Rajalakshmi Engineering College

Name: srikumaran s
Email: 240801331@rajalakshmi.edu.in
Roll no: 240801331
Phone: 9600017405
Branch: REC
Department: I ECE FD
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 0_Arrays and Functions

Attempt : 1
Total Mark : 5
Marks Obtained : 5

Section 1 : Coding

1. Problem Statement

Tim is creating a program to track and analyze student attendance. The program requires two inputs: the total number of students (n) and the total number of class sessions (m). The task is to design and populate an attendance matrix, 'matrix', representing the attendance record of each student for each session.

The program's specific objective is to determine whether the last student on the list attended an even or odd number of classes. This functionality will aid teachers in quickly evaluating the attendance habits of individual students.

Input Format

The first line of input consists of a positive integer n, representing the number of students.

The second line consists of a positive integer m, representing the number of class sessions.

The next n lines consist of m space-separated positive integers representing the number of classes attended by the student.

Output Format

The output displays one of the following results:

If the last session is even the output prints "[LastSession] is even".

If the last session is odd the output prints "[LastSession] is odd".

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 2
2
1 2
3 100
Output: 100 is even

Answer

```
#include<stdio.h>
int main()
{
   int n,m,i,j;
   scanf("%d%d",&n,&m);
   int matrix[n][m];
   for(i=0;i<n;i++)
   {
      for(j=0;j<m;j++)
      {
         scanf("%d",&matrix[i][j]);
      }
   }
   if(matrix[n-1][m-1]%2!=0)
   {
```

```
    printf("%d is odd",matrix[n-1][m-1]);
  }
  else
  {
    printf("%d is even",matrix[n-1][m-1]);
  }
  return 0;

}
```

Status : Correct ***Marks : 1/1***

2. Problem Statement

Write a program that will read a Matrix (two-dimensional arrays) and print the sum of all elements of each row by passing the matrix to a function.

Function Signature: void calculateRowSum(int [ ][ ], int, int)

Input Format

The first line consists of an integer M representing the number of rows.

The second line consists of an integer N representing the number of columns.

The next M lines consist of N space-separated integers in each line representing the elements of the matrix.

Output Format

The output displays the sum of all elements of each row separated by a space.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 3
3
1 2 3
4 5 6

7 8 9

Output: 6 15 24

Answer

```
#include <stdio.h>

// You are using GCC
void calculateRowSum(int matrix[20][20], int rows, int cols) {
    int i,j,sum;
    for(i=0;i<rows;i++)
    {
        sum=0;
        for(j=0;j<cols;j++)
        {
            sum+=matrix[i][j];
        }
        printf("%d ",sum);
    }
}

int main() {
    int matrix[20][20];
    int r, c;

    scanf("%d", &r);
    scanf("%d", &c);

    for (int i = 0; i < r; i++) {
        for (int j = 0; j < c; j++) {
            scanf("%d", &matrix[i][j]);
        }
    }

    calculateRowSum(matrix, r, c);
    return 0;
}
```

Status : Correct ***Marks : 1/1***

3. Problem Statement

Alex, a budding programmer, is tasked with writing a menu-driven program

to perform operations on an array of integers. The operations include finding the smallest number, the largest number, the sum of all numbers, and their average. The program must repeatedly display the menu until Alex chooses to exit.

Write a program to ensure the specified tasks are implemented based on Alex's choices.

Input Format

The first line contains an integer n, representing the number of elements in the array.

The second line contains n space-separated integers representing the array elements.

The subsequent lines contain integers representing the menu choices:

Choice 1: Find and display the smallest number.

Choice 2: Find and display the largest number.

Choice 3: Calculate and display the sum of all numbers.

Choice 4: Calculate and display the average of all numbers as double.

Choice 5: Exit the program.

Output Format

For each valid menu choice, print the corresponding result:

For choice 1, print "The smallest number is: X", where X is the smallest number in the array.

For choice 2, print "The largest number is: X", where X is the largest number in the array.

For choice 3, print "The sum of the numbers is: X", where X is the sum of all numbers in the array.

For choice 4, print "The average of the numbers is: X. XX", where X.XX is the double value representing an average of all numbers in the array, rounded to two

decimal places.

For choice 5, print "Exiting the program".

If an invalid choice is made, print "Invalid choice! Please enter a valid option (1-5)."

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 3
10 20 30
1
5
Output: The smallest number is: 10
Exiting the program

Answer

```
#include<stdio.h>
int main()
{
   int a,i,b,min,max;
   float avg,sum=0;
   scanf("%d",&a);
   int arr[a];
   for(i=0;i<a;i++)
   {
      scanf("%d",&arr[i]);
      sum+=arr[i];
      avg=sum/a;

   }
   while(1>0)
   {
      scanf("%d",&b);
      if(b==1)
      {
         min=arr[0];
         for(i=0;i<a;i++)
```

```
        {
            if(arr[i]<min)
            {
                min=arr[i];
            }
        }
        printf("The smallest number is: %d\n",min);
    }
    else if(b==2)
    {
        max=arr[0];
        for(i=0;i<a;i++)
        {
            if(arr[i]>max)
            {
                max=arr[i];

            }
        }
        printf("The largest number is: %d\n",max);
    }
    else if(b==3)
    {
        printf("The sum of the numbers is: %.0f\n",sum);

    }
    else if(b==4)
    {
        printf("The average of the numbers is: %.2f\n",avg);
    }
    else if(b==5)
    {
        printf("Exiting the program");
        break;
    }
    else
    {
        printf("Invalid choice! please enter a valid option (1-5).");
    }
  }
  return 0;
}
```

Status : Correct **Marks : 1/1**

4. Problem Statement

Saurabh is the manager of a growing tech company. He needs a program to record and analyze the monthly salaries of his employees. The program will take the number of employees and their respective salaries as input and then calculate the average salary, and find the highest and lowest salary among them.

Help Saurabh automate this task efficiently.

Input Format

The first line of input consists of an integer n, representing the number of employees.

The second line consists of n integers, where each integer represents the salary of an employee.

Output Format

The output prints n lines, where each line will display: "Employee i: "Salary

Where i is the employee number (starting from 1) and salary is the respective salary of that employee.

After that, print the average salary in the following format: "Average Salary: "average_salary

Where average_salary is the average salary of all employees, rounded to two decimal places.

Next, print the highest salary in the following format: "Highest Salary: "max_salary

Where max_salary is the highest salary among all employees.

Finally, print the lowest salary in the following format:"Lowest Salary: "min_salary

Where min_salary is the lowest salary among all employees.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
4000
3500
6000
2500
4500

Output: Employee 1: 4000
Employee 2: 3500
Employee 3: 6000
Employee 4: 2500
Employee 5: 4500

Average Salary: 4100.00
Highest Salary: 6000
Lowest Salary: 2500

Answer

```
#include<stdio.h>
int main()
{
   int a,min,max,i;
   float sum=0,avg;
   scanf("%d",&a);
   int arr[a];
   for(i=0;i<a;i++)
   {
      scanf("%d",&arr[i]);
```

```
        sum+=arr[i];
        avg=sum/a;
    }
    max=arr[0];
    min=arr[0];
    for(i=0;i<a;i++)
    {
        if(arr[i]<min)
        {
            min=arr[i];
        }
        else if(arr[i]>max)
        {
            max=arr[i];
        }

    }
    for(i=0;i<a;i++)
    {
        printf("Employee %d: %d\n",i+1,arr[i]);
    }
    printf("\nAverage salary: %.2f\n",avg);
    printf("Highest salary: %d\n",max);
    printf("Lowest salary: %d\n",min);
    return 0;
}
```

Status : Correct **Marks : 1/1**

5. Problem Statement

Write a program that reads an integer 'n' and a square matrix of size 'n x n' from the user. The program should then set all the elements in the lower triangular part of the matrix (including the main diagonal) to zero using a function and display the resulting matrix.

Function Signature: void setZeros(int [ ][ ], int)

Input Format

The first line consists of an integer M representing the number of rows & columns.

The next M lines consist of M space-separated integers in each line representing the elements of the matrix.

