Rajalakshmi Engineering College

Name: Padma Priya D
Email: 240701377@rajalakshmi.edu.in
Roll no: 240701377
Phone: 8668123104
Branch: REC
Department: I CSE FD
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 4_CY

Attempt : 1
Total Mark : 30
Marks Obtained : 30

Section 1 : Coding

1. Problem Statement

Fathima has been tasked with developing a program to manage a queue of customers waiting in line at a service center. Help her write a program simulating a queue data structure using a linked list.

Here is a description of the scenario and the required operations:

Enqueue: Add a customer to the end of the queue.Dequeue: Remove and discard a customer from the front of the queue.Display waiting customers: Display the front and rear customer IDs in the queue.

Write a program that enqueues all the customers into the queue, performs a dequeue operation, and prints the front and rear elements.

Input Format

The first input line consists of an integer N, representing the number of customers to be inserted into the queue.

The second line consists of N space-separated integers, representing the customer IDs.

Output Format

The output prints "Front: X, Rear: Y" where X is the front element and Y is the rear element, after performing the dequeue operation.

Refer to the sample output for the exact text and format.

Sample Test Case

Input: 5
112 104 107 116 109

Output: Front: 104, Rear: 109

Answer

```
// You are using GCC
#include <stdio.h>
#include <stdlib.h>
typedef struct Node {
    int data;
    struct Node* next;
} Node;
typedef struct Queue {
    Node* front;
    Node* rear;
} Queue;
void initQueue(Queue* q) {
    q->front = NULL;
    q->rear = NULL;
}
void enqueue(Queue* q, int value) {
    Node* newNode = (Node*)malloc(sizeof(Node));
    newNode->data = value;
    newNode->next = NULL;

    if (q->rear == NULL) {
```

```
        q->front = newNode;
        q->rear = newNode;
    } else {
        q->rear->next = newNode;
        q->rear = newNode;
    }
}
void dequeue(Queue* q) {
    if (q->front == NULL) return;

    Node* temp = q->front;
    q->front = q->front->next;
    if (q->front == NULL) {
        q->rear = NULL;
    }
    free(temp);
}

int main() {
    int N;
    scanf("%d", &N);

    Queue q;
    initQueue(&q);

    for (int i = 0; i < N; i++) {
        int customerID;
        scanf("%d", &customerID);
        enqueue(&q, customerID);
    }

    dequeue(&q);
    if (q.front != NULL && q.rear != NULL) {
        printf("Front: %d, Rear: %d\n", q.front->data, q.rear->data);
    } else {
        printf("Queue is empty\n");
    }

    while (q.front != NULL) {
        dequeue(&q);
    }
```

2407013777

```
    return 0;
}
```

Status : Correct ***Marks : 10/10***

2. Problem Statement

A customer support system is designed to handle incoming requests using a queue. Implement a linked list-based queue where each request is represented by an integer. After processing the requests, remove any duplicate requests to ensure that each request is unique and print the remaining requests.

Input Format

The first line of input consists of an integer N, representing the number of requests to be enqueued.

The second line consists of N space-separated integers, each representing a request.

Output Format

The output prints space-separated integers after removing the duplicate requests.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
2 4 2 7 5
Output: 2 4 7 5

Answer

```
// You are using GCC
#include <stdio.h>
#include <stdlib.h>
typedef struct Node {
    int data;
```

```
    struct Node* next;
} Node;
void enqueue(Node** front, Node** rear, int value) {
    Node* newNode = (Node*)malloc(sizeof(Node));
    newNode->data = value;
    newNode->next = NULL;
    if (*rear == NULL) {
        *front = newNode;
        *rear = newNode;
    } else {
        (*rear)->next = newNode;
        *rear = newNode;
    }
}
int exists(Node* head, int value) {
    Node* temp = head;
    while (temp != NULL) {
        if (temp->data == value) return 1;
        temp = temp->next;
    }
    return 0;
}
void printUnique(Node* front) {
    Node* uniqueHead = NULL;
    Node* uniqueRear = NULL;
    Node* temp = front;

    while (temp != NULL) {
        if (!exists(uniqueHead, temp->data)) {
            enqueue(&uniqueHead, &uniqueRear, temp->data);
        }
        temp = temp->next;
    }
    temp = uniqueHead;
    while (temp != NULL) {
        printf("%d ", temp->data);
        temp = temp->next;
    }
    printf("\n");
    while (uniqueHead != NULL) {
        Node* t = uniqueHead;
        uniqueHead = uniqueHead->next;
```

2407013777

```
        free(t);
    }
}

int main() {
    int N;
    scanf("%d", &N);

    Node* front = NULL;
    Node* rear = NULL;

    for (int i = 0; i < N; i++) {
        int req;
        scanf("%d", &req);
        enqueue(&front, &rear, req);
    }

    printUnique(front);
    while (front != NULL) {
        Node* t = front;
        front = front->next;
        free(t);
    }

    return 0;
}
```

Status : Correct *Marks : 10/10*

3. Problem Statement

Pathirana is a medical lab specialist who is responsible for managing blood count data for a group of patients. The lab uses a queue-based system to track the blood cell count of each patient. The queue structure helps in processing the data in a first-in-first-out (FIFO) manner.

However, Pathirana needs to remove the blood cell count that is positive even numbers from the queue using array implementation of queue, as they are not relevant to the specific analysis he is performing. The remaining data will then be used for further medical evaluations and

reporting.

Input Format

The first line consists of an integer n, representing the number of a patient's blood cell count.

The second line consists of n space-separated integers, representing a blood cell count value.

Output Format

The output displays space-separated integers, representing the remaining blood cell count after removing the positive even numbers.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
1 2 3 4 5

Output: 1 3 5

Answer

```
// You are using GCC
#include <stdio.h>

int main() {
    int n;
    scanf("%d", &n);

    int queue[15];
    for (int i = 0; i < n; i++) {
        scanf("%d", &queue[i]);
    }
    int filtered[15];
    int count = 0;

    for (int i = 0; i < n; i++) {
        if (!(queue[i] > 0 && queue[i] % 2 == 0)) {
            filtered[count++] = queue[i];
```

```
        }
    }
    for (int i = 0; i < count; i++) {
        printf("%d ", filtered[i]);
    }
    printf("\n");

    return 0;
}
```

Status : Correct ***Marks : 10/10***

240701377