*head)

{

Node \*p, \*q;

if ((head == NULL: || (head->next == NULL))

return head;

q = NULL; p = head;

while (p-> next !=NULL)

{

q = p;

p = p->next;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

return head;

}

1. q = NULL; p->next = head; head = p;

B q->next = NULL; head = p; p->next = head;

1. head = p; p->next = q; q->next = NULL;

D q->next = NULL; p->next = head; head = p;

1. What is the output of following function for head pointing to first node of following linked list? 1->2->3->4->5->6

void fun1(struct node\* head)

{

if(head == NULL)

return;

printf("%d ", head->data);

if(head->next != NULL )

fun1(head->next->next);

printf("%d ", head->data);

}

A 1 4 6 6 4 1

B 1 3 5 1 3 5

C 1 2 3 5

D 1 3 5 5 3 1

1. Assume that X is pointing to a node in singly linked list. What does the following piece of code accomplish:

struct node \*temp = X->next;

X->data = temp->data;

X->next = temp->next;

free(temp);

1. Equivalent to deleting the node X
2. Equivalent to deleting the node following X
3. Consider the following function to traverse a linked list.

void traverse(struct Node \*head)

{

while (head->next != NULL)

{

printf("%d ", head->data);

head = head->next;

}

}

Which of the following is TRUE about above function?

1. The function may crash when the linked list is empty
2. The function doesn't print the last node when the linked list is not empty
3. You are given pointers to first and last nodes of a singly linked list, which of the following operations are dependent on the length of the linked list?
4. Delete the first element
5. Insert a new element as a first element
6. Delete the last element of the list
7. Add a new element at the end of the list
8. Which of the following is true about linked list implementation of stack?

A In push operation, if new nodes are inserted at the beginning of linked list, then in pop operation, nodes must be removed from end.

B In push operation, if new nodes are inserted at the end, then in pop operation, nodes must be removed from the beginning.

C Both of the above

D None of the above

1. Consider the following pseudocode that uses a stack

declare a stack of characters

while ( there are more characters in the word to read )

{

read a character

push the character on the stack

}

while ( the stack is not empty )

{

pop a character off the stack

write the character to the screen

}

What is the output for this user input - terminal?

1. Following is an incorrect pseudocode for the algorithm which is supposed to determine whether a sequence of parentheses is balanced:

declare a character stack

while ( more input is available)

{

read a character

if ( the character is a '(' )

push it on the stack

else if ( the character is a ')' and the stack is not empty )

pop a character off the stack

else

print "unbalanced" and exit

}

print "balanced"

Give an example of an unbalanced expression, which the above algorithm will print as balanced?

1. A single array A[1..MAXSIZE] is used to implement two stacks. The two stacks grow from opposite ends of the array. Variables top1 and top2 (topl< top 2) point to the location of the topmost element in each of the stacks. If the space is to be used efficiently, the condition for “stack full” is (GATE CS 2004)

A.(top1 = MAXSIZE/2) and (top2 = MAXSIZE/2+1)

B. top1 + top2 = MAXSIZE

1. (top1= MAXSIZE/2) or (top2 = MAXSIZE)
2. Top1 = top2 – 1
3. Algorithm to evaluate postfix expression using a Stack

1. While there are input tokens left

o Read the next token from input.

o If the token is a value

+ Push it onto the stack.

o Otherwise, the token is an operator

(operator here includes both operators, and functions).

\* It is known a priori that the operator takes n arguments.

\* If there are fewer than n values on the stack

(Error) The user has not input sufficient values in the expression.

\* Else, Pop the top n values from the stack.

\* Evaluate the operator, with the values as arguments.

\* Push the returned results, if any, back onto the stack.

2. If there is only one value in the stack

o That value is the result of the calculation.

3. If there are more values in the stack

o (Error) The user input has too many values.

1. The following postfix expression with single digit operands is evaluated using a stack:

8 2 3 ^ / 2 3 \* + 5 1 \* -

Note that ^ is the exponentiation operator. The top two elements of the stack after the first \* is evaluated are:

6, 1

1. Following is C like pseudo code of a function that takes a number as an argument, and uses a stack S to do processing. Binary Represetation of a number

void fun(int n)

{

Stack S; // Say it creates an empty stack S

while (n > 0)

{

// This line pushes the value of n%2 to stack S

push(&S, n%2);

n = n/2;

}

// Run while Stack S is not empty

while (!isEmpty(&S))

printf("%d ", pop(&S)); // pop an element from S and print it

}

1. The result evaluating the postfix expression 10 5 + 60 6 / \* 8 – is 142
2. Predict output of following program

#include <stdio.h>

int fun(int n)

{

if (n == 4)

return n;

else return 2\*fun(n+1);

}

int main()

{

printf("%d ", fun(2));

return 0;

}

4,8,16,

1. Consider the following recursive function fun(x, y). What is the value of fun(4, 3)

int fun(int x, int y)

{

if (x == 0)

return y;

return fun(x - 1, x + y);

}

9, 10, 12, 13

20. Level of a node is distance from root to that node. For example, level of root is 1 and levels of left and right children of root is 2. The maximum number of nodes on level i of a binary tree is   
  
In the following answers, the operator '^' indicates power.