Output Format

The output displays the matrix containing M space-separated elements in M lines where the lower triangular elements are replaced with zero.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 3
10 20 30
40 50 60
70 80 90

Output: 0 20 30
0 0 60
0 0 0

Answer

```
#include <stdio.h>

// You are using GCC
void setZeros(int arr[10][10], int n) {
    int i,j;
    for(i=0;i<n;i++)
    {
        for(j=0;j<n;j++)
        {
            if(i==j)
            {
                arr[i][j]=0;
            }
            else if(i>j)
            {
                arr[i][j]=0;
            }
        }
    }
```

```
}
int main() {
    int arr1[10][10];
    int n;

    scanf("%d", &n);

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            scanf("%d", &arr1[i][j]);
        }
    }

    setZeros(arr1, n);

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            printf("%d ", arr1[i][j]);
        }
        printf("\n");
    }

    return 0;
}
```

Status : Correct **Marks : 1/1**

Rajalakshmi Engineering College

Name: srikumaran s
Email: 240801331@rajalakshmi.edu.in
Roll no: 240801331
Phone: 9600017405
Branch: REC
Department: I ECE FD
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 1_COD_Question 1

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Janani is a tech enthusiast who loves working with polynomials. She wants to create a program that can add polynomial coefficients and provide the sum of their coefficients.

The polynomials will be represented as a linked list, where each node of the linked list contains a coefficient and an exponent. The polynomial is represented in the standard form with descending order of exponents.

Input Format

The first line of input consists of an integer n, representing the number of terms in the first polynomial.

The following n lines of input consist of two integers each: the coefficient and the exponent of the term in the first polynomial.

The next line of input consists of an integer m, representing the number of terms in the second polynomial.

The following m lines of input consist of two integers each: the coefficient and the exponent of the term in the second polynomial.

Output Format

The output prints the sum of the coefficients of the polynomials.

Sample Test Case

Input: 3
2 2
3 1
4 0
3
2 2
3 1
4 0

Output: 18

Answer

```
#include<stdio.h>
#include<stdlib.h>
typedef struct Poly
{
   int coeff;
   int expon;
   struct Poly* next;
}Node;
Node*newnode(int coeff,int expon)
{
   Node*new_node=(Node*)malloc(sizeof(Node));
   new_node->coeff=coeff;
   new_node->expon=expon;
   new_node->next=NULL;
   return new_node;
}
void insertNode(Node** head,int coeff,int expon)
{
   Node*temp=*head;
   if(temp==NULL)
```

```
    {
        *head=newnode(coeff,expon);
        return;
    }
    while(temp->next!=NULL)
    {
        temp=temp->next;
    }
    temp->next=newnode(coeff,expon);
}
int main()
{
    int n,coeff,expon;
    scanf("%d",&n);
    Node*poly1;
    Node*poly2;
    for(int i=0;i<n;i++)
    {
        scanf("%d %d",&coeff,&expon);
        insertNode(&poly1,coeff,expon);
    }
    scanf("%d",&n);
    for(int i=0;i<n;i++)
    {
        scanf("%d %d",&coeff,&expon);
        insertNode(&poly2,coeff,expon);
    }
    int sum=0;
    while(poly1!=NULL)
    {
        sum+=poly1->coeff;
        poly1=poly1->next;
    }
    while(poly2!=NULL)
    {
        sum+=poly2->coeff;
        poly2=poly2->next;
    }
    printf("%d",sum);
}
```

Status : Correct *Marks : 10/10*

Rajalakshmi Engineering College

Name: srikumaran s
Email: 240801331@rajalakshmi.edu.in
Roll no: 240801331
Phone: 9600017405
Branch: REC
Department: I ECE FD
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 1_COD_Question 2

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Arun is learning about data structures and algorithms. He needs your help in solving a specific problem related to a singly linked list.

Your task is to implement a program to delete a node at a given position. If the position is valid, the program should perform the deletion; otherwise, it should display an appropriate message.

Input Format

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

The second line consists of N space-separated elements of the linked list.

The third line consists of an integer x, representing the position to delete.

Position starts from 1.

Output Format

The output prints space-separated integers, representing the updated linked list after deleting the element at the given position.

If the position is not valid, print "Invalid position. Deletion not possible."

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
8 2 3 1 7
2
Output: 8 3 1 7

Answer

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

void insert(int);
void display_List();
void deleteNode(int);

struct node {
   int data;
   struct node* next;
} *head = NULL, *tail = NULL;

void insert(int value)
{
   if(head==NULL)
   {
      head=(struct node*)malloc(sizeof(struct node));
      head->data=value;
      head->next=NULL;
   }
   else
   {
```

```
        struct node* temp=head;
        while(temp->next!=NULL)
        {
            temp=temp->next;
        }
        temp->next=(struct node*)malloc(sizeof(struct node));
        temp->next->data=value;
        temp->next->next=NULL;
    }
}
void display_List()
{
    struct node*list=head;
    while(list!=NULL)
    {
        printf("%d ",list->data);
        list=list->next;
    }
}
void deleteNode(int pos)
{
    int size=0;
    struct node*temp=head;
    while(temp!=NULL)
    {
        size++;
        temp=temp->next;
    }
    if(size<pos)
    {
        printf("Invalid position. Deletion not possible.",size);
    }
    else
    {
        pos-=1;
        if(pos==0)
        {
            temp=head->next;
            free(head);
            head=temp;
        }
        else
```

```
        {
            temp=head;
             while(--pos)
             {
                temp=temp->next;
             }
        struct node* temp1=temp->next;
        temp->next=temp->next->next;
        free(temp1);
        }
        display_List();
    }
}

int main() {
    int num_elements, element, pos_to_delete;

    scanf("%d", &num_elements);

    for (int i = 0; i < num_elements; i++) {
        scanf("%d", &element);
        insert(element);
    }

    scanf("%d", &pos_to_delete);

    deleteNode(pos_to_delete);

    return 0;
}
```

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

Rajalakshmi Engineering College

Name: srikumaran s
Email: 240801331@rajalakshmi.edu.in
Roll no: 240801331
Phone: 9600017405
Branch: REC
Department: I ECE FD
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 1_COD_Question 3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Imagine you are working on a text processing tool and need to implement a feature that allows users to insert characters at a specific position.

Implement a program that takes user inputs to create a singly linked list of characters and inserts a new character after a given index in the list.

Input Format

The first line of input consists of an integer N, representing the number of characters in the linked list.

The second line consists of a sequence of N characters, representing the linked list.

The third line consists of an integer index, representing the index(0-based) after

which the new character node needs to be inserted.

The fourth line consists of a character value representing the character to be inserted after the given index.

Output Format

If the provided index is out of bounds (larger than the list size):

1. The first line of output prints "Invalid index".
2. The second line prints "Updated list: " followed by the unchanged linked list values.

Otherwise, the output prints "Updated list: " followed by the updated linked list after inserting the new character after the given index.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
a b c d e
2
X
Output: Updated list: a b c X d e

Answer

```
#include<stdio.h>
#include<stdlib.h>
typedef struct Char
{
   char value;
   struct Char*next;
}Node;
Node*newnode(char value)
{
   Node*new_node=(Node*)malloc(sizeof(Node));
   new_node->value=value;
   new_node->next=NULL;
```

```
    return new_node;
}
void insertNode(Node** head,char value)
{
    Node* temp= *head;
    if(temp==NULL)
    {
        *head=newnode(value);
        return;
    }
    while(temp->next!=NULL)
    {
        temp=temp->next;
    }
    temp->next=newnode(value);
}
int length(Node*head)
{
    int len=0;
    while(head!=NULL)
    {
        head=head->next;
        len++;
    }
    return len;
}
void traverse(Node*head)
{
    while(head!=NULL)
    {
        printf("%c ",head->value);
        head=head->next;
    }
    printf("\n");
}
void insert(Node** head,int pos,char value)
{
    if(pos>=length(*head))
    {
        printf("Invalid index\n");
        return;
    }
```

