RAJALAKSHMI ENGINEERING COLLEGE

An Autonomous Institution
Affiliated to Anna University, Chennai,
Rajalakshmi Nagar, Thandalam – 602 105

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CS23231- DATA STRUCTURES

Laboratory Record Note Book

Name : …………………………………………………………………………………………………

Register No. : ……………………………………………………………………………………..

Year / Branch / Section : ……………………………………………………………………...

Semester : ……………………………………………………………………………………..

AcademicYear: ………………………………………………………………………………...

RAJALAKSHMI ENGINEERING COLLEGE

An Autonomous Institution
Affiliated to Anna University, Chennai,
Rajalakshmi Nagar, Thandalam – 602 105

BONAFIDE CERTIFICATE

Name: ……………………………………………………………………………

Academic Year: …………… Semester: …………… Branch: ………………

Register No.

Certified that this is the bonafide record of work done by the above student in the.. Laboratory during the academic year 2023- 2024

Signature of Faculty in-charge

Submitted for the Practical Examination held on……………………………

Internal Examiner **External Examiner**

Rajalakshmi Engineering College

Name: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

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

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 n;
   scanf("%d",&n);
   int arr[n],h=0;
   float avg=0;
   for(int i=0;i<n;i++){
      scanf("%d",&arr[i]);
   }
   int l=arr[0];
   for(int i=0;i<n;i++){
      printf("Employee %d: %d\n",i+1,arr[i]);
      avg +=arr[i];
      if(h<arr[i]){
         h=arr[i];
      }
      if(l>arr[i]){
         l = arr[i];
      }
   }
   printf("\nAverage Salary: %.2f\n",avg/n);
   printf("Highest Salary: %d\n",h);
   printf("Lowest Salary: %d",l);
}
```

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

2. 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;
   scanf("%d",&n);
   scanf("%d",&m);
   int matrix[n][m];
   for(int i=0;i<n;i++){
      for(int j=0;j<m;j++){
         scanf("%d",&matrix[i][j]);
      }
   }
   if(matrix[n-1][m-1]%2 ==0){
      printf("%d is even",matrix[n-1][m-1]);
   }
   else{
      printf("%d is odd",matrix[n-1][m-1]);
   }
}
```

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 n;
   scanf("%d",&n);
   int arr[n];
   for(int i=0;i<n;i++){
      scanf("%d",&arr[i]);
   }
   int ch=0;
   while(ch !=5){
      scanf("%d",&ch);
      int s=arr[0];
      if(ch == 1){
         for(int i=0;i<n;i++){
            if(s>arr[i]){
               s = arr[i];
            }
         }
         printf("The smallest number is: %d\n",s);
      }
      else if(ch == 2){
         for(int i=0;i<n;i++){
            if(s<arr[i]){
               s = arr[i];
            }
         }
         printf("The largest number is: %d\n",s);
      }
      else if(ch == 3){
         s=0;
         for(int i=0;i<n;i++){
            s+=arr[i];
         }
         printf("The sum of the numbers is: %d\n",s);
      }
      else if(ch == 4){
            float f =0;
            for(int i=0;i<n;i++){
                f+=arr[i];
            }
            printf("The average of the numbers is: %.2f\n",f/n);
        }
        else if(ch!=5){
            printf("Invalid choice!\n");
        }
    }
    printf("Exiting the program");
}
```

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

4. 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>


void setZeros(int arr[10][10], int n) {
   for(int i=0;i<n;i++){
      for(int j=0;j<=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***

5. 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 sum =0;
    for(int i=0;i<rows;i++){
        for(int j=0;j<cols;j++){
            sum+= matrix[i][j];
        }
        printf("%d ",sum);
        sum =0;
    }
}

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***

Rajalakshmi Engineering College

Name: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 0_Pointers

Attempt : 1
Total Mark : 5
Marks Obtained : 5

Section 1 : Coding

1. Problem Statement

Raj wants to create a program using pointers and a structure named Employee to manage employee information.

He seeks your assistance to input the employee's name, salary, and hours worked. Implement a salary increase based on hours worked, and calculate the final salary. Calculate the total salary for 30 days. Display the results of the final and total salary.

Salary increase criteria:

If hours worked >= 12, the increase is Rs. 150.00.If hours worked >= 10, but less than 12, the increase is Rs. 100.00.If hours worked >= 8, but less than 10, the increase is Rs. 50.00.If hours worked < 8, there is no increase.

Input Format

The first line of input consists of a string, representing the Employee's name.

The second line consists of a double-point number, representing the Employee's current salary.

The third line consists of an integer, representing the number of hours worked by the employee.

Output Format

The first line of output prints "Final Salary: Rs. " followed by a double value, representing the final salary, rounded off to two decimal places.

The second line prints "Total Salary: Rs. " followed by a double value, representing the total salary for 30 days, rounded off to two decimal places.

Refer to the sample outputs for formatting specifications.

Sample Test Case

Input: Akil
3000.00
6

Output: Final Salary: Rs. 3000.00
Total Salary: Rs. 90000.00

Answer

```
#include<stdio.h>
#include<string.h>
struct Employee{
   char name[100];
   float sal;
   int hworked;
};
int main()
{
   char n[100];
   float salary;
   int h;
   fgets(n,sizeof(n),stdin);
   scanf("%f",&salary);
```

```
    scanf("%d",&h);
    struct Employee a;
    a.sal = salary;
    strcpy(n,a.name);
    a.hworked = h;
    if(h>=12){
        a.sal+=150;
    }
    else if(h>=10){
        a.sal+=100;
    }
    else if(h>=8){
        a.sal+=50;
    }
    printf("Final salary: Rs. %.2f\n",a.sal);
    printf("Total salary: Rs. %.2f",a.sal*30);

}
```

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

2. Problem Statement

Ria is a mathematician who loves exploring combinatorics. She is working on a project that involves calculating permutations.

Ria wants to create a program that takes the values of n and r as input and calculates the permutations of n elements taken r at a time.

Write a program using pointers and a function calculatePermutations that, given the values of n and r, calculates and prints the permutations of n elements taken r at a time.

Permutation: n! / (n - r)!

Input Format

The first line consists of an integer n, representing the total number of elements.

The second line consists of an integer r, representing the number of elements to be taken at a time.

Output Format

The output prints the result of the permutation.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 4
3
Output: 24

Answer

```
#include<stdio.h>
int permu(int *n,int *r){
   long long int result=1;
   for(int i=0;i<*r;i++){
      result*=(*n-i);
   }
   return result;
}
int main(){
   int n,r;
   scanf("%d",&n);
   scanf("%d",&r);
   if(n<r || n<0 || r<0)
      return 1;
   printf("%lld\n",permu(&n,&r));
}
```

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

3. Problem Statement

Sam is developing a program for analyzing daily temperature fluctuations. Users input the number of days, followed by daily temperature values whose memory is allocated using malloc.

The program calculates and displays the following:

The absolute temperature changes between consecutive days (The first value remains the same).The average temperature of adjacent days.

This allows users to gain insights into daily temperature variations for better analysis.

For Example,

Let us assume the temperature for 3 days as 25.5, 28.0, and 23.5.

The absolute differences:

Day 1: (N/A, as there is no previous day) = 25.50Day 2: abs(28.0 - 25.5) = 2.50Day 2: abs(23.5 - 28.0) = 4.50

The average temperatures:

Day 1: (N/A, as there is no previous day) = 25.50Day 2: (25.5 + 23.5) / 2.0 = 24.50Day 3: (N/A, as there is no next day) = 23.50

Input Format

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

The second line consists of N space-separated float values, representing the temperature values for N days.

Output Format

The first line displays the absolute temperature change for N days as float values, rounded to two decimal places, separated by a space.

The second line displays the average temperature for N days as float values, rounded to two decimal places, separated by a space.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 3
25.5 28.0 23.5

```
Output: 25.50 2.50 4.50
25.50 24.50 23.50
```

Answer

```
#include<stdio.h>
#include<stdlib.h>
#include<math.h>
int main(){
   int n;
   float *arr=(float*)malloc(n*sizeof(float));
   scanf("%d",&n);
   for(int i=0;i<n;i++){
      scanf("%f",&arr[i]);
   }
   printf("%.2f ",arr[0]);
   for(int i=1;i<n;i++){
      printf("%.2f ",fabs(arr[i]-arr[i-1]));
   }
   printf("\n");
   float s;
   for(int i=0;i<n;i++){
      if(i==0){
         printf("%.2f ",arr[0]);
         s = arr[i];
      }
      else if(i==n-1){
         printf("%.2f ",arr[i]);
      }
      else{
         printf("%.2f ",(s+arr[i+1])/2);
         s= (s+arr[i+1])/2;
      }
   }
}
```

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

4. Problem Statement

Daniel is working on a project that involves analyzing data stored in float arrays. He needs to determine whether a given float array contains only positive numbers.

To achieve this, he needs a program that can accurately evaluate the contents of float arrays using malloc().

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 float values, representing the elements of the array.

Output Format

If all the array elements are positive, print "All elements are positive."

If the array contains at least one positive element, print "At least one element is positive."

If there are no positive elements in the array, print "No positive elements in the array."

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
50.0 -2.3 3.7 -4.8 5.2
Output: At least one element is positive.

Answer