(a) 2^(i-1) (b) 2^1 (c) 2^(i+1) (d) 2^[(i+1)/2]

21. Which of the following is a true about Binary Trees

A. Every binary tree is either complete or full.

B Every complete binary tree is also a full binary tree.

C Every full binary tree is also a complete binary tree.

D No binary tree is both complete and full.

E None of the Above.is correct.

22. The maximum number of binary trees that can be formed with three unlabeled nodes is: 5

23. The number of leaf nodes in a rooted tree of n nodes (full binary tree), with each node having 0 or 2 children is:

(a) (n+1)/2 (b) n/2 (c) n-1/2 (d) log n

24. The height of a binary tree is the maximum number of edges in any root to leaf path. The maximum number of nodes in a binary tree of height h is:

(a) 2^h-1 (b) 2^(h-1) -1 (c) 2^(h+1) -1 (d) 2^(h+1)

25. A scheme for storing binary trees in an array X is as follows. Indexing of X starts at 1 instead of 0. the root is stored at X[1]. For a node stored at X[i], the left child, if any, is stored in X[2i] and the right child, if any, in X[2i+1]. The parent of a node stored at location j is given by:

26. Postorder traversal of a given binary search tree, T produces the following sequence of keys 10, 9, 23, 22, 27, 25, 15, 50, 95, 60, 40, 29 Which one of the following sequences of keys can be the result of an in-order traversal of the tree T?

(a) 9, 10, 15, 22, 23, 25, 27, 29, 40, 50, 60, 95

(b) 9, 10, 15, 22, 40, 50, 60, 95, 23, 25, 27, 29

(c) 29, 15, 9, 10, 25, 22, 23, 27, 40, 60, 50, 95

(d) 95, 50, 60, 40, 27, 23, 22, 25, 10, 9, 15, 29

27. Consider a node X in a Binary Tree. Given that X has two children, let Y be Inorder successor of X. Which of the following is true about Y?

(a) Y has no left child (b) Y has no right child (d) Y has both children (d) None of the above.

|  |
| --- |
|  |

28. The height of a tree is the length of the longest root-to-leaf path in it. The maximum and minimum number of nodes in a binary tree of height 5 are:

(a) 63 and 6 (b) 64 and 5 (c) 32 and 6 (d) 31 and 5

29. A binary tree T has 20 leaves. The number of nodes in T having two children is:

(a) 17 (b) 18 (c) 19 (d) any number between 10 and 20

30. Following function is supposed to calculate the maximum depth or height of a Binary tree -- the number of edges along the longest path from the root node down to the farthest leaf node.

|  |
| --- |
| int maxDepth(struct node\* node)  {     if (node==NULL)         return -1;     else     {         /\* compute the depth of each subtree \*/         int lDepth = maxDepth(node->left);         int rDepth = maxDepth(node->right);           /\* use the larger one \*/         if (lDepth > rDepth)             return X;         else return Y;     }  } |

(a) X = lDepth; Y = rDepth (b) X = lDepth+1; Y = rDepth+1 (c) X = lDepth-1; Y = rDepth- 1(d) X=0 Y=1

31.Match the following regarding the binary tree traverals:

Inorder :

Postorder :

Preorder :

32. What is common in three different types of traversals (Inorder, Preorder and Postorder)?

(a). Root is visited before right subtree

(b). Left subtree is always visited before right subtree

(c) Root is visited after left subtree

(d) None of the above

33. What is the maximum height of any AVL-tree with 7 nodes? Assume that the height of a tree with a single node is 0. => 3

34. Which statements is FALSE regarding a AVL tree:

(a). AVL tree is a binary search tree

(b). The height difference between the left and right subtree is at most 1

(c) Insertions and Deletions can occur at any node

(d) Insertion or Deletion will always require one or more rotations.

35. In delete operation of BST, we need inorder successor (or predecessor) of a node when the node to be deleted has both left and right child as non-empty. Which of the following is true about inorder successor needed in delete operation?