```
    Node*temp= *head;
    for(int i=0;i<pos;i++)
    {
        temp=temp->next;
    }
    Node*new_node=newnode(value);
    new_node->next=temp->next;
    temp->next=new_node;
}
int main()
{
    int n;
    char value;
    Node*head=NULL;
    scanf("%d",&n);
    for(int i=0;i<=n;i++)
    {
        scanf("%c ",&value);
        if(value==' ' || value=='\n')
        {
            continue;
        }
        insertNode(&head,value);
    }
    scanf("%d %c",&n,&value);
    insert(&head,n,value);
    printf("Updated list: ");
    traverse(head);
}
```

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

Rajalakshmi Engineering College

Name: srikumaran s
Email: 240801331@rajalakshmi.edu.in
Roll no: 240801331
Phone: 9600017405
Branch: REC
Department: I ECE FD
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 1_COD_Question 4

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

As part of a programming assignment in a data structures course, students are required to create a program to construct a singly linked list by inserting elements at the beginning.

You are an evaluator of the course and guide the students to complete the task.

Input Format

The first line of input consists of an integer N, which is the number of elements.

The second line consists of N space-separated integers.

Output Format

The output prints the singly linked list elements, after inserting them at the beginning.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
78 89 34 51 67

Output: 67 51 34 89 78

Answer

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

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

void insertAtFront(struct Node** head,int activity)
{
   struct Node*newnode=(struct Node*)malloc(sizeof(struct Node));
   newnode->data=activity;
   newnode->next=*head;
   *head=newnode;
}
void printList(struct Node* head)
{
   while(head!=NULL)
   {
      printf("%d ",head->data);
      head=head->next;
   }
}

int main(){
   struct Node* head = NULL;

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

```
    for (int i = 0; i < n; i++) {
        int activity;
        scanf("%d", &activity);
        insertAtFront(&head, activity);
    }

    printList(head);
    struct Node* current = head;
    while (current != NULL) {
        struct Node* temp = current;
        current = current->next;
        free(temp);
    }

    return 0;
}
```

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

Rajalakshmi Engineering College

Name: srikumaran s
Email: 240801331@rajalakshmi.edu.in
Roll no: 240801331
Phone: 9600017405
Branch: REC
Department: I ECE FD
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 1_COD_Question 5

Attempt : 1
Total Mark : 10
Marks Obtained : 0

Section 1 : Coding

1. Problem Statement

Imagine you are tasked with developing a simple GPA management system using a singly linked list. The system allows users to input student GPA values, insertion should happen at the front of the linked list, delete record by position, and display the updated list of student GPAs.

Input Format

The first line of input contains an integer n, representing the number of students.

The next n lines contain a single floating-point value representing the GPA of each student.

The last line contains an integer position, indicating the position at which a student record should be deleted. Position starts from 1.

Output Format

After deleting the data in the given position, display the output in the format "GPA: " followed by the GPA value, rounded off to one decimal place.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 4
3.8
3.2
3.5
4.1
2
Output: GPA: 4.1
GPA: 3.2
GPA: 3.8

Answer

```
// You are using GCC
```

Status : Wrong ***Marks : 0/10***

Rajalakshmi Engineering College

Name: srikumaran s
Email: 240801331@rajalakshmi.edu.in
Roll no: 240801331
Phone: 9600017405
Branch: REC
Department: I ECE FD
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 2_COD_Question 3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Bob is tasked with developing a company's employee record management system. The system needs to maintain a list of employee records using a doubly linked list. Each employee is represented by a unique integer ID.

Help Bob to complete a program that adds employee records at the front, traverses the list, and prints the same for each addition of employees to the list.

Input Format

The first line of input consists of an integer N, representing the number of employees.

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

Output Format

For each employee ID, the program prints "Node Inserted" followed by the current state of the doubly linked list in the next line, with the data values of each node separated by spaces.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 4
101 102 103 104
Output: Node Inserted
101
Node Inserted
102 101
Node Inserted
103 102 101
Node Inserted
104 103 102 101

Answer

```
#include <iostream>
using namespace std;

struct node {
    int info;
    struct node* prev, * next;
};

struct node* start = NULL;

// You are using GCC
void traverse() {
    struct node*temp=start;
    printf("Node Inserted\n");
    while(temp!=NULL)
    {
        printf("%d ",temp->info);
        temp=temp->next;
    }
```

```
    printf("\n");
}

void insertAtFront(int data) {
    struct node*temp=(struct node*)malloc(sizeof(struct node));
    temp->info=data;
    temp->prev=NULL;
    temp->next=NULL;
    if(start == NULL)
    {
        start=temp;
        return;
    }
    else if(start->prev==NULL)
    {
        start->prev=temp;
        temp->next=start;
        start=temp;
        return;
    }

}

int main() {
    int n, data;
    cin >> n;
    for (int i = 0; i < n; ++i) {
        cin >> data;
        insertAtFront(data);
        traverse();
    }
    return 0;
}
```

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

Rajalakshmi Engineering College

Name: srikumaran s
Email: 240801331@rajalakshmi.edu.in
Roll no: 240801331
Phone: 9600017405
Branch: REC
Department: I ECE FD
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 2_COD_Question 4

Attempt : 1
Total Mark : 10
Marks Obtained : 0

Section 1 : Coding

1. Problem Statement

Ravi is developing a student registration system for a college. To efficiently store and manage the student IDs, he decides to implement a doubly linked list where each node represents a student's ID.

In this system, each student's ID is stored sequentially, and the system needs to display all registered student IDs in the order they were entered.

Implement a program that creates a doubly linked list, inserts student IDs, and displays them in the same order.

Input Format

The first line contains an integer N the number of student IDs.

The second line contains N space-separated integers representing the student IDs.

Output Format

The output should display the single line containing N space-separated integers representing the student IDs stored in the doubly linked list.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
10 20 30 40 50
Output: 10 20 30 40 50

Answer

```
#include<stdio.h>
#include<stdlib.h>
struct node
{
   int data;
   struct node*next,*prev;
};
typedef struct node Node;
Node*start=NULL;
void display()
{
   Node*temp=(Node*)malloc(sizeof(Node));
   temp=start;
   while(temp!=NULL)
   {
      printf("%d ",temp->data);
      temp=temp->next;
   }
}
void insert(int value)
{
   Node*temp=(Node*)malloc(sizeof(Node));
   temp->data=value;
   temp->next=NULL;
```

```
    temp->next=NULL;
    if(start==NULL)
    {
       start=temp;
    }
    Node*tra=(Node*)malloc(sizeof(Node));
    while(tra->next!=NULL)
    {
       tra=tra->next;

    tra->next=temp;
    temp->prev=tra;

}
int main()
{
    int a,b;    scanf("%d",&a)
for(int i=0;i<n;i++)
{
    scanf("%d",&b)
    insert(b)
}
display()
}
```

Status : Wrong **Marks : 0/10**

Rajalakshmi Engineering College

Name: srikumaran s
Email: 240801331@rajalakshmi.edu.in
Roll no: 240801331
Phone: 9600017405
Branch: REC
Department: I ECE FD
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 4_COD_Question 1

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Imagine a bustling coffee shop, where customers are placing their orders for their favorite coffee drinks. The cafe owner Sheeren wants to efficiently manage the queue of coffee orders using a digital system. She needs a program to handle this queue of orders.

You are tasked with creating a program that implements a queue for coffee orders. Each character in the queue represents a customer's coffee order, with 'L' indicating a latte, 'E' indicating an espresso, 'M' indicating a macchiato, 'O' indicating an iced coffee, and 'N' indicating a nabob.

Customers can place orders and enjoy their delicious coffee drinks.

Input Format

The input consists of integers corresponding to the operation that needs to be performed:

Choice 1: Enqueue the coffee order into the queue. If the choice is 1, the following input is a space-separated character ('L', 'E', 'M', 'O', 'N').

Choice 2: Dequeue a coffee order from the queue.

Choice 3: Display the orders in the queue.

Choice 4: Exit the program.

Output Format

The output displays messages according to the choice and the status of the queue:

If the choice is 1:

1. Insert the given order into the queue and display "Order for [order] is enqueued." where [order] is the coffee order that is inserted.
2. If the queue is full, print "Queue is full. Cannot enqueue more orders."

If the choice is 2:

1. Dequeue a character from the queue and display "Dequeued Order: " followed by the corresponding order that is dequeued.
2. If the queue is empty without any orders, print "No orders in the queue."

If the choice is 3:

1. The output prints "Orders in the queue are: " followed by the space-separated orders present in the queue.
2. If there are no orders in the queue, print "Queue is empty. No orders available."

If the choice is 4:

1. Exit the program and print "Exiting program"

If any other choice is entered, the output prints "Invalid option."

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

Sample Test Case

Input: 1 L
1 E
1 M
1 O
1 N
1 O
3
2
3
4

Output: Order for L is enqueued.
Order for E is enqueued.
Order for M is enqueued.
Order for O is enqueued.
Order for N is enqueued.
Queue is full. Cannot enqueue more orders.
Orders in the queue are: L E M O N
Dequeued Order: L
Orders in the queue are: E M O N
Exiting program

Answer