```
#include<stdio.h>
#include<stdlib.h>
int main(){
   int n,t=1,c=0;
   scanf("%d",&n);
   float *p;
   p = (float*) malloc(n*sizeof(p));
   for(int i=0;i<n;i++){
      scanf("%f",&p[i]);
      if(p[i]<0){
            t=0;
        }
        else{
            c++;
        }
    }
    if(t==0 && c==0 ){
        printf("No positive elements in the array.");
    }
    else if( t==1){
        printf("All elements are positive.");
    }
    else {
        printf("At least one element is positive.");
    }
}
```

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

5. Problem Statement

Rajwinder wants a program to determine retirement details for a person based on their age.

Create a program that uses a structure called Person to hold the age as an attribute with a pointer.

If the age is under 18, display "Invalid".If the age is 65 or older, print "Already retired!".Otherwise, calculate and output the retirement year, remaining years, and remaining days until retirement.

Note: Age 65 is considered as retirement age. Assume the current year as 2023 and there are 365 days per year for calculation.

Input Format

The input consists of an integer representing the person's age.

Output Format

If the age is under 18, the output displays "Invalid" and terminates.

If the age is 65 or older, the output displays "Already retired!" and terminates.

Otherwise, the output displays the following.

1. The first line displays "Retirement Year: " followed by an integer representing the retirement year.
2. The second line displays "Remaining Years: " followed by an integer representing the remaining years left for retirement.
3. The third line displays "Remaining Days: " followed by an integer representing the remaining days left for retirement.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 43

Output: Retirement Year: 2045
Remaining Years: 22
Remaining Days: 8030

Answer

```
#include<stdio.h>
struct Person{
   int age;
};
int main(){
   struct Person a;
   scanf("%d",&a.age);
   if(a.age<18){
      printf("Invalid");
   }
   else if(a.age>65){
      printf("Already retired!");
   }
   else{
      printf("Retirement Year: %d",2023+(65-a.age));
      printf("Remaining Years: %d",(65-a.age));
      printf("Remaining Days: %d",(65-a.age)*365);
   }
}
```

Status : Correct *Marks : 1/1*

Rajalakshmi Engineering College

Name: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 1_MCQ

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : MCQ

1. The following function takes a singly linked list of integers as a parameter and rearranges the elements of the lists.

The function is called with the list containing the integers 1, 2, 3, 4, 5, 6, 7 in the given order. What will be the contents of the list after the function completes execution?

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

void rearrange (struct node* list) {
   struct node *p,q;
   int temp;
   if (! List || ! list->next) return;
```

```
  p=list; q=list->next;
  while(q) {
    temp=p->value; p->value=q->value;
    q->value=temp;p=q->next;
    q=p?p->next:0;
  }
}
```

Answer

2, 1, 4, 3, 6, 5, 7

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

2. Consider the singly linked list: 15 -> 16 -> 6 -> 7 -> 17. You need to delete all nodes from the list which are prime.

What will be the final linked list after the deletion?

Answer

15 -&gt; 16 -&gt; 6

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

3. Given a pointer to a node X in a singly linked list. If only one point is given and a pointer to the head node is not given, can we delete node X from the given linked list?

Answer

Possible if X is not last node.

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

4. The following function reverse() is supposed to reverse a singly linked list. There is one line missing at the end of the function.

What should be added in place of "/*ADD A STATEMENT HERE*/", so that the function correctly reverses a linked list?

```
struct node {
   int data;
   struct node* next;
};
static void reverse(struct node** head_ref) {
   struct node* prev   = NULL;
   struct node* current = *head_ref;
   struct node* next;
   while (current != NULL) {
      next  = current->next;
      current->next = prev;
      prev = current;
      current = next;
   }
   /*ADD A STATEMENT HERE*/
}
```

Answer

*head_ref = prev;

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

5. Which of the following statements is used to create a new node in a singly linked list?

```
struct node {
   int data;
   struct node * next;
}
typedef struct node NODE;
NODE *ptr;
```

Answer

ptr = (NODE*)malloc(sizeof(NODE));

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

6. Consider the singly linked list: 13 -> 4 -> 16 -> 9 -> 22 -> 45 -> 5 -> 16 ->

6, and an integer K = 10, you need to delete all nodes from the list that are less than the given integer K.

What will be the final linked list after the deletion?

Answer

13 -&gt; 16 -&gt; 22 -&gt; 45 -&gt; 16

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

7. Linked lists are not suitable for the implementation of?

Answer

Binary search

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

8. In a singly linked list, what is the role of the "tail" node?

Answer

It stores the last element of the list

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

9. Given the linked list: 5 -> 10 -> 15 -> 20 -> 25 -> NULL. What will be the output of traversing the list and printing each node's data?

Answer

5 10 15 20 25

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

10. Consider an implementation of an unsorted singly linked list. Suppose it has its representation with a head pointer only. Given the representation, which of the following operations can be implemented in O(1) time?

i) Insertion at the front of the linked list

ii) Insertion at the end of the linked list

iii) Deletion of the front node of the linked list

iv) Deletion of the last node of the linked list

Answer

I and III

Status : Correct *Marks : 1/1*

Rajalakshmi Engineering College

Name: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 1_COD_Question 1

Attempt : 3
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 expo;
   struct poly*next;
}node;
node*newnode(int coeff,int expon){
   node* newnode = (node*)malloc(sizeof(node));
   newnode->coeff = coeff;
   newnode->expo=expon;
   newnode->next=NULL;
   return newnode;
}
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: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

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 = 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: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 1_COD_Question 3

Attempt : 2
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>
int main(){
   int n;
   scanf("%d",&n);
   char *ptr;
   ptr =(char*)malloc((n+1)*sizeof(char));
   for(int i=0;i<n;i++){
      scanf(" %c",(ptr+i));
   }
   int pos;
   char letter;
```

```
    scanf("%d",&pos);
    scanf(" %c",&letter);
    if(pos<n){
        for(int i=n;i>pos;i--){
            *(ptr+i)=*(ptr+(i-1));
        }
        *(ptr+(pos+1)) = letter;
        n++;
        printf("Updated list: ");
        int i;
        for(i=0;i<n;i++){
            printf("%c ",*(ptr+i));
        }
        printf("\n");
    }
    else{
        printf("Invalid index\n");
        printf("Updated List: ");
        for(int i=0;i<n;i++){
            printf("%c ",*(ptr+i));
        }
    }
}
```

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

Rajalakshmi Engineering College

Name: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

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: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

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 : 10

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

```
#include<stdio.h>
#include<stdlib.h>
typedef struct Node{
   float gpa;
   struct Node* next;
}node;
node* createnode(float gpa){
   node* newnode = (node*)malloc(sizeof(node));
   newnode->gpa=gpa;
   newnode->next = NULL;
   return newnode;
}
void insertbg(node** head,float gpa){
   node*newnode = createnode(gpa);
   newnode->next = *head;
   *head=newnode;
}
void deletep(node** head,int pos){
   if(*head==NULL)return;
   node* temp=*head;
```

```
    if(pos==1){
        *head=temp->next;
        free(temp);
        return;
    }
    for(int i=1;temp!=NULL && i<pos-1;i++){
        temp=temp->next;
    }
    if(temp==NULL||temp->next==NULL)return;
    node*next=temp->next->next;
    free(temp->next);
    temp->next=next;
}
void print(node*head){
    while(head!=NULL){
        printf("GPA: %.1f\n",head->gpa);
        head=head->next;
    }
}
int main(){
    int n;
    scanf("%d",&n);
    node*head = NULL;
    float gpa;
    for(int i=0;i<n;i++){
        scanf("%f",&gpa);
        insertbg(&head,gpa);
    }
    int pos;
    scanf("%d",&pos);
    deletep(&head,pos);
    print(head);
    return 0;
}
```

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

Rajalakshmi Engineering College

Name: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 1_COD_Question 6

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

John is tasked with creating a program to manage student roll numbers using a singly linked list.

Write a program for John that accepts students' roll numbers, inserts them at the end of the linked list, and displays the numbers.

Input Format

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

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

Output Format

The output prints the space-separated integers singly linked list, after inserting the roll numbers of students at the end.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
23 85 47 62 31

Output: 23 85 47 62 31

Answer

```
#include<stdio.h>
#include<stdlib.h>
typedef struct student{
   int roll;
   struct student* next;
}node;
node* newnode(int rollno){
   node* data = (node*)malloc(sizeof(data));
   data->roll = rollno;
   data->next= NULL;
   return data;
}
void traverse(node* head){
   while(head!=NULL){
      printf("%d ",head->roll);
      head=head->next;
   }
}
int main(){
   int n,rollno;
   scanf("%d",&n);
   scanf("%d",&rollno);
   node* head = newnode(rollno);
   node* temp = head;
   while(--n){
      scanf("%d",&rollno);
      temp->next = newnode(rollno);
      temp=temp->next;
```

