Rajalakshmi Engineering College

Name: Jeffery Antony J
Email: 241901041@rajalakshmi.edu.in
Roll no: 241901041
Phone: 7305663808
Branch: REC
Department: I CSE (CS) FA
Batch: 2028
Degree: B.E - CSE (CS)

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

Manoj is learning data structures and practising queues using linked lists. His professor gave him a problem to solve. Manoj started solving the program but could not finish it. So, he is seeking your assistance in solving it.

The problem is as follows: Implement a queue with a function to find the Kth element from the end of the queue.

Help Manoj with the program.

Input Format

The first line of input consists of an integer N, representing the number of elements in the queue.

The second line consists of N space-separated integers, representing the queue elements.

The third line consists of an integer K.

Output Format

The output prints an integer representing the Kth element from the end of the queue.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
2 4 6 7 5
3

Output: 6

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;
    }
}


int findKthFromEnd(int k) {
    struct Node* mainPtr = front;
    struct Node* refPtr = front;
    int count = 0;

    while (count < k) {
        refPtr = refPtr->next;
        count++;
    }

    while (refPtr != NULL) {
        mainPtr = mainPtr->next;
        refPtr = refPtr->next;
    }

    return mainPtr->data;
}

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

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

    scanf("%d", &K);
    int result = findKthFromEnd(K);
    printf("%d", result);

    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

```
#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 removeDuplicates() {
    int seen[101] = {0};
    struct Node* current = front;
    struct Node* prev = NULL;

    while (current != NULL) {
        if (seen[current->data]) {
            prev->next = current->next;
            if (current == rear) {
                rear = prev;
            }
            free(current);
            current = prev->next;
        } else {
            seen[current->data] = 1;
            prev = current;
            current = current->next;
        }
    }
}

void printQueue() {
    struct Node* temp = front;
    while (temp != NULL) {
        printf("%d ", temp->data);
        temp = temp->next;
    }
}

int main() {
```

```
    int N, val;
    scanf("%d", &N);
    for (int i = 0; i < N; i++) {
        scanf("%d", &val);
        enqueue(val);
    }
    removeDuplicates();
    printQueue();
    return 0;
}
```

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

3. Problem Statement

Saran is developing a simulation for a theme park where people wait in a queue for a popular ride.

Each person has a unique ticket number, and he needs to manage the queue using a linked list implementation.

Your task is to write a program for Saran that reads the number of people in the queue and their respective ticket numbers, enqueue them, and then calculate the sum of all ticket numbers to determine the total ticket value present in the queue.

Input Format

The first line of input consists of an integer N, representing the number of people in the queue.

The second line consists of N space-separated integers, representing the ticket numbers.

Output Format

The output prints an integer representing the sum of all ticket numbers.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
2 4 6 7 5

Output: 24

Answer

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

struct Node {
    int ticket;
    struct Node* next;
};

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

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

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

int calculateSum() {
    int sum = 0;
    struct Node* temp = front;
    while (temp != NULL) {
        sum += temp->ticket;
        temp = temp->next;
    }
    return sum;
```

```
}

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

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

    int total = calculateSum();
    printf("%d", total);

    return 0;
}
```

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