```
#include <stdio.h>
#define MAX_SIZE 5

char orders[MAX_SIZE];
int front = -1;
int rear = -1;

void initializeQueue() {
    front = -1;
    rear = -1;
}
// You are using GCC
```

```
int isEmpty() {
   if(front==-1)
   return 1;
   else
   return 0;
}

int isFull() {
   if(rear==MAX_SIZE-1)
   return 1;
   else
   return 0;
}

int enqueue(char order) {
   if(isFull())
   {
      printf("Queue is full. Cannot enqueue more orders.\n");
      return 0;
   }
   else
   {
      rear++;
      orders[rear]=order;
      if(front==-1)
      front=0;
      printf("Order for %c is enqueued.\n",order);
      return 1;
   }
}

int dequeue() {
   if(isEmpty())
   printf("No orders  in the queue.\n");
   else{
      char s=orders[front];
      if(front==rear)
      initializeQueue();
      else
      front++;
      printf("Dequeued Order: %c\n",s);
```

```
    }
    return 0;
}

void display() {
    if(isEmpty())
    printf("Queue is Empty. No orders available.\n");
    else{
        printf("Orders in the queue are: ");
        for(int i=front;i<=rear;i++)
        printf("%c ",orders[i]);
    }
    printf("\n");
}
int main() {
    char order;
    int option;
    initializeQueue();
    while (1) {
        if (scanf("%d", &option) != 1) {
            break;
        }
        switch (option) {
            case 1:
                if (scanf(" %c", &order) != 1) {
                    break;
                }
                if (enqueue(order)) {
                }
                break;
            case 2:
                dequeue();
                break;
            case 3:
                display();
                break;
            case 4:
                printf("Exiting program");
                return 0;
            default:
                printf("Invalid option.\n");
                break;
```

```
    }
  }
  return 0;
}
```

Status : Correct **_Marks : 10/10_**

Rajalakshmi Engineering College

Name: srikumaran s
Email: 240801331@rajalakshmi.edu.in
Roll no: 240801331
Phone: 9600017405
Branch: REC
Department: I ECE FD
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 4_COD_Question 2

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

In a bustling IT department, staff regularly submit helpdesk tickets to request technical assistance. Managing these tickets efficiently is vital for providing quality support.

Your task is to develop a program that uses an array-based queue to handle and prioritize helpdesk tickets based on their unique IDs.

Implement a program that provides the following functionalities:

Enqueue Helpdesk Ticket: Add a new helpdesk ticket to the end of the queue. Provide a positive integer representing the ticket ID for the new ticket.Dequeue Helpdesk Ticket: Remove and process the next helpdesk ticket from the front of the queue. The program will display the ticket ID of the processed ticket.Display Queue: Display the ticket IDs of all the

helpdesk tickets currently in the queue.

Input Format

The input consists of integers corresponding to the operation that needs to be performed:

Choice 1: Enqueue the ticket ID into the queue. If the choice is 1, the following input is a space-separated integer, representing the ticket ID to be enqueued into the queue.

Choice 2: Dequeue a ticket from the queue.

Choice 3: Display the ticket IDs in the queue.

Choice 4: Exit the program.

Output Format

The output displays messages according to the choice and the status of the queue:

If the choice is 1:

1. Insert the given ticket ID into the queue and display "Helpdesk Ticket ID [id] is enqueued." where [id] is the ticket ID that is inserted.
2. If the queue is full, print "Queue is full. Cannot enqueue."

If the choice is 2:

1. Dequeue a ticket ID from the queue and display "Dequeued Helpdesk Ticket ID: " followed by the corresponding ID that is dequeued.
2. If the queue is empty without any elements, print "Queue is empty."

If the choice is 3:

1. The output prints "Helpdesk Ticket IDs in the queue are: " followed by the space-separated ticket IDs present in the queue.
2. If there are no elements in the queue, print "Queue is empty."

If the choice is 4:

1. Exit the program and print "Exiting the program"

If any other choice is entered, print "Invalid option."

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 1 101
1 202
1 203
1 204
1 205
1 206
3
2
3
4

Output: Helpdesk Ticket ID 101 is enqueued.
Helpdesk Ticket ID 202 is enqueued.
Helpdesk Ticket ID 203 is enqueued.
Helpdesk Ticket ID 204 is enqueued.
Helpdesk Ticket ID 205 is enqueued.
Queue is full. Cannot enqueue.
Helpdesk Ticket IDs in the queue are: 101 202 203 204 205
Dequeued Helpdesk Ticket ID: 101
Helpdesk Ticket IDs in the queue are: 202 203 204 205
Exiting the program

Answer

```
#include <stdio.h>
#define MAX_SIZE 5

int ticketIDs[MAX_SIZE];
int front = -1;
int rear = -1;
int lastDequeued;

void initializeQueue() {
    front = -1;
    rear = -1;
}
```

```
// You are using GCC
int isEmpty() {
  if(front==-1)
  return 1;
  else
  return 0;
}

int isFull() {
   if(rear==MAX_SIZE-1)
   return 1;
   else
   return 0;
}

int enqueue(int ticketID) {
    if(isFull())
    printf("Queue is full. Cannot enqueue.\n");
    else{
        rear++;
        ticketIDs[rear]=ticketID;
        if(front==-1)
        front=0;
        printf("Helpdesk Ticket ID %d is enqueued.\n",ticketID);
    }
    return 0;
}

int dequeue() {
   if(isEmpty())
   return 0;
   else{
       lastDequeued=ticketIDs[front];
       if(rear==front)
       initializeQueue();
       else
       front++;
       return 1;
   }
}

void display() {
```

```
    if(isEmpty())
    printf("Queue is empty.\n");
    else{
        printf("Helpdesk Ticket IDs in the queue are: ");
        for(int i=front;i<=rear;i++)
        printf("%d ",ticketIDs[i]);
        printf("\n");
    }
}

int main() {
    int ticketID;
    int option;
    initializeQueue();
    while (1) {
        if (scanf("%d", &option) == EOF) {
            break;
        }
        switch (option) {
            case 1:
                if (scanf("%d", &ticketID) == EOF) {
                    break;
                }
                enqueue(ticketID);
                break;
            case 2:
                if (dequeue()) {
                    printf("Dequeued Helpdesk Ticket ID: %d\n", lastDequeued);
                } else {
                    printf("Queue is empty.\n");
                }
                break;
            case 3:
                display();
                break;
            case 4:
                printf("Exiting the program\n");
                return 0;
            default:
                printf("Invalid option.\n");
                break;
        }
    }
```

