Rajalakshmi Engineering College

Name: Sajine Santhakumar
Email: 240701459@rajalakshmi.edu.in
Roll no: 240701459
Phone: 9952076750
Branch: REC
Department: I CSE FE
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>
typedef struct node{
   int data;
   struct node*next;
}queue;
queue*front=NULL,*rear=NULL;
void enqueue(int e){
   queue*newnode=(queue*)malloc(sizeof(queue));
   newnode->data=e;
   newnode->next=NULL;
   if(rear==NULL){
      front=rear=newnode;
   }
   else{
      rear->next=newnode;
      rear=newnode;
   }
}
void sum(){
   int sum=0;
   queue*position=front;
   while(position!=NULL){
        sum+=position->data;
        position=position->next;
    }
    printf("%d\n",sum);
}
int main(){
    int n,e;
    scanf("%d",&n);
    for(int i=0;i<n;i++){
        scanf("%d",&e);
        enqueue(e);
    }
    sum();
    return 0;
}
```

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

2. Problem Statement

John is working on a project to manage and analyze the data from various sensors in a manufacturing plant. Each sensor provides a sequence of integer readings, and John needs to process this data to get some insights. He wants to implement a queue to handle these sensor readings efficiently. The requirements are as follows:

Enqueue Operations: Each sensor reading needs to be added to the circular queue.Average Calculation: Calculate and print the average of every pair of consecutive sensor readings.Sum Calculation: Compute the sum of all sensor readings.Even and Odd Count: Count and print the number of even and odd sensor readings.

Assist John in implementing the program.

Input Format

The first input line contains an integer n, which represents the number of sensor readings.

The second line contains n space-separated integers, each representing a sensor reading.

Output Format

The first line should print "Averages of pairs:" followed by the averages of every pair of consecutive sensor readings, separated by spaces.

The second line should print "Sum of all elements: " followed by the sum of all sensor readings.

The third line should print "Number of even elements: " followed by the count of even sensor readings.

The fourth line should print "Number of odd elements: " followed by the count of odd sensor readings.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 5
1 2 3 4 5

Output: Averages of pairs:
1.5 2.5 3.5 4.5 3.0
Sum of all elements: 15
Number of even elements: 2
Number of odd elements: 3

Answer

```
// You are using GCC
#include<stdio.h>
int main(){
    int n;
    scanf("%d",&n);
    int readings[10];
    for(int i=0;i<n;i++){
        scanf("%d",&readings[i]);
    }
    printf("Averages of pairs:\n");
    for(int i=0;i<n;i++){
        int next=(i+1)%n;
        float avg=(readings[i]+readings[next]);
        printf(" %.1f",avg/2);
    }
    int sum=0,even=0,odd=0;
    for(int i=0;i<n;i++){
        sum+=readings[i];
        if(readings[i]%2==0)
          even++;
        else
          odd++;
    }
    printf("\nSum of all elements: %d\n",sum);
    printf("Number of even elements: %d\n",even);
    printf("Number of odd elements: %d\n",odd);
    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

```
#include<stdio.h>
#include<stdlib.h>
struct Node{
   int data;
   struct Node*next;
};
struct queue{
   struct Node*front;
   struct Node*rear;
};
struct Node*createNode(int data){
   struct Node*newnode=(struct Node*)malloc(sizeof(struct Node));
   newnode->data=data;
   newnode->next=NULL;
   return newnode;
}
void enqueue(struct queue* q,int data){
   struct Node*temp=createNode(data);
   if(q->rear==NULL){
      q->front=q->rear=temp;
      return;
   }
   q->rear->next=temp;
   q->rear=temp;
}
void dequeuePositiveEvens(struct queue* q){
   struct Node*current=q->front;
   while(current!=NULL){
      if(current->data>0&&current->data%2==0){
         printf("%d ",current->data);
      }
      current=current->next;
   }
   printf("\n");
}
int main(){
    int n,value;
    scanf("%d",&n);
    struct queue q={NULL,NULL};
    for(int i=0;i<n;i++){
        scanf("%d",&value);
        enqueue(&q,value);
    }
    dequeuePositiveEvens(&q);
    return 0;
}
```

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