

# Rajalakshmi Engineering College

Name: Naveed Sheriff  
Email: 240701348@rajalakshmi.edu.in  
Roll no: 240701348  
Phone: 9025573780  
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

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***

```
#include<stdio.h>
```

```
#include<stdlib.h>
```

```
struct node{  
    int data;  
    struct node*next;  
};
```

```
struct node*head = NULL;
```

```
void enqueue(int v)  
{
```

```
    struct node*newnode = (struct node*)malloc(sizeof(struct node));  
    newnode->data = v;  
    newnode->next = NULL;
```

```
    if(head == NULL)  
        head = newnode;
```

```
    else{  
        newnode->next = head;  
        head = newnode;  
    }
```

```
}  
/*void display()  
{  
    struct node*temp = head;  
    while(temp != NULL)  
    {  
        printf("%d ",temp->data);  
        temp = temp->next;  
    }  
    printf("\n");  
}*/
```

```
void ts(int n)  
{  
    struct node*temp = head;  
    int s = 0;  
    int i = 1;  
    while(i<=n)  
    {  
        s = s + temp->data;  
        temp = temp->next;  
        i += 1;  
    }  
    printf("%d",s);  
}
```

```
int main()  
{  
    int n,e;  
    scanf("%d",&n);  
    for(int i=1; i<=n ; i++)  
    {  
        scanf("%d",&e);  
        enqueue(e);  
    }  
    //display();  
    ts(n);  
}
```

**Status :** Correct

**Marks :** 10/10

## 2. 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***

```
#include<stdio.h>
#include<stdlib.h>
```

```
struct node
```

```
{  
    int data;  
    struct node*next;  
};
```

```
struct node*head = NULL;
```

```
void enqueue(int v)
```

```
{  
    struct node*newnode = (struct node*)malloc(sizeof(struct node));  
    newnode->data = v;  
    newnode->next = NULL;  
    if(head == NULL)  
        head = newnode;  
    else  
    {  
        newnode->next = head;  
        head = newnode;  
    }  
}
```

```
void find(int n , int k)
```

```
{  
    int i = 1;  
    struct node*temp = head;  
    int value;  
    while(i<=k)  
    {  
        if(i == k)  
        {  
            value = temp->data;  
            break;  
        }  
        temp = temp->next;  
        i += 1;  
    }  
    printf("%d",value);  
}
```

```
int main()
```

```
{
    int n,e,k;
    scanf("%d",&n);
    for(int i=1;i<=n ; i++)
    {
        scanf("%d",&e);
        enqueue(e);
    }
    scanf("%d",&k);
    find(n,k);
}
```

**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>
```

```
#define max 20
```

```
int queue[max];
```

```
int front = -1;
```

```
int rear = -1;
```

```
void enqueue(int v)
```

```
{
```

```
    if(rear == max - 1)
```

```
    {
```

```
        printf("stack is overflow\n");
```

```
        return;
```

```
    }
```

```
    else{
```

```
        rear += 1;
```

```
        queue[rear] = v;
```

```
        if(front == -1)
```

```
            front = 0;
```

```
    }
```

```
}
```

```
void dequeue(int n)
```

```
{
```

```
    for(int i=front;i<=rear;i++)
```

```
    {
```

```
        if(!(queue[i]>0 && queue[i]%2 == 0))
```

```
        {
```

```
            printf("%d ",queue[i]);
```

```
        }
```

```
    }
```

```
}
```

```
int main()
```

```
{
```

```
    int n,e;
```

```
    scanf("%d",&n);
```

```
    for(int i=0; i<n ; i++)
```

```
    {
```

```
        scanf("%d",&e);
```

```
        enqueue(e);
```

```
    }
```

```
    dequeue(n);
```

```
}
```

**Status :** Correct

**Marks : 10/10**