```
    return 0;
}
```

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

Rajalakshmi Engineering College

Name: srikumaran s
Email: 240801331@rajalakshmi.edu.in
Roll no: 240801331
Phone: 9600017405
Branch: REC
Department: I ECE FD
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 4_COD_Question 3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Write a program to implement a queue using an array and pointers. The program should provide the following functionalities:

Insert an element into the queue.Delete an element from the queue.Display the elements in the queue.

The queue has a maximum capacity of 5 elements. If the queue is full and an insertion is attempted, a "Queue is full" message should be displayed. If the queue is empty and a deletion is attempted, a "Queue is empty" message should be displayed.

Input Format

Each line contains an integer representing the chosen option from 1 to 3.

Option 1: Insert an element into the queue followed by an integer representing the element to be inserted, separated by a space.

Option 2: Delete an element from the queue.

Option 3: Display the elements in the queue.

Output Format

For option 1 (insertion):-

1. The program outputs: "<data> is inserted in the queue." if the data is successfully inserted.
2. "Queue is full." if the queue is already full and cannot accept more elements.

For option 2 (deletion):-

1. The program outputs: "Deleted number is: <data>" if an element is successfully deleted and returns the value of the deleted element.
2. "Queue is empty." if the queue is empty no elements can be deleted.

For option 3 (display):-

1. The program outputs: "Elements in the queue are: <element1> <element2> ... <elementN>" where <element1>, <element2>, ..., <elementN> represent the elements present in the queue.
2. "Queue is empty." if the queue is empty no elements can be displayed.

For invalid options, the program outputs: "Invalid option."

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 1 10

3
5
Output: 10 is inserted in the queue.
Elements in the queue are: 10
Invalid option.

Answer

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

#define max 5

int queue[max];
int front = -1, rear = -1;

// You are using GCC
int insertq(int *data)
{
   if(rear==max-1)
   return 0;
   else{
      rear++;
      queue[rear]=*data;
      if(front ==-1)
      front++;
      return 1;
   }
}

int delq()
{
   if(front==-1)
   printf("Queue is empty.\n");
   else{
      printf("Deleted number is: %d\n",queue[front]);
      if(front==rear)
      front=rear=-1;
      else
      front++;;
   }
   return 0;
}
```

```
void display()
{
    if(front==-1)
    printf("Queue is empty.\n");
    else{
        printf("Elements in the queue are: ");
        for(int i=front;i<=rear;i++)
        printf("%d ",queue[i]);
        printf("\n");
    }
}

int main()
{
    int data, reply, option;
    while (1)
    {
        if (scanf("%d", &option) != 1)
            break;
        switch (option)
        {
            case 1:
                if (scanf("%d", &data) != 1)
                    break;
                reply = insertq(&data);
                if (reply == 0)
                    printf("Queue is full.\n");
                else
                    printf("%d is inserted in the queue.\n", data);
                break;
            case 2:
                delq();  //    Called without arguments
                break;
            case 3:
                display();
                break;
            default:
                printf("Invalid option.\n");
                break;
        }
    }
    return 0;
```

}

Status : Correct *Marks : 10/10*

Rajalakshmi Engineering College

Name: srikumaran s
Email: 240801331@rajalakshmi.edu.in
Roll no: 240801331
Phone: 9600017405
Branch: REC
Department: I ECE FD
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 4_COD_Question 4

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

In an office setting, a print job management system is used to efficiently handle and process print jobs. The system is implemented using a queue data structure with an array.

The program provides the following operations:

Enqueue Print Job: Add a print job with a specified number of pages to the end of the queue.Dequeue Print Job: Remove and process the next print job in the queue.Display Queue: Display the print jobs in the queue

The program should ensure that print jobs are processed in the order they are received.

Input Format

The input consists of integers corresponding to the operation that needs to be performed:

Choice 1: Enqueue the print job into the queue. If the choice is 1, the following input is a space-separated integer, representing the pages to be enqueued into the queue.

Choice 2: Dequeue a print job from the queue.

Choice 3: Display the print jobs in the queue.

Choice 4: Exit the program.

Output Format

The output displays messages according to the choice and the status of the queue:

If the choice is 1:

1. Insert the given page into the queue and display "Print job with [page] pages is enqueued." where [page] is the number of pages that are inserted.
2. If the queue is full, print "Queue is full. Cannot enqueue."

If the choice is 2:

1. Dequeue a page from the queue and display "Processing print job: [page] pages" where [page] is the corresponding page that is dequeued.
2. If the queue is empty without any elements, print "Queue is empty."

If the choice is 3:

1. The output prints "Print jobs in the queue: " followed by the space-separated pages present in the queue.
2. If there are no elements in the queue, print "Queue is empty."

If the choice is 4:

1. Exit the program and print "Exiting program"

If any other choice is entered, the output prints "Invalid option."

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 1
10
1
20
1
30
1
40
1
50
1
60
3
2
3
4

Output: Print job with 10 pages is enqueued.
Print job with 20 pages is enqueued.
Print job with 30 pages is enqueued.
Print job with 40 pages is enqueued.
Print job with 50 pages is enqueued.
Queue is full. Cannot enqueue.
Print jobs in the queue: 10 20 30 40 50
Processing print job: 10 pages
Print jobs in the queue: 20 30 40 50
Exiting program

Answer

```
// You are using GCC
void enqueue(int pages){
    if(rear==MAX_SIZE-1)
   printf("Queue is full. Cannot enqueue.\n");
    else{
        rear++;
```

```
        queue[rear]=pages;
        if(front==-1)
        front++;
        printf("Print job with %d pages is enqueued.\n",pages);
    }
}

void dequeue(){
    if(rear==-1)
    printf("Queue is empty.\n");
    else{
        printf("Processing print job: %d pages\n",queue[front]);
        if(front==rear)
        front=rear=-1;
        else
        front++;
    }
}

void display(){
    if(rear==-1)
    printf("Queue is empty.\n");
    else{
        printf("Print jobs in the queue: ");
        for(int i=front;i<=rear;i++)
        printf("%d ",queue[i]);
        printf("\n");
    }
}
```

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

Rajalakshmi Engineering College

Name: srikumaran s
Email: 240801331@rajalakshmi.edu.in
Roll no: 240801331
Phone: 9600017405
Branch: REC
Department: I ECE FD
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 5_COD_Question 1

Attempt : 1
Total Mark : 10
Marks Obtained : 0

Section 1 : Coding

1. Problem Statement

John is learning about Binary Search Trees (BST) in his computer science class. He wants to create a program that allows users to delete a node with a given value from a BST and print the remaining nodes using an in-order traversal.

Implement a function to help him delete a node with a given value from a BST.

Input Format

The first line of input consists of an integer N, representing the number of nodes in the BST.

The second line consists of N space-separated integers, representing the values of the BST nodes.

The third line consists of an integer V, which is the value to delete from the BST.

Output Format

The output prints the space-separated values in the BST in an in-order traversal, after the deletion of the specified value.

If the specified value is not available in the tree, print the given input values in-order traversal.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
10 5 15 2 7
15

Output: 2 5 7 10

Answer

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

struct TreeNode {
    int data;
    struct TreeNode* left;
    struct TreeNode* right;
};

struct TreeNode* createNode(int key) {
    struct TreeNode* newNode = (struct TreeNode*)malloc(sizeof(struct
TreeNode));
    newNode->data = key;
    newNode->left = newNode->right = NULL;
    return newNode;
}

// You are using GCC
struct TreeNode* insert(struct TreeNode* root, int key) {
    if(root==NULL)
```

```
    createNode(key)
    else if(e>root->data)
        root->right=insert(root->right,key);
    else if(e<root->data)
        root->left=insert(root->left,key);
    return root;




}

struct TreeNode* findMin(struct TreeNode* root) {
    if(root==NULL)
        return NULL;
    else if(key>root->data)
        root->right=findmin(root->right,key);

    else if(key<root->data)
        root->left=findmin(root->left,key);









}

struct TreeNode* deleteNode(struct TreeNode* root, int key) {
    //Type your code here
}

void inorderTraversal(struct TreeNode* root) {
    //Type your code here
}
```

```
int main()
{
    int N, rootValue, V;
    scanf("%d", &N);
    struct TreeNode* root = NULL;
    for (int i = 0; i < N; i++) {
        int key;
        scanf("%d", &key);
        if (i == 0) rootValue = key;
        root = insert(root, key);
    }
    scanf("%d", &V);
    root = deleteNode(root, V);
    inorderTraversal(root);
    return 0;
}
```

Status : Wrong **Marks : 0/10**

Rajalakshmi Engineering College

Name: srikumaran s
Email: 240801331@rajalakshmi.edu.in
Roll no: 240801331
Phone: 9600017405
Branch: REC
Department: I ECE FD
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 5_COD_Question 2

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Mike is learning about Binary Search Trees (BSTs) and wants to implement various operations on them. He wants to write a basic program for creating a BST, inserting nodes, and printing the tree in the pre-order traversal.

Write a program to help him solve this program.

Input Format

The first line of input consists of an integer N, representing the number of values to insert into the BST.

The second line consists of N space-separated integers, representing the values to insert into the BST.

Output Format

The output prints the space-separated values of the BST in the pre-order traversal.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
3 1 5 2 4

Output: 3 1 2 5 4

Answer

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

struct Node {
    int data;
    struct Node* left;
    struct Node* right;
};

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

// You are using GCC
struct Node* insert(struct Node* root, int value) {
    if(root==NULL){
        return createNode(value);
    }
    if(value< root->data){
        root->left=insert(root->left,value);
    }
    else if(value> root->data){
        root->right=insert(root->right,value);
    }
      return root;
```

```
}
void printPreorder(struct Node* node) {
    if(node!=NULL)
    {

    printf("%d ",node->data);
    printPreorder(node->left);
    printPreorder(node->right);
    }

}
int main() {
    struct Node* root = NULL;

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

    for (int i = 0; i < n; i++) {
        int value;
        scanf("%d", &value);
        root = insert(root, value);
    }

    printPreorder(root);
    return 0;
}
```

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

Rajalakshmi Engineering College

Name: srikumaran s
Email: 240801331@rajalakshmi.edu.in
Roll no: 240801331
Phone: 9600017405
Branch: REC
Department: I ECE FD
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 5_COD_Question 3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

You are required to implement basic operations on a Binary Search Tree (BST), like insertion and searching.

