DS : Queues Quiz

Data Structures Quiz on Queues and Linked List.

Question Prompt: 1
Total Points: 1

Which one of the following is an application of Queue Data Structure?

- ☐ When a resource is shared among multiple consumers
- ☐ When data is transferred asynchronously (data not necessarily received at same rate as sent) between two processes
- ☐ Load Balancing
- ☑ All of the above
- ☐ None

Question Prompt: 2
Total Points: 2

How many queues are needed to implement a stack. Consider the situation where no other data structure like arrays, linked list is available to you

- ☐ 1
- ☑ 2
- ☐ 3
- ☐ 4

Question Prompt: 3
Total Points: 2

Suppose a circular queue of capacity (n – 1) elements is implemented with an array of n elements. Assume that the insertion and deletion operation are carried out using REAR and FRONT as array index variables, respectively. Initially, REAR = FRONT = 0. The conditions to detect queue full and queue empty are

- ☑ Full: (REAR+1) mod n == FRONT, empty: REAR == FRONT
- ☐ Full: (REAR+1) mod n == FRONT, empty: (FRONT+1) mod n == REAR
- ☐ Full: REAR == FRONT, empty: (REAR+1) mod n == FRONT
- ☐ Full: (FRONT+1) mod n == REAR, empty: REAR == FRONT

Question Prompt: 4
Total Points: 1

If the elements “D”, “C”, “B” and “A” are placed in a queue and are deleted one at a time, in what order will they be removed?

- ☐ ABCD
- ☐ DCAB
- ☑ DCBA
- ☐ DABC

Question Prompt: 5
Total Points: 2

A normal queue, if implemented using an array of size MAX_SIZE, gets full when

- ☐ Front = (rear + 1)mod MAX_SIZE
- ☐ Front = rear + 1
- ☑ Rear = MAX_SIZE – 1

- [ ] Rear = front

Question Prompt: 6
Total Points: 1

Which of the following is not the type of queue?

- [ ] Ordinary queue
- [x] Single ended queue
- [ ] Circular queue
- [ ] Priority queue

Question Prompt: 7
Total Points: 1

What is the complexity of searching for a particular element in a Singly Linked List?

- [x] O(n)
- [ ] O(1)
- [ ] logn
- [ ] nlogn

Question Prompt: 8
Total Points: 1

Which of the following data structures can be used for parentheses matching?

- [ ] n-ary tree
- [ ] queue
- [ ] priority queue
- [x] stack

Question Prompt: 9
Total Points: 2

What does the following function do for a given Linked List with first node as head? void fun1(struct node* head) { if(head == NULL) return; fun1(head->next); printf("%d ", head->data); }

- [ ] Prints all nodes of linked lists
- [x] Prints all nodes of linked list in reverse order
- [ ] Prints alternate nodes of Linked List
- [ ] Prints alternate nodes in reverse order

Question Prompt: 10
Total Points: 1

Which of the following points is/are true about Linked List data structure when it is compared with array?

- [ ] Arrays have better cache locality that can make them better in terms of performance
- [ ] It is easy to insert and delete elements in Linked List
- [ ] Random access is not allowed in a typical implementation of Linked Lists
- [ ] The size of array has to be pre-decided, linked lists can change their size any time
- [x] All the above

Question Prompt: 11
Total Points: 3

The following function reverse() is supposed to reverse a singly linked list. There is one line missing at the end of the function. /* Link list node */ struct node { int data; struct node* next; }; /* head_ref is a double pointer which points to head (or start) pointer of linked list */ static void reverse(struct node head_ref) { struct node* prev = NULL; struct node* current = *head_ref; struct node* next; while (current != NULL) { next = current->next; current->next = prev; prev = current; current = next; } /*ADD A STATEMENT HERE*/ }**

- [✓ *head_ref = prev;]
- *head_ref = current;
- *head_ref = next;
- *head_ref = NULL;

Question Prompt: 12
Total Points: 3

What is the output of following function for start pointing to first node of following linked list? 1->2->3->4->5->6 void fun(struct node* start) { if(start == NULL) return; printf("%d ", start->data); if(start->next != NULL) fun(start->next->next); printf("%d ", start->data); }

- 1 4 6 6 4 1
- 1 3 5 1 3 5
- 1 2 3 5
- [✓ 1 3 5 5 3 1]