(a). Inorder Successor is always a leaf node.

(b). Inorder Successor is always a leaf node or a node with empty left child.

© norder Successor is always a leaf node or a node with empty right child.

(d) Inorder successor may be an ancestor of that node.

36. Which of the following traversal outputs the data in sorted order in a BST?

(a). Preorder (b) Postorder (c) Inorder (d) None

37. Suppose the numbers 7, 5, 1, 8, 3, 6, 0, 9, 4, 2 are inserted in that order into an initially empty binary search tree. The binary search tree uses the usual ordering on natural numbers. What is the in-order traversal sequence of the resultant tree?

(a). 7 5 1 0 3 2 4 6 8 9

(b) 0 2 4 3 1 6 5 9 8 7

© 0 1 2 3 4 5 6 7 8 9

(d) 9 8 6 4 2 3 0 1 5 7

38. The following numbers are inserted into an empty binary search tree in the given order: 10, 1, 3, 5, 15, 12, 16. What is the height of the binary search tree (the height is the maximum distance of a leaf node from the root)?

(a). 2 (b). 3 (c) 4 (d) 6

39. While inserting the elements 71, 65, 84, 69, 67, 83 in an empty binary search tree (BST) in the sequence shown, the element in the lowest level is:

(a). 65 (b) 67 (c) 69 (d) 83

40. Suppose that we have numbers between 1 and 100 in a binary search tree and want to search for the number 55. Which of the following sequences CANNOT be the sequence of nodes examined?

(a). { 10, 75, 64, 43, 60, 57, 55}

(b). {90, 12, 68, 34, 62, 45, 55}

© {9, 85, 47, 68, 43, 57, 55}

(d) {79, 14, 72, 56, 16, 53, 55}

41. How many distinct binary search trees can be constructed out of 3 distinct keys?

4,5,6,7,

42. A scheme for storing binary trees in an array X is as follows. Indexing of X starts at 1 instead of 0. the root is stored at X[1]. For a node stored at X[i], the left child, if any, is stored in X[2i] and the right child, if any, in X[2i+1]. To be able to store any binary tree on n vertices the minimum size of X should be.

(a) n+1 (b) log2n (c) 2^n – 1 (d) 2n + 1 (It is 2^n -1 due to skewed tree)

43. Consider a node X in a Binary Tree. Given that X has two children, let Y be Inorder successor of X. Which of the following is true about Y?

(a). Y has no right child (b) Y has no left child (c) Y has both children (d) None of the above

44. What is the balance factor of the root node in this tree?.

100

/ \

50 200

/ / \

10 150 300

/

5

45. The following is an AVL Tree?.

100

/ \

50 200

/ \ / \

10 60 150 300

/ \ \

5 180 400

46. After inserting 70, in this AVL tree and re-balancing the tree, the new root of the AVL tree is:

60

/ \

20 100

/ \

80 120

60, 100, 70, 80

47. Which of the following is NOT correct statement about an AVL tree:

(a). AVL tree is also a Binary Search tree

(b). AVL tree has all nodes that a balance factor of -1, 0 or +1

(c). All leaves of AVL tree are at the same level (or depth from the root).

(d). Rotations can be applied to convert an imbalanced tree to a AVL tree.

48. A max-heap is a heap where the value of each parent is greater than or equal to the values of its children. Which of the following is a max-heap implemented using an array?.

(a) 25,12,16,13,10,8,14

(b) 25,14,16,13,10,8,12

(c) 15,14,16,13,10,8,12

(d) 25,14,12,13,10,8,16

49. Consider any array representation of an n element binary heap where the elements are stored from index 1 to index n of the array. For the element stored at index i of the array (i <= n), the index of the parent is:

(a) i – 1 (b) floor (i/2) (c) ceiling(i/2) (d) (i+1)/2

50. Consider a max heap, represented by the array: 40, 30, 20, 10, 15, 16, 17, 8, 4. Now consider that a value 35 is inserted into this heap. After insertion, the new heap is :

(a). 40,30,20,10,15,16,17,8,4,35

(b). 40,30,20,10,35,16,17,8,4,15

(c) 40,30,20,10,35,16,17,8,4,15

(d) 40,35,20,10,15,16,17,8,4,30

51. Which of the following sequences of array elements forms a heap?

(a). 23, 17, 14, 6, 13, 10, 1, 12,7,5

(b) 23, 17, 14, 6, 13, 10, 1, 5, 7,12

(c) 23,17,14,7,13,10,1,5,6,12 OK

(d) 23, 17, 14, 7, 13,10, 1,12, 5, 7