Insertion: Given a list of integers, construct a Binary Search Tree by repeatedly inserting each integer into the tree according to the rules of a BST.

Searching: Given an integer, search for its presence in the constructed Binary Search Tree. Print whether the integer is found or not.

Write a program to calculate this efficiently.

Input Format

The first line of input consists of an integer n, representing the number of nodes

in the binary search tree.

The second line consists of the values of the nodes, separated by space as integers.

The third line consists of an integer representing, the value that is to be searched.

Output Format

The output prints, "Value <value> is found in the tree." if the given value is present, otherwise it prints: "Value <value> is not found in the tree."

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 7
8 3 10 1 6 14 23
6
Output: Value 6 is found in the tree.

Answer

```
// You are using GCC

struct Node* insertNode(struct Node* root, int value) {
    if(root==NULL)
        return createNode(value);
    if(value<root->data)
        root->left=insertNode(root->left,value);
    else if(value>root->data)
        root->right=insertNode(root->right,value);
    return root;
}
struct Node* searchNode(struct Node* root, int value) {
    while(root!=NULL){
        if(value==root->data)
            return root;
        else if(value<root->data)
            root=root->left;
        else
            root=root->right;
```

```
    }
    return NULL;
}
```

Status : Correct *Marks : 10/10*

Rajalakshmi Engineering College

Name: srikumaran s
Email: 240801331@rajalakshmi.edu.in
Roll no: 240801331
Phone: 9600017405
Branch: REC
Department: I ECE FD
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 5_COD_Question 4

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

John, a computer science student, is learning about binary search trees (BST) and their properties. He decides to write a program to create a BST, display it in post-order traversal, and find the minimum value present in the tree.

Help him by implementing the program.

Input Format

The first line of input consists of an integer N, representing the number of elements to insert into the BST.

The second line consists of N space-separated integers data, which is the data to be inserted into the BST.

Output Format

The first line of output prints the space-separated elements of the BST in post-order traversal.

The second line prints the minimum value found in the BST.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 3
5 10 15
Output: 15 10 5
The minimum value in the BST is: 5

Answer

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

struct Node {
    int data;
    struct Node* left;
    struct Node* right;
};

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

struct Node* insert(struct Node* root, int data) {
    if(root==NULL){
        return createNode(data);
    }
    if(data<root->data)
    root->left=insert(root->left,data);
    else if(data>root->data)
    root->right=insert(root->right,data);
```

```
    return root;
}
void displayTreePostOrder(struct Node* root) {
    if(root!=NULL){
    displayTreePostOrder(root->left);
    displayTreePostOrder(root->right);
    printf("%d ",root->data);
    }
}

int findMinValue(struct Node* root) {
    while(root && root->left!=NULL)
    root=root->left;
    return root->data;
}
int main() {
    struct Node* root = NULL;
    int n, data;
    scanf("%d", &n);

    for (int i = 0; i < n; i++) {
        scanf("%d", &data);
        root = insert(root, data);
    }

    displayTreePostOrder(root);
    printf("\n");

    int minValue = findMinValue(root);
    printf("The minimum value in the BST is: %d", minValue);

    return 0;
}
```

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

Rajalakshmi Engineering College

Name: srikumaran s
Email: 240801331@rajalakshmi.edu.in
Roll no: 240801331
Phone: 9600017405
Branch: REC
Department: I ECE FD
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 5_COD_Question 5

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

In his computer science class, John is learning about Binary Search Trees (BST). He wants to build a BST and find the maximum value in the tree.

Help him by writing a program to insert nodes into a BST and find the maximum value in the tree.

Input Format

The first line of input consists of an integer N, representing the number of nodes in the BST.

The second line consists of N space-separated integers, representing the values of the nodes to insert into the BST.

Output Format

The output prints the maximum value in the BST.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
10 5 15 2 7

Output: 15

Answer

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

struct TreeNode {
    int data;
    struct TreeNode* left;
    struct TreeNode* right;
};
struct TreeNode* createNode(int key) {
    struct TreeNode* newNode = (struct TreeNode*)malloc(sizeof(struct
TreeNode));
    newNode->data = key;
    newNode->left = newNode->right = NULL;
    return newNode;
}
// You are using GCC
struct TreeNode* insert(struct TreeNode* root, int key) {
    if(root==NULL)
    return createNode(key);
    if(key<root->data)
    root->left=insert(root->left,key);
    else if(key>root->data)
    root->right=insert(root->right,key);
    return root;
}

int findMax(struct TreeNode* root) {
    if(root==NULL)
```

```
    return -1;
    while(root->right!=NULL)
    root=root->right;
    return root->data;
}
int main() {
    int N, rootValue;
    scanf("%d", &N);

    struct TreeNode* root = NULL;

    for (int i = 0; i < N; i++) {
        int key;
        scanf("%d", &key);
        if (i == 0) rootValue = key;
        root = insert(root, key);
    }

    int maxVal = findMax(root);
    if (maxVal != -1) {
        printf("%d", maxVal);
    }

    return 0;
}
```

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

Rajalakshmi Engineering College

Name: srikumaran s
Email: 240801331@rajalakshmi.edu.in
Roll no: 240801331
Phone: 9600017405
Branch: REC
Department: I ECE FD
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 6_COD_Question 1

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

John and Mary are collaborating on a project that involves data analysis. They each have a set of age data, one sorted in ascending order and the other in descending order. However, their analysis requires the data to be in ascending order.

Write a program to help them merge the two sets of age data into a single sorted array in ascending order using merge sort.

Input Format

The first line of input consists of an integer N, representing the number of age values in each dataset.

The second line consists of N space-separated integers, representing the ages of participants in John's dataset (in ascending order).

The third line consists of N space-separated integers, representing the ages of participants in Mary's dataset (in descending order).

Output Format

The output prints a single line containing space-separated integers, which represents the merged dataset of ages sorted in ascending order.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
1 3 5 7 9
10 8 6 4 2
Output: 1 2 3 4 5 6 7 8 9 10

Answer

```
#include <stdio.h>

// You are using GCC
void merge(int arr[], int left[], int right[], int left_size, int right_size) {
    int i=0,j=0,k=0;
    while(i<left_size && j<right_size)
    {
        if(left[i]<right[j])
        arr[k++]=left[i++];
        else
        arr[k++]=right[j++];
    }



    while(i<left_size)
    arr[k++]=left[i++];


    while(j<right_size)
    arr[k++]=right[j++];
```

```
}
void mergeSort(int arr[], int size) {
    if(size<2)
    return ;
    int mid=size/2;
    int left[mid], right[size-mid];
    for(int i=0;i<mid;i++)

    left[i]=arr[i];

    for(int i=mid;i<size;i++)

    right[i-mid]=arr[i];

    mergeSort(left,mid);
    mergeSort(right,size-mid);
    merge(arr,left,right,mid,size-mid);



}
int main() {
    int n, m;
    scanf("%d", &n);
    int arr1[n], arr2[n];
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr1[i]);
    }
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr2[i]);
    }
    int merged[n + n];
    mergeSort(arr1, n);
    mergeSort(arr2, n);
    merge(merged, arr1, arr2, n, n);
    for (int i = 0; i < n + n; i++) {
        printf("%d ", merged[i]);
    }
    return 0;
}
```

Status : Correct *Marks : 10/10*

Rajalakshmi Engineering College

Name: srikumaran s
Email: 240801331@rajalakshmi.edu.in
Roll no: 240801331
Phone: 9600017405
Branch: REC
Department: I ECE FD
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 6_COD_Question 2

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Nandhini asked her students to arrange a set of numbers in ascending order. She asked the students to arrange the elements using insertion sort, which involves taking each element and placing it in its appropriate position within the sorted portion of the array.