```
    }
    traverse(head);
}
```

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

Rajalakshmi Engineering College

Name: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 1_COD_Question 7

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Dev is tasked with creating a program that efficiently finds the middle element of a linked list. The program should take user input to populate the linked list by inserting each element into the front of the list and then determining the middle element.

Assist Dev, as he needs to ensure that the middle element is accurately identified from the constructed singly linked list:

If it's an odd-length linked list, return the middle element.If it's an even-length linked list, return the second middle element of the two elements.

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 integers, representing the elements of the list.

Output Format

The first line of output displays the linked list after inserting elements at the front.

The second line displays "Middle Element: " followed by the middle element of the linked list.

Refer to the sample output for formatting specifications.

Sample Test Case

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

Answer

```
#include <stdio.h>
#include <stdlib.h>
struct Node {
   int data;
   struct Node* next;
};


struct Node* push(Node* head,int value){
   Node* newnode= (struct Node*)malloc(sizeof(struct Node*));
   newnode->next = head;
   newnode->data = value;
   return newnode;
}
int printMiddle(struct Node* head){
   int len=0;
   Node* temp=head;
   while(temp!=NULL){
      temp=temp->next;
```

```
        len++;
    }
    int pos=len/2;
    for(int i=0;i<pos;i++){
        head = head->next;
    }
    return head->data;
}


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

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

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

    struct Node* current = head;
    while (current != NULL) {
        printf("%d ", current->data);
        current = current->next;
    }
    printf("\n");


    int middle_element = printMiddle(head);
    printf("Middle Element: %d\n", middle_element);


    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: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_week 1_CY

Attempt : 1
Total Mark : 30
Marks Obtained : 13

Section 1 : Coding

1. Problem Statement

Hasini is studying polynomials in her class. Her teacher has introduced a new concept of two polynomials using linked lists.

The teacher provides Hasini with a program that takes two polynomials as input, represented as linked lists, and then displays them together. The polynomials are simplified and should be displayed in the format ax^b, where a is the coefficient and b is the exponent.

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 first line of output prints the first polynomial.

The second line of output prints the second polynomial.

The polynomials should be displayed in the format ax^b, where a is the coefficient and b is the exponent.

Refer to the sample output for formatting specifications.

Sample Test Case

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

```
Output: 1x^2 + 2x + 3
2x^2 + 1x + 4
```

Answer

```
#include<stdio.h>
#include<stdlib.h>
typedef struct Node{
   int n,m;
   struct Node *next;
}node;
node *temp;
void create(node **head,int x,int y){
   node *newnode;
```

```
    newnode = (node*) malloc(sizeof(node));
    newnode->n = x;
    newnode->m = y;
    newnode->next=NULL;
    if(*head == NULL){
       *head = temp = newnode;
    }
    else{
       temp->next = newnode;
       temp = newnode;
       temp->next = NULL;
    }
 }
 int main(){
    node *head=NULL;
    int n,x,y;
    scanf("%d",&n);
    for(int i=0;i<n;i++){
       scanf("%d %d\n",&x,&y);
       create(&head,x,y);
    }
    temp = head;
    while(temp!=NULL){
       switch(temp->m){
          case 0:
             printf(" + %d",temp->n);
             break;
          case 1:
             printf(" + %dx",temp->n);
             break;
          case 2:
             printf("%dx^2",temp->n);
       }
       temp=temp->next;
    }

    scanf("%d",&n);
    free(head);
    head = NULL;
    for(int i=0;i<n;i++){
       scanf("%d %d\n",&x,&y);
       create(&head,x,y);
    }
    temp = head;
    while(temp!=NULL){
        switch(temp->m){
            case 0:
                printf(" + %d",temp->n);
                break;
            case 1:
                printf(" + %dx",temp->n);
                break;
            case 2:
                printf("%dx^2",temp->n);
        }
        temp=temp->next;
    }
}
```

Status : Partially correct ***Marks : 3/10***

2. Problem Statement

John is working on a math processing application, and his task is to simplify polynomials entered by users. The polynomial is represented as a linked list, where each node contains two properties:

Coefficient of the term.

Exponent of the term.

John's goal is to combine all the terms that have the same exponent, effectively simplifying the polynomial.

Input Format

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

The next n lines of input consist of two integers, representing the coefficient and exponent of the polynomial in each line separated by space.

Output Format

The first line of output prints the original polynomial in the format 'cx^e + cx^e

+ ...' (where c is the coefficient and e is the exponent of each term).

The second line of output displays the simplified polynomial in the same format as the original polynomial.

If the polynomial is 0, then only '0' will be printed.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 3
5 2
3 1
6 2
Output: Original polynomial: 5x^2 + 3x^1 + 6x^2
Simplified polynomial: 11x^2 + 3x^1

Answer

// You are using Java

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

3. Problem Statement

Akila is a tech enthusiast and wants to write a program to add two polynomials. Each polynomial is represented as a linked list, where each node in the list represents a term in the polynomial.

A term in the polynomial is represented in the format ax^b, where a is the coefficient and b is the exponent.

Akila needs your help to implement a program that takes two polynomials as input, adds them, and stores the result in ascending order in a new polynomial-linked list. Write a program to help her.

Input Format

The input consists of lines containing pairs of integers representing the coefficients and exponents of polynomial terms.

Each line represents a single term, with the coefficient and exponent separated by a space.

The input for each polynomial ends with a line containing "0 0".

Output Format

The output consists of three lines representing the first, second, and resulting polynomial after the addition operation, with terms sorted in ascending order of exponents.

Each line contains terms of the polynomial in the format "coefficientx^exponent", separated by " + ".

If the resulting polynomial is zero, the output is "0".

Refer to the sample output for the formatting specifications.

Sample Test Case

```
Input: 3 4
2 3
1 2
0 0
1 2
2 3
3 4
0 0
Output: 1x^2 + 2x^3 + 3x^4
1x^2 + 2x^3 + 3x^4
2x^2 + 4x^3 + 6x^4
```

Answer

```
#include<stdio.h>
#include<stdlib.h>
typedef struct term{
   int coeff;
   int expo;
```

```
    struct term* next;
}node;

node* createnode(int coeff,int expo){
    node*newnode = (node*)malloc(sizeof(node));
    newnode->coeff = coeff;
    newnode->expo= expo;
    newnode->next = NULL;
    return newnode;
}
void insert(node **head,int coeff,int expo){
    if(coeff ==0) return;
    node* newnode = createnode(coeff,expo);
    if(*head==NULL||expo<(*head)->expo){
        newnode->next = *head;
        *head=newnode;
        return;
    }
    node* curr = *head;
    node* prev = NULL;
    while(curr!=NULL && curr->expo<expo){
        prev = curr;
        curr = curr->next;
    }
    if(curr!=NULL && curr->expo == expo){
        curr->coeff +=coeff;
        if(curr->coeff==0){
            if(prev == NULL){
                *head = curr->next;
            }
            else{
                prev->next = curr->next;
            }
            free(curr);
        }
        free(newnode);
    }
    else{
        newnode ->next = curr;
        if(prev!=NULL){
            prev->next =newnode;
        }
    }
}
node * readpolly(){
    int coeff,expo;
    node* head = NULL;
    while(1){
        scanf("%d %d",&coeff,&expo);
        if(coeff ==0 &&expo==0)break;
        insert(&head,coeff,expo);
    }
    return head;
}
void print(node* head){
    if(head == NULL){
        printf("0\n");
        return;
    }
    while(head){
        printf("%dx^%d",head->coeff,head->expo);
        if(head->next)printf(" + ");
        head = head->next;
    }
    printf("\n");
}
node* add(node *p1,node*p2){
    node* result = NULL;
    while(p1){
        insert(&result,p1->coeff,p1->expo);
        p1=p1->next;
    }
    while(p2){
        insert(&result,p2->coeff,p2->expo);
        p2=p2->next;
    }
    return result;
}

int main(){
    node * poly1 = readpolly();
    node* poly2 = readpolly();
    node* sum = add(poly1,poly2);
    print(poly1);
    print(poly2);
    print(sum);
}
```

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

Rajalakshmi Engineering College

Name: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 2_MCQ_Updated

Attempt : 1
Total Mark : 20
Marks Obtained : 13

Section 1 : MCQ

1. What happens if we insert a node at the beginning of a doubly linked list?

Answer

The previous pointer of the new node is NULL

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

2. Which of the following information is stored in a doubly-linked list's nodes?

Answer

All of the mentioned options

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

3. What will be the output of the following code?

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

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

int main() {
   struct Node* head = NULL;
   struct Node* temp = (struct Node*)malloc(sizeof(struct Node));
   temp->data = 2;
   temp->next = NULL;
   temp->prev = NULL;
   head = temp;
   printf("%d\n", head->data);
   free(temp);
   return 0;
}
```

Answer

2

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

4. What is the correct way to add a node at the beginning of a doubly linked list?

Answer

void addFirst(int data){   Node* newNode = new Node(data);   newNode-&gt;next = head;   if (head != NULL) {   head-&gt;prev = newNode;   }   head = newNode;   }

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

5. What does the following code snippet do?

