योगः कर्मसु कौशलम्

PARUL UNIVERSITY

Parul® University

303105201 - Design of Data Structures

Academic Year - 2024-2025 – Sem-3

ASSIGNEMNT-1

(Based on Unit-1 & 2)

Submission Date – Latest by 20-July-2024

Long Questions.

SR.No	Question
1	Classify data structures with diagram.
2	Interpret Big O complexity chart.
3	Discuss Time complexity
4.	Describe sparse matrix. Find the address of A [2][1] if base address is 1024 for an integer array A[5][4] in row major order and word size is 2 byte.
5.	Given a two dimensional array A1(1:8, 7:14) stored in row-major order with base address 100 and size of each element is 4 bytes, find address of the element A1(4, 12).
6.	Convert infix to Postfix and Prefix 1. (A + B) / C – D * E 2. P ^ Q ^ R + S / T 3. A * B – (C / D + (E – F)) ^ G
7.	Describe Circular Queue algorithm for inserting and deleting an element with diagram. Why circular queue is better than linear queue?
8.	Discuss Push and Pop algorithm of stack.
9.	List applications of stack and Convert 2 * 3 / (2-1) + 5 * 3 infix expression into postfix format. Showing stacks status after every step in tabular form and evaluates that postfix notation
10.	Transform the following expression to postfix and evaluate postfix expression by assuming A=1,B=2,C=3,D=4,E=6,F=6,G=1,I=3 and J=3 A + B – C * D / E + F * G / (I + J)
11.	Transform the following expression to postfix and evaluate postfix expression (by assuming P=1, Q=2, R=3, S=5, T=5, U=6, V=4 and W=3) P + Q – R * S / T + U * V / W
12.	The initial configuration of the queue is having the elements ‘x, y, z, a’ forming a queue with size 4 to get the new configuration ‘a, z, y, x’ one needs a minimum of a) 3 additions and 4 deletion b) 3 deletions and 4 additions c) 3 deletions and 3 additions d) 4 deletions and 4 additions.

Short Questions.

SR.No	Question
1.	Define dynamic memory allocation?
2.	Define referential structure?
3.	Array is a heterogeneous data type. (True/False). Justify your answer.
4.	A m*n matrix which contains very few non-zero elements. A matrix contains more number of ZERO values than NON-ZERO values. Such matrix is known as ?
5.	Insert an element in array at index k will take how much time? (a) O(1) (b) O(n-k) (c) O(log(n-k)) (d) O(n2-k)
6.	State True/False: Arrays have better cache locality that can make them better in terms of performance.
7.	Differentiate between LIFO and FIFO access mechanism.
8.	How linked list is better compared to stack, queue and array? Explain with concept of dynamic memory allocation.
9.	In which type of scenario, linear queue (simple queue) is better than circular queue?
10.	After evaluation of 3,5,4,*,+ result is ?
11.	What will be the value of Front and Rear pointers when Queue is empty?

MCQ Questions (Unit-1)

1. A program P reads in 500 integers in the range [0..100] representing the scores of 500 students. It then prints the frequency of each score above 50. What would be the best way for P to store the frequencies?

   (A) An array of 50 numbers (B) An array of 100 numbers

   (C) An array of 500 numbers (D) A dynamically allocated array of 550 numbers

2. Consider the following C program.

```
#include < stdio.h >
int main () {
   int a [4] [5] = {{1, 2, 3, 4, 5},
                    {6, 7, 8, 9, 10},
                    {11, 12, 13, 14, 15},
                    {16, 17, 18, 19, 20}};
   printf (“%d\n”, *(*(a+**a+2) +3));
   return (0);
}
```

   The output of the program is ________.

3. If array A is made to hold the string “abcde”, which of the below four test cases will be

   successful in exposing the flaw in this procedure?

   (A) None (B) 2 only (C) 3 and 4 only (D) 4 only

4. Consider the following pseudo-code:

```
function Example(n: integer)
   sum := 0
   for i := 1 to n do
      for j := 1 to i do
         sum := sum + 1
   return sum
```

What is the time complexity of the above pseudo-code?

A) O(n)

B) O(n log n)

C) O(n^2)

D) O(n^3)

MCQ QUESTIONS: - (UNIT-2)

1. Which one of the following is an application of Stack Data Structure?

   (A) Managing function calls (B) The stock span problem

   (C) Arithmetic expression evaluation (D) All of the above

2. Consider the following sequence of operations on an initially empty stack:

   1. push(10)
   2. push(20)
   3. push(30)
   4. pop()
   5. push(40)
   6. pop()
   7. pop()
   8. pop()

   What is the sequence of values popped from the stack?
   A) 30, 40, 20, 10
   B) 40, 30, 20, 10
   C) 30, 40, 10, 20
   D) 40, 30, 10, 20

3. Consider a stack implemented using a dynamic array that doubles its size whenever the stack is full. Suppose we start with an empty stack with an initial capacity of 2. The following operations are performed on the stack:

   1. push(10)
   2. push(20)
   3. push(30)
   4. push(40)
   5. pop()
   6. pop()

7. push(50)
8. push(60)
9. push(70)

What is the final capacity of the dynamic array after these operations?

A) 4
B) 8
C) 16
D) 32

4. What is the time complexity of an infix to postfix conversion algorithm?

   (A) O(N log N) (B) O(N) (C) O(N^2) (D) O(N log N)

5. The recurrence relation capturing the optimal execution time of the Towers of Hanoi problem with n discs is

   (A) T(n) = 2T (n- 2) + 2 (B) T(n) = 2T(n-1) + n

   (C) T (n)=2T(n/2) + 1 (D) T (n) =2T(n - 1) + 1

6. Which one of the following is an application of Stack Data Structure?

   (A) Managing function calls (B) The stock span problem

   (C) Arithmetic expression evaluation (D) All of the above

7. Consider a linear queue implemented using an array `Q` of size `N`. The queue uses two variables `front` and `rear` to keep track of the front and rear positions of the queue respectively. Initially, the queue is empty with `front = -1` and `rear = -1`.

   Which of the following conditions correctly checks if the queue is full?
   A) `rear == N - 1`
   B) `front == N - 1`
   C) `(rear + 1) % N == front`
   D) `rear == front`

8. In circular queue, to manage rear pointer, which formula is used? Where rear is a pointer to insert an element and N is size of queue.

   (A) rear=rear-1 (B) rear=(rear+1) % N

   (C) rear=rear+1 (D) rear=(rear+1)/N

9. What is the main advantage of using a priority queue?

   A. Easy to implement B. Fast access to the largest (or smallest) element

   C. Saves memory D. Simplifies the code