Rajalakshmi Engineering College

Name: Aishwarya R
Email: 241501011@rajalakshmi.edu.in
Roll no: 241501011
Phone: null
Branch: REC
Department: I AI & ML FA
Batch: 2028
Degree: B.E - AI & ML

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 4_MCQ_Updated

Attempt : 1
Total Mark : 20
Marks Obtained : 19

Section 1 : MCQ

1. What will be the output of the following code?

```c
#include <stdio.h>
#include <stdlib.h>
#define MAX_SIZE 5
typedef struct {
    int* arr;
    int front;
    int rear;
    int size;
} Queue;
Queue* createQueue() {
    Queue* queue = (Queue*)malloc(sizeof(Queue));
    queue->arr = (int*)malloc(MAX_SIZE * sizeof(int));
    queue->front = -1;
    queue->rear = -1;
```

```
    queue->size = 0;
    return queue;
}
int isEmpty(Queue* queue) {
    return (queue->size == 0);
}
int main() {
    Queue* queue = createQueue();
    printf("Is the queue empty? %d", isEmpty(queue));
    return 0;
}
```

Answer

Is the queue empty? 1

Status : Correct   *Marks : 1/1*

2.   What is the functionality of the following piece of code?

```
public void function(Object item)
{
    Node temp=new Node(item,trail);
    if(isEmpty())
    {
    head.setNext(temp);
    temp.setNext(trail);
    }
    else
    {
      Node cur=head.getNext();
      while(cur.getNext()!=trail)
      {
        cur=cur.getNext();
      }
      cur.setNext(temp);
    }
    size++;
}
```

Answer

Insert at the rear end of the dequeue

Status : Correct   ***Marks : 1/1***

3. Which of the following can be used to delete an element from the front end of the queue?

Answer

public Object deleteFront() throws emptyDEQException{if(isEmpty())throw new emptyDEQException("Empty");else{Node temp = head.getNext();Node cur = temp.getNext();Object e = temp.getEle();head.setNext(cur);size--;return e;}}

Status : Correct   ***Marks : 1/1***

4. When new data has to be inserted into a stack or queue, but there is no available space. This is known as

Answer

overflow

Status : Correct   ***Marks : 1/1***

5. Which operations are performed when deleting an element from an array-based queue?

Answer

Dequeue

Status : Correct   ***Marks : 1/1***

6. Front and rear pointers are tracked in the linked list implementation of a queue. Which of these pointers will change during an insertion into the EMPTY queue?

Answer

Both front and rear pointer

7. Insertion and deletion operation in the queue is known as

Answer

Enqueue and Dequeue

Status : Correct   *Marks : 1/1*

8. What does the front pointer in a linked list implementation of a queue contain?

Answer

The address of the first element

Status : Correct   *Marks : 1/1*

9. Which of the following properties is associated with a queue?

Answer

First In First Out

Status : Correct   *Marks : 1/1*

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

Answer

All of the mentioned options

Status : Correct   *Marks : 1/1*

11. In what order will they be removed If the elements "A", "B", "C" and "D" are placed in a queue and are deleted one at a time

Answer

ABCD

12.   The essential condition that is checked before insertion in a queue is?

Answer

Overflow

Status : Correct   *Marks : 1/1*

13.   In linked list implementation of a queue, the important condition for a queue to be empty is?

Answer

FRONT is null

Status : Correct   *Marks : 1/1*

14.   The process of accessing data stored in a serial access memory is similar to manipulating data on a

Answer

Queue

Status : Correct   *Marks : 1/1*

15.   In a linked list implementation of a queue, front and rear pointers are tracked. Which of these pointers will change during an insertion into a non-empty queue?

Answer

Only rear pointer

Status : Correct   *Marks : 1/1*

16.   What are the applications of dequeue?

Answer

All the mentioned options

Status : Correct   *Marks : 1/1*

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

Answer

Rear = MAX_SIZE − 1

Status : Correct   *Marks : 1/1*

18.   What will the output of the following code?

```
#include <stdio.h>
#include <stdlib.h>
typedef struct {
    int* arr;
    int front;
    int rear;
    int size;
} Queue;
Queue* createQueue() {
    Queue* queue = (Queue*)malloc(sizeof(Queue));
    queue->arr = (int*)malloc(5 * sizeof(int));
    queue->front = 0;
    queue->rear = -1;
    queue->size = 0;
    return queue;
}
int main() {
    Queue* queue = createQueue();
    printf("%d", queue->size);
    return 0;
}
```

Answer

0

19.   What will be the output of the following code?

```c
#include <stdio.h>
#define MAX_SIZE 5
typedef struct {
    int arr[MAX_SIZE];
    int front;
    int rear;
    int size;
} Queue;

void enqueue(Queue* queue, int data) {
    if (queue->size == MAX_SIZE) {
        return;
    }
    queue->rear = (queue->rear + 1) % MAX_SIZE;
    queue->arr[queue->rear] = data;
    queue->size++;
}
int dequeue(Queue* queue) {
    if (queue->size == 0) {
        return -1;
    }
    int data = queue->arr[queue->front];
    queue->front = (queue->front + 1) % MAX_SIZE;
    queue->size--;
    return data;
}
int main() {
    Queue queue;
    queue.front = 0;
    queue.rear = -1;
    queue.size = 0;
    enqueue(&queue, 1);
    enqueue(&queue, 2);
```

```
    enqueue(&queue, 3);
    printf("%d ", dequeue(&queue));
    printf("%d ", dequeue(&queue));
    enqueue(&queue, 4);
    enqueue(&queue, 5);
    printf("%d ", dequeue(&queue));
    printf("%d ", dequeue(&queue));
    return 0;
}
```

Answer

1 2 3 5

Status : Wrong   *Marks : 0/1*

20.   After performing this set of operations, what does the final list look to contain?

```
InsertFront(10);
InsertFront(20);
InsertRear(30);
DeleteFront();
InsertRear(40);
InsertRear(10);
DeleteRear();
InsertRear(15);
display();
```

Answer

10 30 40 15

Status : Correct   *Marks : 1/1*