```
struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
newNode->data = value;
newNode->next = NULL;
newNode->prev = NULL;
```

Answer

Creates a new node and initializes its data to 'value'

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

6. What is the main advantage of a two-way linked list over a one-way linked list?

Answer

Two-way linked lists allow for traversal in both directions.

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

7. Where Fwd and Bwd represent forward and backward links to the adjacent elements of the list. Which of the following segments of code deletes the node pointed to by X from the doubly linked list, if it is assumed that X points to neither the first nor the last node of the list?

A doubly linked list is declared as

```
struct Node {
    int Value;
    struct Node *Fwd;
    struct Node *Bwd;
);
```

Answer

X-&gt;Bwd-&gt;Fwd = X-&gt;Bwd ; X-&gt;Fwd-&gt;Bwd = X-&gt;Fwd;

Status : Wrong ***Marks : 0/1***

8. How do you delete a node from the middle of a doubly linked list?

Answer

All of the mentioned options

Status : Correct *Marks : 1/1*

9. How do you reverse a doubly linked list?

Answer

By swapping the next and previous pointers of each node

Status : Correct *Marks : 1/1*

10. Consider the following function that refers to the head of a Doubly Linked List as the parameter. Assume that a node of a doubly linked list has the previous pointer as prev and the next pointer as next.

Assume that the reference of the head of the following doubly linked list is passed to the below function 1 <--> 2 <--> 3 <--> 4 <--> 5 <-->6. What should be the modified linked list after the function call?

```
Procedure fun(head_ref: Pointer to Pointer of node)
   temp = NULL
   current = *head_ref

   While current is not NULL
      temp = current->prev
      current->prev = current->next
      current->next = temp
      current = current->prev
   End While

   If temp is not NULL
      *head_ref = temp->prev
   End If
End Procedure
```

Answer

5 &lt;--&gt; 4 &lt;--&gt; 3 &lt;--&gt; 2 &lt;--&gt; 1 &lt;--&gt;6.

Status : Wrong ***Marks : 0/1***

11. What is a memory-efficient double-linked list?

Answer

Each node has only one pointer to traverse the list back and forth

Status : Wrong ***Marks : 0/1***

12. Which of the following is true about the last node in a doubly linked list?

Answer

Its next pointer is NULL

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

13. Which pointer helps in traversing a doubly linked list in reverse order?

Answer

tail

Status : Wrong ***Marks : 0/1***

14. What will be the effect of setting the prev pointer of a node to NULL in a doubly linked list?

Answer

The node will become the new tail

Status : Wrong ***Marks : 0/1***

15. Which of the following statements correctly creates a new node for a doubly linked list?

Answer

```
struct Node* newNode = (struct Node*) malloc(sizeof(struct Node));
```

Status : Correct　　　　　　　　　　　　　　　　　　　　　　*Marks : 1/1*

16. What will be the output of the following program?

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

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

int main() {
   struct Node* head = NULL;
   struct Node* tail = NULL;
   for (int i = 0; i < 5; i++) {
      struct Node* temp = (struct Node*)malloc(sizeof(struct Node));
      temp->data = i + 1;
      temp->prev = tail;
      temp->next = NULL;
      if (tail != NULL) {
         tail->next = temp;
      } else {
         head = temp;
      }
      tail = temp;
   }
   struct Node* current = head;
   while (current != NULL) {
      printf("%d ", current->data);
      current = current->next;
   }
   return 0;
}
```

Answer

1 2 3 4 5

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

17. Which of the following is false about a doubly linked list?

Answer

Implementing a doubly linked list is easier than singly linked list

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

18. Which code snippet correctly deletes a node with a given value from a doubly linked list?

```
void deleteNode(Node** head_ref, Node* del_node) {
   if (*head_ref == NULL || del_node == NULL) {
      return;
   }
   if (*head_ref == del_node) {
      *head_ref = del_node->next;
   }
   if (del_node->next != NULL) {
      del_node->next->prev = del_node->prev;
   }
   if (del_node->prev != NULL) {
      del_node->prev->next = del_node->next;
   }
   free(del_node);
}
```

Answer

Deletes the node at a given position in a doubly linked list.

Status : Wrong ***Marks : 0/1***

19. How many pointers does a node in a doubly linked list have?

Answer

2

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

20. Consider the provided pseudo code. How can you initialize an empty two-way linked list?

```
Define Structure Node
   data: Integer
   prev: Pointer to Node
   next: Pointer to Node
End Define

Define Structure TwoWayLinkedList
   head: Pointer to Node
   tail: Pointer to Node
End Define
```

Answer

struct TwoWayLinkedList* list = malloc(sizeof(struct Node));

Status : Wrong ***Marks : 0/1***

Rajalakshmi Engineering College

Name: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 2_COD_Question 1

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Your task is to create a program to manage a playlist of items. Each item is represented as a character, and you need to implement the following operations on the playlist.

Here are the main functionalities of the program:

Insert Item: The program should allow users to add items to the front and end of the playlist. Items are represented as characters.Display Playlist: The program should display the playlist containing the items that were added.

To implement this program, a doubly linked list data structure should be used, where each node contains an item character.

Input Format

The input consists of a sequence of space-separated characters, representing the items to be inserted into the doubly linked list.

The input is terminated by entering - (hyphen).

Output Format

The first line of output prints "Forward Playlist: " followed by the linked list after inserting the items at the end.

The second line prints "Backward Playlist: " followed by the linked list after inserting the items at the front.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: a b c -

Output: Forward Playlist: a b c
Backward Playlist: c b a

Answer

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

struct Node {
    char item;
    struct Node* next;
    struct Node* prev;
};

// You are using GCC
void insertAtEnd(struct Node** head, char item) {
   struct Node* newnode = (struct Node*)malloc(sizeof(struct Node));
   newnode->item = item;
   newnode->next = NULL;
   if(*head == NULL){
       newnode->prev = NULL;
       *head=newnode;
       return;
   }
```

```
    struct Node*temp = *head;
    while(temp->next!=NULL){
        temp=temp->next;
    }
    temp->next = newnode;
    newnode->prev = temp;
}
void displayForward(struct Node* head) {
    struct Node* temp = head;
    while(temp!=NULL){
        printf("%c ",temp->item);
        temp= temp->next;
    }
    printf("\n");
}

void displayBackward(struct Node* tail) {
    struct Node* temp = tail;
    while(temp!=NULL){
        printf("%c ",temp->item);
        temp=temp->prev;
    }
    printf("\n");
}

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

int main() {
    struct Node* playlist = NULL;
    char item;

    while (1) {
        scanf(" %c", &item);
        if (item == '-') {
            break;
        }
        insertAtEnd(&playlist, item);
    }

    struct Node* tail = playlist;
    while (tail->next != NULL) {
        tail = tail->next;
    }

    printf("Forward Playlist: ");
    displayForward(playlist);

    printf("Backward Playlist: ");
    displayBackward(tail);

    freePlaylist(playlist);

    return 0;
}
```

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

Rajalakshmi Engineering College

Name: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 2_COD_Question 2

Attempt : 1
Total Mark : 10
Marks Obtained : 0

Section 1 : Coding

1. Problem Statement

Moniksha, a chess coach organizing a tournament, needs a program to manage participant IDs efficiently. The program maintains a doubly linked list of IDs and offers two functions: Append to add IDs as students register, and Print Maximum ID to identify the highest ID for administrative tasks.

This tool streamlines tournament organization, allowing Moniksha to focus on coaching her students effectively.

Input Format

The first line consists of an integer n, representing the number of participant IDs to be added.

The second line consists of n space-separated integers representing the participant IDs.

Output Format

The output displays a single integer, representing the maximum participant ID.

If the list is empty, the output prints "Empty list!".

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 3
163 137 155
Output: 163

Answer

-

Status : Skipped ***Marks : 0/10***

Rajalakshmi Engineering College

Name: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

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;
   while(temp!= NULL){
       printf("%d ",temp->info);
       temp = temp->next;
   }
   printf("\n");
}
```

```
void insertAtFront(int data) {
  struct node* newnode = (struct node*)malloc(sizeof(node));
  newnode->info = data;
  newnode->prev =NULL;
  newnode->next = start;
  if(start!= NULL){
     start->prev = newnode;
  }
  start = newnode;
  printf("Node Inserted\n");
}

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: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

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 : 10

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 id;
   struct node* prev;
   struct node* next;
};
struct node *head =NULL;
struct node *tail = NULL;
void insert(int id){
   struct node* nnode = (struct node*)malloc(sizeof(node));
   nnode->id = id;
   nnode->next = NULL;
   if(head == NULL){
      nnode->prev = NULL;
      head = nnode;
      tail = nnode;
   }
   else{
      tail->next = nnode;
      nnode->prev = tail;
      tail = nnode;
   }
}
```

```
void display(){
    struct node*temp = head;
    while(temp !=NULL){
        printf("%d ",temp->id);
        temp = temp->next;
    };
    printf("\n");
}
int main(){
    int N,id;
    scanf("%d",&N);
    for(int i=0;i<N;i++){
        scanf("%d",&id);
        insert(id);
    }
    display();
    return 0;
}
```