Assist them in the task.

Input Format

The first line of input consists of the value of n, representing the number of array elements.

The second line consists of n elements, separated by a space.

Output Format

The output prints the sorted array, separated by a space.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
67 28 92 37 59

Output: 28 37 59 67 92

Answer

```
#include <stdio.h>

// You are using GCC

void insertionSort(int arr[], int n) {
    for(int i=1;i<n;i++){
        int key=arr[i];
        int j=i-1;
        while(j>=0 && arr[j]>key){
            arr[j+1]=arr[j];
            j--;
        }
        arr[j+1]=key;
    }



}

void printArray(int arr[], int n) {
    for(int i=0;i<n;i++){
        printf("%d ",arr[i]);
    }
}

int main() {
    int n;
    scanf("%d", &n);
    int arr[n];
    for (int i = 0; i < n; i++) {
```

```
        scanf("%d", &arr[i]);
    }

    insertionSort(arr, n);
    printArray(arr, n);
    return 0;
}
```

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

Rajalakshmi Engineering College

Name: srikumaran s
Email: 240801331@rajalakshmi.edu.in
Roll no: 240801331
Phone: 9600017405
Branch: REC
Department: I ECE FD
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 6_COD_Question 3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

You are the lead developer of a text-processing application that assists writers in organizing their thoughts. One crucial feature is a character-sorting service that helps users highlight the most critical elements of their text.

To achieve this, you decide to enhance the service to sort characters in descending order using the Quick-Sort algorithm. Implement the algorithm to efficiently rearrange the characters, ensuring that it is sorted in descending order.

Input Format

The first line of the input consists of a positive integer value N, representing the number of characters to be sorted.

The second line of input consists of N space-separated lowercase alphabetical characters.

Output Format

The output displays the set of alphabetical characters, sorted in descending order.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 5
a d g j k
Output: k j g d a

Answer

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

// You are using GCC
void swap(char* a, char* b) {
    char temp=*a;
    *a=*b;
    *b=temp;
}

int partition(char arr[], int low, int high) {
    char pivot=arr[high];
    int i=low-1;
    for(int j=low;j<high;j++){
        if(arr[j]>pivot){
            i++;
            swap(&arr[i],&arr[j]);
        }
    }
    swap(&arr[i+1],&arr[high]);
    return i+1;
}
```

```
void quicksort(char arr[], int low, int high) {
    if(low<high){
        int pi=partition(arr,low,high);
        quicksort(arr,low,pi-1);
        quicksort(arr,pi+1,high);
    }
}

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

    char characters[n];

    for (int i = 0; i < n; i++) {
        char input;
        scanf(" %c", &input);
        characters[i] = input;
    }

    quicksort(characters, 0, n - 1);

    for (int i = 0; i < n; i++) {
        printf("%c ", characters[i]);
    }

    return 0;
}
```

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

Rajalakshmi Engineering College

Name: srikumaran s
Email: 240801331@rajalakshmi.edu.in
Roll no: 240801331
Phone: 9600017405
Branch: REC
Department: I ECE FD
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 6_COD_Question 4

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Kavya, a software developer, is analyzing data trends. She has a list of integers and wants to identify the nth largest number in the list after sorting the array using QuickSort.

To optimize performance, Kavya is required to use QuickSort to sort the list before finding the nth largest number.

Input Format

The first line of input consists of an integer n, representing the size of the array.

The second line consists of n space-separated integers, representing the elements of the array nums.

The third line consists of an integer k, representing the position of the largest

number you need to print after sorting the array.

Output Format

The output prints the k-th largest number in the sorted array (sorted in ascending order).

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 6
-1 0 1 2 -1 -4
3
Output: 0

Answer

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

// You are using GCC
int partition(int arr[], int low, int high) {
    int pivot=arr[high];
    int i=low-1;
    for(int j=low;j<high;j++){
        if(arr[j]<pivot){
            i++;
            int temp=arr[i];
            arr[i]=arr[j];
            arr[j]=temp;
        }
    }
    int temp=arr[i+1];
    arr[i+1]=arr[high];
    arr[high]=temp;
    return i+1;
}

void quickSort(int arr[], int low, int high) {
    if(low<high){
        int pi=partition(arr,low,high);
```

```
        quickSort(arr,low,pi-1);
        quickSort(arr,pi+1,high);
    }
}

void findNthLargest(int* nums, int n, int k) {
    quickSort(nums,0,n-1);
    printf("%d\n",nums[n-k]);
}

int main() {
    int n, k;
    scanf("%d", &n);
    int* nums = (int*)malloc(n * sizeof(int));
    for (int i = 0; i < n; i++) {
        scanf("%d", &nums[i]);
    }
    scanf("%d", &k);
    findNthLargest(nums, n, k);
    free(nums);
    return 0;
}
```

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

Rajalakshmi Engineering College

Name: srikumaran s
Email: 240801331@rajalakshmi.edu.in
Roll no: 240801331
Phone: 9600017405
Branch: REC
Department: I ECE FD
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 6_COD_Question 5

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Jose has an array of N fractional values, represented as double-point numbers. He needs to sort these fractions in increasing order and seeks your help.

Write a program to help Jose sort the array using the merge sort algorithm.

Input Format

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

The second line consists of N double-point numbers, separated by spaces, representing the fractions array.

Output Format

The output prints N double-point numbers, sorted in increasing order, and rounded to three decimal places.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 4
0.123 0.543 0.321 0.789

Output: 0.123 0.321 0.543 0.789

Answer

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

// You are using GCC
int compare(double a, double b) {
   return a>b;
}
void merge(double arr[], int l, int m, int r) {
   int n1=m-l+1, n2=r-m;
   double L[n1],R[n2];
   for(int i=0;i<n1;i++)
     L[i]=arr[i+l];
   for(int i=0;i<n2;i++)
     R[i]=arr[m+1+i];
   int i=0,j=0,k=l;
   while(i<n1 && j<n2){
      if(!compare(L[i],R[j]))
         arr[k++]=L[i++];
      else
         arr[k++]=R[j++];
   }
   while(i<n1)
      arr[k++]=L[i++];
   while(j<n2)
      arr[k++]=R[j++];
}
void mergeSort(double arr[], int l, int r) {
   if(l<r){
```

```
        int m=(l+r)/2;
        mergeSort(arr,l,m);
        mergeSort(arr,m+1,r);\
        merge(arr,l,m,r);
    }
}

int main() {
    int n;
    scanf("%d", &n);
    double fractions[n];
    for (int i = 0; i < n; i++) {
        scanf("%lf", &fractions[i]);
    }
    mergeSort(fractions, 0, n - 1);
    for (int i = 0; i < n; i++) {
        printf("%.3f ", fractions[i]);
    }
    return 0;
}
```

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

Rajalakshmi Engineering College

Name: srikumaran s
Email: 240801331@rajalakshmi.edu.in
Roll no: 240801331
Phone: 9600017405
Branch: REC
Department: I ECE FD
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 7_COD_Question 2

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Priya is developing a simple student management system. She wants to store roll numbers in a hash table using Linear Probing, and later search for specific roll numbers to check if they exist.

Implement a hash table using linear probing with the following operations:

Insert all roll numbers into the hash table.For a list of query roll numbers, print "Value x: Found" or "Value x: Not Found" depending on whether it exists in the table.

Input Format

The first line contains two integers, n and table_size — the number of roll numbers to insert and the size of the hash table.

The second line contains n space-separated integers — the roll numbers to insert.

The third line contains an integer q — the number of queries.

The fourth line contains q space-separated integers — the roll numbers to search for.

Output Format

The output print q lines — for each query value x, print: "Value x: Found" or "Value x: Not Found"

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5 10
21 31 41 51 61
3
31 60 51

Output: Value 31: Found
Value 60: Not Found
Value 51: Found

Answer

