Rajalakshmi Engineering College

Name: sugitha nesamani
Email: 241801279@rajalakshmi.edu.in
Roll no: 241801279
Phone: 8637611457
Branch: REC
Department: I AI & DS AF
Batch: 2028
Degree: B.E - AI & DS

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

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, queue[15], filteredQueue[15];
    int i, j = 0;

    // Read number of elements
    scanf("%d", &n);

    // Read the blood cell count values
    for (i = 0; i < n; i++) {
        scanf("%d", &queue[i]);
    }

    // Filter out positive even numbers
    for (i = 0; i < n; i++) {
        if (!(queue[i] > 0 && queue[i] % 2 == 0)) {
            filteredQueue[j++] = queue[i];
        }
    }
```

```
    // Print the remaining values
    for (i = 0; i < j; i++) {
        printf("%d ", filteredQueue[i]);
    }

    printf("\n");

    return 0;
}
```

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

2. 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>
struct Node {
    int data;
    struct Node* next;
};


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

void enqueue(int value) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    newNode->data = value;
    newNode->next = NULL;

    if (rear == NULL) {
        front = rear = newNode;
    } else {
        rear->next = newNode;
        rear = newNode;
    }
}


void dequeue() {
    if (front == NULL) {
        return;
    }
    struct Node* temp = front;
    front = front->next;
```

```
    if (front == NULL)
        rear = NULL;
    free(temp);
}

void printFrontRear() {
    if (front != NULL && rear != NULL) {
        printf("Front: %d, Rear: %d\n", front->data, rear->data);
    }
}

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

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

    dequeue();
    printFrontRear();

    return 0;
}
```

Status : Correct *Marks : 10/10*

3. Problem Statement

Sara builds a linked list-based queue and wants to dequeue and display all positive even numbers in the queue. The numbers are added at the end of the queue.

Help her by writing a program for the same.

Input Format

The first line of input consists of an integer N, representing the number of elements Sara wants to add to the queue.

The second line consists of N space-separated integers, each representing an element to be enqueued.

Output Format

The output prints space-separated the positive even integers from the queue, maintaining the order in which they were enqueued.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
1 2 3 4 5
Output: 2 4

Answer

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

// Define a node in the queue
struct Node {
    int data;
    struct Node* next;
};

    struct Node* createNode(int value) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    newNode->data = value;
    newNode->next = NULL;
    return newNode;
}

void enqueue(struct Node** front, struct Node** rear, int value) {
    struct Node* newNode = createNode(value);
    if (*rear == NULL) {
        *front = *rear = newNode;
    } else {
        (*rear)->next = newNode;
```

```
        *rear = newNode;
    }
}

void dequeueAndPrintPositiveEvens(struct Node* front) {
    while (front != NULL) {
        if (front->data > 0 && front->data % 2 == 0) {
            printf("%d ", front->data);
        }
        front = front->next;
    }
    printf("\n");
}

int main() {
    int N, value;
    struct Node* front = NULL;
    struct Node* rear = NULL;
    scanf("%d", &N);
    for (int i = 0; i < N; i++) {
        scanf("%d", &value);
        enqueue(&front, &rear, value);
    }

    // Dequeue and print only positive even numbers
    dequeueAndPrintPositiveEvens(front);

    return 0;
}
```

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