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

Rajalakshmi Engineering College

Name: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

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() {
   return front == -1;
}

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

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

int dequeue() {
   if (isEmpty()){
      printf("No orders in the queue.\n");
      return 0;
   }
   printf("Dequeued Order: %c",orders[front]);
   if(front == rear){
      front = rear = -1;
   }
   else{
      front++;
   }
   return 1;
}

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: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 4_COD_Question 2

Attempt : 2
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() {
   return front == -1 ;
}

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

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

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

void display() {
   if(isEmpty()){
      printf("Queue is empty.\n");
      return;
   }
   printf("Helpdesk Ticket IDs in the queue are:\n");
   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: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

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 isEmpty(){
   return front == -1;
}
int isFull(){
    return rear == max-1;
}
int insertq(int *data)
{
   if(isFull()){
      return 0;
   }
   if(isEmpty()){
      front = 0;
      rear = -1;
   }
   queue[++rear] = *data;
   return 1;
}

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

```
    if(front == rear){
      rear = front = -1;
    }
    else{
      front++;
    }
    return 1;
}

void display()
{
    if(isEmpty()){
      printf("Queue is empty.\n");
      return;
    }
    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: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

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

```
#include<stdio.h>
#define max 5
int rear = -1,front = -1;
int data[max];
int isFull(){
    return rear == max -1;
}
int isEmpty(){
    return front == -1;
}
int queue(int n){
    if(isFull()){
       printf("Queue is full. Cannot enqueue.\n");
       return 0;
    }
    if(isEmpty()){
       front = 0;
       rear = -1;
    }
    printf("Print job with %d pages is enqueued.\n",n);
    data[++rear] = n;
    return 1;
}
int dequeue(){
    if(isEmpty()){
       printf("Queue is empty");
       return 0;
    }
    printf("Processing print job: %d pages\n",data[front]);
    if(front == rear){
       front = rear = -1;
    }
    else{
       front++;
    }
    return 1;
}
void print(){
    if(isEmpty()){
       printf("Queue is empty.\n");
       return;
    }
    printf("Print jobs in the queue: ");
    for(int i=front;i<=rear;i++){
       printf("%d ",data[i]);
    }
    printf("\n");
}
```

```
int main(){
   int ch=0;
   while(ch!=4){
     scanf("%d",&ch);
     if(ch ==1){
        int n;
        scanf("%d",&n);
        queue(n);
     }
     else if(ch == 2){
        dequeue();
     }
     else if(ch == 3){
        print();
     }
     else if(ch!=4){
        printf("Invalid option.\n");
     }
   }
   printf("Exiting program");
}
```

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

Rajalakshmi Engineering College

Name: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 4_COD_Question 5

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

You are tasked with implementing basic operations on a queue data structure using a linked list.

You need to write a program that performs the following operations on a queue:

Enqueue Operation: Implement a function that inserts an integer element at the rear end of the queue.Print Front and Rear: Implement a function that prints the front and rear elements of the queue. Dequeue Operation: Implement a function that removes the front element from the queue.

Input Format

The first line of input consists of an integer N, representing the number of elements to be inserted into the queue.

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

Output Format

The first line prints "Front: X, Rear: Y" where X is the front and Y is the rear elements of the queue.

The second line prints the message indicating that the dequeue operation (front element removed) is performed: "Performing Dequeue Operation:".

The last line prints "Front: M, Rear: N" where M is the front and N is the rear elements after the dequeue operation.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 5
12 56 87 23 45

Output: Front: 12, Rear: 45
Performing Dequeue Operation:
Front: 56, Rear: 45

Answer

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

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

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

int isempty(){
   return front == NULL;
}
void enqueue(int d) {
   struct Node*n= (struct Node*)malloc(sizeof(struct Node));
   n->data = d;
   n->next = NULL;
   if(isempty()){
       front = rear = n;
   }
   else{
       rear->next = n;
       rear = n;
   }
}

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

void dequeue() {
   if(isempty()){
       return;
   }
   else{
       struct Node* temp = front;
       front = front->next;
       free(temp);
   }
}
int main() {
   int n, data;
   scanf("%d", &n);
   for (int i = 0; i < n; i++) {
       scanf("%d", &data);
       enqueue(data);
   }
   printFrontRear();
   printf("Performing Dequeue Operation:\n");
   dequeue();
   printFrontRear();
   return 0;
}
```

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

Rajalakshmi Engineering College

Name: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 4_PAH

Attempt : 1
Total Mark : 50
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

Guide Harish in developing a simple queue system for a customer service center. The customer service center can handle up to 25 customers at a time. The queue needs to support basic operations such as adding a customer to the queue, serving a customer (removing them from the queue), and displaying the current queue of customers.

Use an array for implementation.

Input Format

The first line of the input consists of an integer N, the number of customers arriving at the service center.

The second line consists of N space-separated integers, representing the customer IDs in the order they arrive.

Output Format

After serving the first customer in the queue, display the remaining customers in the queue.

If a dequeue operation is attempted on an empty queue, display "Underflow".

If the queue is empty, display "Queue is empty".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
101 102 103 104 105

Output: 102 103 104 105

Answer

```
#include<stdio.h>
int main(){
   int n,queue[25];
   scanf("%d",&n);
   if(n==0){
      int dummy;
      scanf("%d",&dummy);
      printf("Underflow\n");
      printf("Queue is empty\n");
   }
   else{
      for(int i=0;i<n;i++){
         scanf("%d",&queue[i]);
      }
      for(int i=1;i<n;i++){
         printf("%d ",queue[i]);
      }
      printf("\n");
   }
}
```

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

2. Problem Statement

You are tasked with developing a simple ticket management system for a customer support department. In this system, customers submit support tickets, which are processed in a First-In-First-Out (FIFO) order. The system needs to handle the following operations:

Ticket Submission (Enqueue Operation): New tickets are submitted by customers. Each ticket is assigned a unique identifier (represented by an integer). When a new ticket arrives, it should be added to the end of the queue.

Ticket Processing (Dequeue Operation): The support team processes tickets in the order they are received. The ticket at the front of the queue is processed first. After processing, the ticket is removed from the queue.

Display Ticket Queue: The system should be able to display the current state of the ticket queue, showing the sequence of ticket identifiers from front to rear.

Input Format

The first input line contains an integer n, the number of tickets submitted by customers.

The second line consists of a single integer, representing the unique identifier of each submitted ticket, separated by a space.

Output Format

The first line displays the "Queue: " followed by the ticket identifiers in the queue after all tickets have been submitted.

The second line displays the "Queue After Dequeue: " followed by the ticket identifiers in the queue after processing (removing) the ticket at the front.

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

Sample Test Case

Input: 6
14 52 63 95 68 49
Output: Queue: 14 52 63 95 68 49
Queue After Dequeue: 52 63 95 68 49

Answer

```
#include<stdio.h>
int main(){
   int n,queue[20];
   scanf("%d",&n);
   for(int i=0;i<n;i++){
      scanf("%d",&queue[i]);
   }
   printf("Queue: ");
   for(int i=0;i<n;i++){
      printf("%d ",queue[i]);
   }
   printf("\n");
   printf("Queue After Dequeue: ");
   for(int i=1;i<n;i++){
      printf("%d ",queue[i]);
   }
   printf("\n");
}
```

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

3. Problem Statement

You've been assigned the challenge of developing a queue data structure using a linked list.

The program should allow users to interact with the queue by enqueuing positive integers and subsequently dequeuing and displaying elements.

Input Format

The input consists of a series of integers, one per line. Enter positive integers into the queue.

Enter -1 to terminate input.

Output Format

The output prints the space-separated dequeued elements.

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

Sample Test Case

Input: 1
2
3
4
-1
Output: Dequeued elements: 1 2 3 4

Answer

// You are using GCC

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

4. Problem Statement

Sharon is developing a queue using an array. She wants to provide the functionality to find the Kth largest element. The queue should support the addition and retrieval of the Kth largest element effectively. The maximum capacity of the queue is 10.

Assist her in 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.

The third line consists of an integer K.

Output Format

For each enqueued element, print a message: "Enqueued: " followed by the element.

The last line prints "The [K]th largest element: " followed by the Kth largest element.

Refer to the sample output for formatting specifications.