```
#include <stdio.h>

#define MAX 100

// You are using GCC
void initializeTable(int table[], int size) {
   for(int i=0;i<size;i++)
   table[i]= -1;
}

int linearProbe(int table[], int size, int num) {
   int index =num%size;
   while(table[index]!=-1)
     index =(index + 1)% size;
    return index;
```

```
}

void insertIntoHashTable(int table[], int size, int arr[], int n) {
    for(int i=0;i<n;i++)
      table[linearProbe(table,size,arr[i])]=arr[i];
}

int searchInHashTable(int table[], int size, int num) {
    int index =num % size;
    while(table[index]!= -1){
        if (table[index] == num)
          return 1;
        index = (index + 1)% size;
        }
        return 0;
}
int main() {
    int n, table_size;
    scanf("%d %d", &n, &table_size);

    int arr[MAX], table[MAX];
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    initializeTable(table, table_size);
    insertIntoHashTable(table, table_size, arr, n);

    int q, x;
    scanf("%d", &q);
    for (int i = 0; i < q; i++) {
        scanf("%d", &x);
        if (searchInHashTable(table, table_size, x))
            printf("Value %d: Found\n", x);
        else
            printf("Value %d: Not Found\n", x);
    }

    return 0;
}
```

Status : Correct *Marks : 10/10*

Rajalakshmi Engineering College

Name: srikumaran s
Email: 240801331@rajalakshmi.edu.in
Roll no: 240801331
Phone: 9600017405
Branch: REC
Department: I ECE FD
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 7_COD_Question 3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

In a messaging application, users maintain a contact list with names and corresponding phone numbers. Develop a program to manage this contact list using a dictionary implemented with hashing.

The program allows users to add contacts, delete contacts, and check if a specific contact exists. Additionally, it provides an option to print the contact list in the order of insertion.

Input Format

The first line consists of an integer n, representing the number of contact pairs to be inserted.

Each of the next n lines consists of two strings separated by a space: the name of the contact (key) and the corresponding phone number (value).

The last line contains a string k, representing the contact to be checked or removed.

Output Format

If the given contact exists in the dictionary:

1. The first line prints "The given key is removed!" after removing it.
2. The next n - 1 lines print the updated contact list in the format: "Key: X; Value: Y" where X represents the contact's name and Y represents the phone number.

If the given contact does not exist in the dictionary:

1. The first line prints "The given key is not found!".
2. The next n lines print the original contact list in the format: "Key: X; Value: Y" where X represents the contact's name and Y represents the phone number.

Refer to the sample outputs for the formatting specifications.

Sample Test Case

Input: 3
Alice 1234567890
Bob 9876543210
Charlie 4567890123
Bob

Output: The given key is removed!
Key: Alice; Value: 1234567890
Key: Charlie; Value: 4567890123

Answer

```
// You are using GCC
#include<stdio.h>
#include<string.h>
#define MAX 50

typedef struct{
```

```
    char name[11];
    char phone[11];
}contact;
void removecontact(contact contacts[],int*n,char *key){
    int found = -1;
    for(int i=0;i< *n;i++){
        if(strcmp(contacts[i].name,key) ==0) {
            found = i;
            break;
        }
    }
    if (found != -1){
        printf("the given key is removed!\n");
        for(int i = found;i< *n - 1;i++){
            contacts[i] = contacts[i + 1];
        }
        (*n)--;
    }
    else{
        printf("the given key is not found!\n");
    }
}

void printcontacts(contact contacts[],int n){
    for(int i=0;i<n;i++){
        printf("key:%s;value:%s\n",contacts[i].name,contacts[i].phone);
    }
}
int main(){
    int n;
    scanf("%d",&n);
    contact contacts[MAX];
    for(int i=0;i<n;i++){
        scanf("%s %s",contacts[i].name,contacts[i].phone);
    }
    char key[11];
    scanf("%s",key);
    removecontact(contacts,&n,key);
    printcontacts(contacts,n);
    return 0;
}
```

Status : Correct *Marks : 10/10*

Rajalakshmi Engineering College

Name: srikumaran s
Email: 240801331@rajalakshmi.edu.in
Roll no: 240801331
Phone: 9600017405
Branch: REC
Department: I ECE FD
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 7_COD_Question 4

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Develop a program using hashing to manage a fruit contest where each fruit is assigned a unique name and a corresponding score. The program should allow the organizer to input the number of fruits and their names with scores.

Then, it should enable them to check if a specific fruit, identified by its name, is part of the contest. If the fruit is registered, the program should display its score; otherwise, it should indicate that it is not included in the contest.

Input Format

The first line consists of an integer N, representing the number of fruits in the contest.

The following N lines contain a string K and an integer V, separated by a space, representing the name and score of each fruit in the contest.

The last line consists of a string T, representing the name of the fruit to search for.

Output Format

If T exists in the dictionary, print "Key "T" exists in the dictionary.".

If T does not exist in the dictionary, print "Key "T" does not exist in the dictionary.".

Refer to the sample outputs for the formatting specifications.

Sample Test Case

Input: 2
banana 2
apple 1
Banana

Output: Key "Banana" does not exist in the dictionary.

Answer

```
// You are using GCC
#include<stdio.h>
#include<string.h>
#define MAX 15

typedef struct{
   char name[20];
   int score;
}fruit;
int searchfruit(fruit fruits[],int n,char *key){
   for(int i=0;i<n;i++){
      if(strcmp(fruits[i].name,key) == 0){
         printf("key \"%s\" exists in the dictionary.\n",key);
         return 1;
      }
   }
```

```
    printf("key \"%s\" does not exist in the dictionary.\n",key);
    return 0;
}
int main() {
    int n;
    scanf("%d",&n);
    fruit fruits[MAX];

    for(int i=0;i<n;i++){
        scanf("%s %d",fruits[i].name,&fruits[i].score);
    }
    char key[20];
    scanf("%s",key);
    searchfruit(fruits,n,key);
    return 0;
}
```

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

Rajalakshmi Engineering College

Name: srikumaran s
Email: 240801331@rajalakshmi.edu.in
Roll no: 240801331
Phone: 9600017405
Branch: REC
Department: I ECE FD
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 7_COD_Question 5

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

You are provided with a collection of numbers, each represented by an array of integers. However, there's a unique scenario: within this array, one element occurs an odd number of times, while all other elements occur an even number of times. Your objective is to identify and return the element that occurs an odd number of times in this arrangement.

Utilize mid-square hashing by squaring elements and extracting middle digits for hash codes. Implement a hash table for efficient integer occurrence tracking.

Note: Hash function: squared = key * key.

Example

Input:

7

2 2 3 3 4 4 5

Output:

5

Explanation

The hash function and the calculated hash indices for each element are as follows:

2 -> hash(2*2) % 100 = 4

3 -> hash(3*3) % 100 = 9

4 -> hash(4*4) % 100 = 16

5 -> hash(5*5) % 100 = 25

The hash table records the occurrence of each element's hash index:

Index 4: 2 occurrences

Index 9: 2 occurrences

Index 16: 2 occurrences

Index 25: 1 occurrence

Among the elements, the integer 5 occurs an odd number of times (1 occurrence) and satisfies the condition of the problem. Therefore, the program outputs 5.

Input Format

The first line of input consists of an integer N, representing the size of the array.

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

Output Format

The output prints a single integer representing the element that occurs an odd

number of times.

If no such element exists, print -1.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 7
2 2 3 3 4 4 5

Output: 5

Answer

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

#define MAX_SIZE 100

// You are using GCC
unsigned int hash(int key, int tableSize) {
    return(key*key)%tableSize;
}

int getOddOccurrence(int arr[], int size) {
    int hashTable[MAX_SIZE] = {0};
    for(int i=0;i<size;i++){
        hashTable[hash(arr[i],MAX_SIZE)]++;
    }
    for(int i=0;i<size;i++){
        if(hashTable[hash(arr[i],MAX_SIZE)]%2 !=0){
            return arr[i];
        }
    }
    return -1;
}

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

```
    int arr[MAX_SIZE];
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }

    printf("%d\n", getOddOccurrence(arr, n));

    return 0;
}
```

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