Sample Test Case

```
Input: 5
23 45 93 87 25
4
Output: Enqueued: 23
Enqueued: 45
Enqueued: 93
Enqueued: 87
Enqueued: 25
The 4th largest element: 25
```

Answer

```
#include<stdio.h>
#define MAX 10

void sortDescending(int arr[],int n){
   for(int i=0;i<n-1;i++){
      for(int j=i+1;j<n;j++){
         if(arr[i]<arr[j]){
            int temp = arr[i];
            arr[i] = arr[j];
            arr[j] = temp;
         }
      }
   }
}

int main(){
   int queue[MAX];
   int n,k;
   scanf("%d",&n);
```

```
    for(int i=0;i<n;i++){
        scanf("%d",&queue[i]);
        printf("Enqueued: %d\n",queue[i]);
    }
    scanf("%d",&k);
    sortDescending(queue,n);
    printf("The %dth largest element: %d\n",k,queue[k-1]);
}
```

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

5. Problem Statement

Amar is working on a project where he needs to implement a special type of queue that allows selective dequeuing based on a given multiple. He wants to efficiently manage a queue of integers such that only elements not divisible by a given multiple are retained in the queue after a selective dequeue operation.

Implement a program to assist Amar in managing his selective queue.

Example

Input:

5

10 2 30 4 50

5

Output:

Original Queue: 10 2 30 4 50

Queue after selective dequeue: 2 4

Explanation:

After selective dequeue with a multiple of 5, the elements that are multiples of 5 should be removed. Therefore, only 10, 30, and 50 should be removed from the queue. The updated Queue is 2 4.

Input Format

The first line contains an integer n, representing the number of elements initially present in the queue.

The second line contains n space-separated integers, representing the elements of the queue.

The third line contains an integer multiple, representing the divisor for selective dequeue operation.

Output Format

The first line of output prints "Original Queue: " followed by the space-separated elements in the queue before the dequeue operation.

The second line prints "Queue after selective dequeue: " followed by the remaining space-separated elements in the queue, after deleting elements that are the multiples of the specified number.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 5
10 2 30 4 50
5
Output: Original Queue: 10 2 30 4 50
Queue after selective dequeue: 2 4

Answer

```
#include<stdio.h>
int main(){
   int n,multiple;
   int queue[50],result[50];
   scanf("%d",&n);
   for(int i =0;i<n;i++){
      scanf("%d",&queue[i]);
   }
   scanf("%d",&multiple);
```

```
    printf("Original Queue: ");
    for(int i=0;i<n;i++){
        printf("%d ",queue[i]);
    }
    printf("\n");

    int j=0;
    for(int i=0;i<n;i++){
        if(queue[i]%multiple!=0){
            result[j++] = queue[i];
        }
    }
    printf("Queue after selective dequeue: ");
    for(int i=0;i<j;i++){
        printf("%d",result[i]);
    }
    printf("\n");
    return 0;
}
```

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

Rajalakshmi Engineering College

Name: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
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 : 10

Section 1 : Coding

1. 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

-

Status : Skipped ***Marks : 0/10***

2. Problem Statement

Imagine you are developing a basic task management system for a small team of software developers. Each task is represented by an integer, where positive integers indicate valid tasks and negative integers indicate erroneous tasks that need to be removed from the queue before processing.

Write a program using the queue with a linked list that allows the team to add tasks to the queue, remove all erroneous tasks (negative integers), and then display the valid tasks that remain in the queue.

Input Format

The first line consists of an integer N, representing the number of tasks to be added to the queue.

The second line consists of N space-separated integers, representing the tasks. Tasks can be both positive (valid) and negative (erroneous).

Output Format

The output displays the following format:

For each task enqueued, print a message "Enqueued: " followed by the task value.

The last line displays the "Queue Elements after Dequeue: " followed by removing all erroneous (negative) tasks and printing the valid tasks remaining in the queue in the order they were enqueued.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
12 -54 68 -79 53
Output: Enqueued: 12
Enqueued: -54
Enqueued: 68
Enqueued: -79
Enqueued: 53
Queue Elements after Dequeue: 12 68 53

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;
    }
    printf("Enqueued: %d\n",value);
}

void removeNegatives(){
    struct Node* curr = front;
    struct Node* prev = NULL;
    while(curr!=NULL){
        if(curr->data<0){
            if(curr == front){
                front = front->next;
                free(curr);
                curr = front;
            }
            else{
                prev->next = curr->next;
                if(curr== rear){
                    rear = prev;
                }
                free(curr);
                curr = prev->next;
            }
        }
        else{
            prev = curr;
            curr = curr->next;
        }
    }
}

void displayQueue(){
    struct Node* temp = front;
    printf("Queue Elements after Dequeue: ");
    while(temp!=NULL){
        printf("%d ",temp->data);
        temp = temp->next;
    }
}

int main(){
    int n,value;
    scanf("%d",&n);
    for(int i=0;i<n;i++){
        scanf("%d",&value);
        enqueue(value);
    }
    removeNegatives();
    displayQueue();

}
```

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

3. 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

-

Status : Skipped ***Marks : 0/10***

Rajalakshmi Engineering College

Name: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 5_MCQ

Attempt : 1
Total Mark : 15
Marks Obtained : 11

Section 1 : MCQ

1. Which of the following is the correct post-order traversal of a binary search tree with nodes: 50, 30, 20, 55, 32, 52, 57?

Answer

20, 32, 30, 52, 57, 55, 50

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

2. Which of the following operations can be used to traverse a Binary Search Tree (BST) in ascending order?

Answer

Inorder traversal

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

3. While inserting the elements 5, 4, 2, 8, 7, 10, 12 in a binary search tree, the element at the lowest level is _________.

Answer

12

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

4. Find the pre-order traversal of the given binary search tree.

Answer

1, 4, 2, 18, 14, 13

Status : Wrong ***Marks : 0/1***

5. Find the in-order traversal of the given binary search tree.

Answer

13, 2, 1, 4, 14, 18

Status : Wrong ***Marks : 0/1***

6. Which of the following is a valid preorder traversal of the binary search tree with nodes: 18, 28, 12, 11, 16, 14, 17?

Answer

18, 12, 11, 16, 14, 17, 28

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

7. While inserting the elements 71, 65, 84, 69, 67, 83 in an empty binary search tree (BST) in the sequence shown, the element in the lowest level is ________.

Answer

67

Status : Correct *Marks : 1/1*

8. Which of the following is the correct pre-order traversal of a binary search tree with nodes: 50, 30, 20, 55, 32, 52, 57?

Answer

50, 30, 20, 32, 55, 52, 57

Status : Correct *Marks : 1/1*

9. Find the preorder traversal of the given binary search tree.

Answer

9, 2, 1, 6, 4, 7, 10, 14

Status : Correct *Marks : 1/1*

10. Which of the following is the correct in-order traversal of a binary search tree with nodes: 9, 3, 5, 11, 8, 4, 2?

Answer

2, 3, 4, 5, 8, 9, 11

Status : Correct *Marks : 1/1*

11. Find the postorder traversal of the given binary search tree.

Answer

1, 2, 4, 13, 14, 18

Status : Wrong *Marks : 0/1*

12. In a binary search tree with nodes 18, 28, 12, 11, 16, 14, 17, what is the value of the left child of the node 16?

Answer

14

Status : Correct *Marks : 1/1*

13. Find the post-order traversal of the given binary search tree.

Answer

10, 17, 20, 18, 15, 32, 21

Status : Correct *Marks : 1/1*

14. The preorder traversal of a binary search tree is 15, 10, 12, 11, 20, 18, 16, 19. Which one of the following is the postorder traversal of the tree?

Answer

19, 16, 18, 20, 11, 12, 10, 15

Status : Wrong *Marks : 0/1*

15. How many distinct binary search trees can be created out of 4 distinct keys?

Answer

14

Status : Correct *Marks : 1/1*

Rajalakshmi Engineering College

Name: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

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 : 10

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){
       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;
}

struct TreeNode* findMin(struct TreeNode* root) {
   if(root->left == NULL){
       return root;
   }
   findMin(root->left);
}

struct TreeNode* deleteNode(struct TreeNode* root, int key) {
   if(root==0)
   return 0;
   if(key<root->data)
   root->left=deleteNode(root->left,key);
   else if(key>root->data)
   root->right=deleteNode(root->right,key);
   else{
       if(root->left==0){
           struct TreeNode *temp=root->right;
           free(root);
           return temp;
       }
       if(root->right==0){
           struct TreeNode *temp=root->left;
           free(root);
           return temp;
       }
       else{
           struct TreeNode *temp=findMin(root->right);
           root->data=temp->data;
           root->right=deleteNode(root->right,temp->data);
       }
   }
   return root;
}

void inorderTraversal(struct TreeNode* root) {
   if(root!=0){
       inorderTraversal(root->left);
       printf("%d ",root->data);
       inorderTraversal(root->right);
   }
}

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 : Correct *Marks : 10/10*

Rajalakshmi Engineering College

Name: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

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* root) {
    if(root!=NULL){
        printf("%d ",root->data);
        printPreorder(root->left);
        printPreorder(root->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: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

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

```
#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;
}

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;
}

int search(struct Node* root,int key){
    if(root==NULL)
        return 0;
    if(root->data == key)
        return 1;
    if(key<root->data)
        return search(root->left,key);
    else
        return search(root->right,key);
}

int main(){
    struct Node* root = NULL;
    int key,n;
    scanf("%d",&n);
    for(int i=0;i<n;i++){
        int value;
        scanf("%d ",&value);
        root = insert(root,value);
    }
    scanf("%d",&key);
    if(search(root,key)){
        printf("Value %d is found in the tree.\n",key);
    }
    else{
        printf("Value %d is not found in the tree.\n",key);
    }
    return 0;
}
```

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

Rajalakshmi Engineering College

Name: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

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

```
#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;
}

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;
}

int search(struct Node* root,int key){
    if(root==NULL)
        return 0;
    if(root->data == key)
        return 1;
    if(key<root->data)
        return search(root->left,key);
    else
        return search(root->right,key);
}

int main(){
    struct Node* root = NULL;
    int key,n;
    scanf("%d",&n);
    for(int i=0;i<n;i++){
        int value;
        scanf("%d ",&value);
        root = insert(root,value);
    }
    scanf("%d",&key);
    if(search(root,key)){
        printf("Value %d is found in the tree.\n",key);
    }
    else{
        printf("Value %d is not found in the tree.\n",key);
    }
    return 0;
}
```

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

Rajalakshmi Engineering College

Name: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

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;
}

// You are using GCC
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)
        return;
    displayTreePostOrder(root->left);
    displayTreePostOrder(root->right);
    printf("%d ",root->data);
}

int findMinValue(struct Node* root) {
    struct Node* current = root;
    while(current && current ->left != NULL)
        current = current->left;
    return  current->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: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

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: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 5_PAH_Updated

Attempt : 1
Total Mark : 50
Marks Obtained : 37.5

Section 1 : Coding

1. Problem Statement

Arun is exploring operations on binary search trees (BST). He wants to write a program with an unsorted distinct integer array that represents the BST keys and construct a height-balanced BST from it.

After constructing, he wants to perform the following operations that can alter the structure of the tree and traverse them using a level-order traversal:

InsertionDeletion

Your task is to assist Arun in completing the program without any errors.

Input Format

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

The second line consists of N space-separated integers, representing the initial keys.

The third line consists of an integer X, representing the new key to be inserted into the BST.

The fourth line consists of an integer Y, representing the key to be deleted from the BST.

Output Format

The first line of output prints "Initial BST: " followed by a space-separated list of keys in the initial BST after constructing it in level order traversal.

The second line prints "BST after inserting a new node X: " followed by a space-separated list of keys in the BST after inserting X n level order traversal.

The third line prints "BST after deleting node Y: " followed by a space-separated list of keys in the BST after deleting Y n level order traversal.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
25 14 56 28 12
34
12

Output: Initial BST: 25 14 56 12 28
BST after inserting a new node 34: 25 14 56 12 28 34
BST after deleting node 12: 25 14 56 28 34

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;
}
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;
}

void inordertr(struct treenode* root){
   if(root==NULL)
      return ;
   inordertr(root->left);
   printf("%d ",root->data);
   inordertr(root->right);
}

struct treenode* trimbst(struct treenode* root,int min,int max){
   if(root==NULL)
      return NULL;
   root->left = trimbst(root->left,min,max);
   root->right = trimbst(root->right,min,max);
   if(root->data<min){
      struct treenode* rightchild = root->right;
      free(root);
      return rightchild;
   }
   if(root->data>max){
      struct treenode* leftchild = root->left;
      free(root);
      return leftchild;
   }
```

```
    return root;
}
int main(){
    int n,min,max;
    scanf("%d",&n);
    struct treenode* root = NULL;
    for(int i=0;i<n;i++){
        int key;
        scanf("%d ",&key);
        root = insert(root,key);
    }
    scanf("%d %d",&min,&max);
    root = trimbst(root,min,max);
    inordertr(root);
}
```

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

2. Problem Statement

Viha, a software developer, is working on a project to automate searching for a target value in a Binary Search Tree (BST). She needs to create a program that takes an integer target value as input and determines if that value is present in the BST or not.

Write a program to assist Viha.

Input Format

The first line of input consists of integers separated by spaces, which represent the elements to be inserted into the BST. The input is terminated by entering -1.

The second line consists of an integer target, which represents the target value to be searched in the BST.

Output Format

If the target value is found in the BST, print "[target] is found in the BST".

Else, print "[target] is not found in the BST"

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5 3 7 1 4 6 8 -1
4
Output: 4 is found in the BST

Answer

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

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

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;
}

int search(struct node* root,int target){
   if(root ==NULL) return 0;
   if(target == root->data) return 1;
   if(target <root->data)
      return search(root->left,target);
   else
      return search(root->right,target);
}

int main(){
   struct node* root = NULL;
   int value;
   while(1){
      scanf("%d",&value);
      if(value == -1) break;
      root = insert(root,value);
   }
   int target;
   scanf("%d",&target);
   if(search(root,target))
      printf("%d is found in the BST\n",target);
   else
      printf("%d is not found in the BST\n",target);
   return 0;
}
```

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

3. Problem Statement

Aishu is participating in a coding challenge where she needs to reconstruct a Binary Search Tree (BST) from given preorder traversal data and then print the in-order traversal of the reconstructed BST.

Since Aishu is just learning about tree data structures, she needs your help to write a program that does this efficiently.

Input Format

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

The second line of input contains n integers separated by spaces, which represent the preorder traversal of the BST.

Output Format

The output displays n space-separated integers, representing the in-order traversal of the reconstructed BST.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 6
10 5 1 7 40 50

Output: 1 5 7 10 40 50

Answer

```
#include<stdio.h>
#include<stdlib.h>
#include<limits.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;
}
struct treenode* buildbst(int preorder[],int* index,int min,int max,int n){
   if(*index >= n)
      return NULL;
   int key = preorder[*index];
   if(key<min || key>max)
      return NULL;
   struct treenode* root = createnode(key);
   (*index)++;
   root->left = buildbst(preorder,index,min,key-1,n);
   root->right = buildbst(preorder,index,key+1,max,n);
   return root;
}

void inordertraversal(struct treenode* root){
   if(root == NULL)
      return;
   inordertraversal(root->left);
   printf("%d ",root->data);
   inordertraversal(root->right);
}

int main(){
   int n;
   scanf("%d",&n);
   int preorder[n];
   for(int i =0;i<n;i++){
      scanf("%d",&preorder[i]);
   }
   int index = 0;
   struct treenode* root = buildbst(preorder,&index,INT_MIN,INT_MAX,n);
   inordertraversal(root);
}
```

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

4. Problem Statement

Yogi is working on a program to manage a binary search tree (BST) containing integer values. He wants to implement a function that removes nodes from the tree that fall outside a specified range defined by a minimum and maximum value.

Help Yogi by writing a function that achieves this.

Input Format

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

The second line consists of N space-separated integers, representing the elements to be inserted into the BST.

The third line consists of two space-separated integers min and max, representing the minimum value and the maximum value of the range.

Output Format

The output prints the remaining elements of the BST in an in-order traversal, after removing nodes that fall outside the specified range.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
10 5 15 20 12
5 15
Output: 5 10 12 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;
}
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;
}
```

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

struct treenode* trimbst(struct treenode* root,int min,int max){
    if(root==NULL)
        return NULL;
    root->left = trimbst(root->left,min,max);
    root->right = trimbst(root->right,min,max);
    if(root->data<min){
        struct treenode* rightchild = root->right;
        free(root);
        return rightchild;
    }
    if(root->data>max){
        struct treenode* leftchild = root->left;
        free(root);
        return leftchild;
    }
    return root;
}
int main(){
    int n,min,max;
    scanf("%d",&n);
    struct treenode* root = NULL;
    for(int i=0;i<n;i++){
        int key;
        scanf("%d ",&key);
        root = insert(root,key);
    }
    scanf("%d %d",&min,&max);
    root = trimbst(root,min,max);
    inordertr(root);
}
```

Status : Partially correct ***Marks : 7.5/10***

5. Problem Statement

Joseph, a computer science student, is interested in understanding binary search trees (BST) and their node arrangements. He wants to create a program to explore BSTs by inserting elements into a tree and displaying the nodes using post-order traversal of the tree.

Write a program to help Joseph implement 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 output prints N space-separated integer values after the post-order traversal.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 4
10 15 5 3

Output: 3 5 15 10

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 postorder(struct node* root){
   if(root == NULL) return;
   postorder(root->left);
   postorder(root->right);
   printf("%d ",root->data);
}

int main(){
   int n,data;
   struct node* root = NULL;
   scanf("%d",&n);
   for(int i=0;i<n;i++){
       scanf("%d",&data);
       root= insert(root,data);
   }
   postorder(root);
}
```

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

Rajalakshmi Engineering College

Name: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 5_CY_Updated

Attempt : 1
Total Mark : 30
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

John is building a system to store and manage integers using a binary search tree (BST). He needs to add a feature that allows users to search for a specific integer key in the BST using recursion.

Implement functions to create the BST and perform a recursive search for an integer.

Input Format

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

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

The third line consists of an integer representing, the key to be searched.

Output Format

The output prints whether the given key is present in the binary search tree or not.

Refer to the sample output for the exact format.

Sample Test Case

Input: 7
10 5 15 3 7 12 20
12
Output: The key 12 is found in the binary search tree

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;
}

int search(struct node*root,int key){
   if(root == NULL)
      return 0;
```

```
    if(root->data == key)
        return 1;
    if(key<root->data)
        return search(root->left,key);
    else
        return search(root->right,key);
}
int main(){
    int n,k,val;
    scanf("%d",&n);
    struct node* root = NULL;
    for(int i =0;i<n;i++){
        scanf("%d",&val);
        root = insert(root,val);
    }
    scanf("%d",&k);
    if(search(root,k))
        printf("The key %d is found in the binary search tree\n", k);
    else
        printf("The key %d is not found in the binary search tree\n",k);
}
```

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

2. Problem Statement

Jake is learning about binary search trees(BST) and their operations. He wants to implement a program that can delete a node from a BST based on the given key value and print the remaining nodes in an in-order traversal.

Assist Jake in the program.

Input Format

The first line of input consists of an integer n, representing the number of elements in BST.

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

The third line consists of an integer x, representing the key value of the node to be deleted.

Output Format

The first line of output prints "Before deletion: " followed by the in-order traversal of the initial BST.

The second line prints "After deletion: " followed by the in-order traversal after the deletion of the key value.

If the key value is not present in the BST, print the original tree as it is.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
8 6 4 3 1
4
Output: Before deletion: 1 3 4 6 8
After deletion: 1 3 6 8

Answer

-

Status : Skipped ***Marks : 0/10***

3. Problem Statement

Emily is studying binary search trees (BST). She wants to write a program that inserts characters into a BST and then finds and prints the minimum and maximum values.

Guide her with the program.

Input Format

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

The second line consists of N space-separated characters.

Output Format

The first line of output prints "Minimum value: " followed by the minimum value of the given inputs.

The second line prints "Maximum value: " followed by the maximum value of the given inputs.

Refer to the sample outputs for formatting specifications.

Sample Test Case

Input: 5
Z E W T Y

Output: Minimum value: E
Maximum value: Z

Answer

-

Status : Skipped ***Marks : 0/10***

Rajalakshmi Engineering College

Name: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 6_MCQ_Updated_1

Attempt : 1
Total Mark : 20
Marks Obtained : 18

Section 1 : MCQ

1. What is the main advantage of Quicksort over Merge Sort?

Answer

Quicksort requires less auxiliary space

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

2. Let P be a quick sort program to sort numbers in ascending order using the first element as a pivot. Let t1 and t2 be the number of comparisons made by P for the inputs {1, 2, 3, 4, 5} and {4, 1, 5, 3, 2}, respectively. Which one of the following holds?

Answer

t1 &lt; t2

Status : Wrong ***Marks : 0/1***

3. In a quick sort algorithm, where are smaller elements placed to the pivot during the partition process, assuming we are sorting in increasing order?

Answer

To the left of the pivot

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

4. Which of the following modifications can help Quicksort perform better on small subarrays?

Answer

Switching to Insertion Sort for small subarrays

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

5. Consider the Quick Sort algorithm, which sorts elements in ascending order using the first element as a pivot. Then which of the following input sequences will require the maximum number of comparisons when this algorithm is applied to it?

Answer

52 25 76 67 89

Status : Wrong ***Marks : 0/1***

6. What happens during the merge step in Merge Sort?

Answer

Two sorted subarrays are combined into one sorted array

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

7. The following code snippet is an example of a quick sort. What do the 'low' and 'high' parameters represent in this code?

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

Answer

The range of elements to sort within the array

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

8. Which of the following is true about Quicksort?

Answer

It is an in-place sorting algorithm

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

9. What is the best sorting algorithm to use for the elements in an array that are more than 1 million in general?

Answer

Quick sort.

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

10. Which of the following sorting algorithms is based on the divide and conquer method?

Answer

Merge Sort

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

11. Which of the following methods is used for sorting in merge sort?

Answer

merging

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

12. In a quick sort algorithm, what role does the pivot element play?

Answer

It is used to partition the array

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

13. Which of the following is not true about QuickSort?

Answer

It can be implemented as a stable sort

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

14. Which of the following scenarios is Merge Sort preferred over Quick Sort?

Answer

When sorting linked lists

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

15. Merge sort is ________.

Answer

Comparison-based sorting algorithm

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

16. Which of the following strategies is used to improve the efficiency of Quicksort in practical implementations?

Answer

Choosing the pivot randomly or using the median-of-three method

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

17. Which of the following statements is true about the merge sort algorithm?

Answer

It requires additional memory for merging

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

18. Why is Merge Sort preferred for sorting large datasets compared to Quick Sort?

Answer

Merge Sort has better worst-case time complexity

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

19. Is Merge Sort a stable sorting algorithm?

Answer

Yes, always stable.

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

20. What happens when Merge Sort is applied to a single-element array?

Answer

The array remains unchanged and no merging is required

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

Rajalakshmi Engineering College

Name: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

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 merged[], int arr1[], int arr2[], int n1, int n2) {
   int i=0,j=0,k=0;
   while(i<n1 && j<n2){
      if(arr1[i]<arr2[j]) merged[k++] = arr1[i++];
      else merged[k++] = arr2[j++];
   }
   while(i<n1)merged[k++] = arr1[i++];
   while(j<n2) merged[k++] = arr2[j++];
}

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

```
    int j=0,k=0;
    while(i<mid && j<n - mid){
        if(left[i]<right[j]) arr[k++] = left[i++];
        else arr[k++] = right[j++];
    }
    while(i<mid) arr[k++] = left[i++];
    while(j<n-mid)arr[k++] = right[j++];
}

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: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

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: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

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: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

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>

void swap(int *a,int *b){
   int t = *a;
   *a=*b;
   *b = t;
}

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++;
         swap(&arr[i],&arr[j]);
      }
   }
   swap(&arr[i+1],&arr[high]);
   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 arr[], int n, int k) {
    quickSort(arr,0,n-1);
    printf("%d\n",arr[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: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

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>

void merge(double arr[], int left, int mid, int right) {
   int n1 = mid-left+1,n2 = right-mid;
   double L[n1],R[n2];
   for(int i =0;i<n1;i++)L[i] = arr[left+i];
   for(int j=0;j<n2;j++) R[j] = arr[mid+1+j];
   int i =0,j=0,k=left;
   while(i<n1 && j<n2){
      if(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 mid = l+(r-l)/2;
      mergeSort(arr,l,mid);
      mergeSort(arr,mid+1,r);
      merge(arr,l,mid,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: Mohamed Aseel
Email: 240701319@rajalakshmi.edu.in
Roll no: 240701319
Phone: 9159006104
Branch: REC
Department: I CSE FC
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 6_CY_Updated

Attempt : 1
Total Mark : 30
Marks Obtained : 0

Section 1 : Coding

1. Problem Statement

Meera is organizing her art supplies, which are represented as a list of integers: red (0), white (1), and blue (2). She needs to sort these supplies so that all items of the same color are adjacent, in the order red, white, and blue. To achieve this efficiently, Meera decides to use QuickSort to sort the items. Can you help Meera arrange her supplies in the desired order?

Input Format

The first line of input consists of an integer n, representing the number of items in the list.

The second line consists of n space-separated integers, where each integer is either 0 (red), 1 (white), or 2 (blue).

Output Format

The output prints the sorted list of integers in a single line, where integers are arranged in the order red (0), white (1), and blue (2).

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 6
2 0 2 1 1 0

Output: Sorted colors:
0 0 1 1 2 2

Answer

-

Status : - ***Marks : 0/10***

2. Problem Statement

Sheela wants to distribute cookies to her children, but each child will only be happy if the cookie size meets or exceeds their individual greed factor. She has a limited number of cookies and wants to make as many children happy as possible. Priya decides to sort both the greed factors and cookie sizes using QuickSort to efficiently match cookies with children. Your task is to help Sheela determine the maximum number of children that can be made happy.

Input Format

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

The second line contains n space-separated integers, where each integer represents the greed factor of a child.

The third line contains an integer m, representing the number of cookies.

The fourth line contains m space-separated integers, where each integer represents the size of a cookie.

Output Format

The output prints a single integer, representing the maximum number of children that can be made happy.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 3
1 2 3
2
1 1
Output: The child with greed factor: 1

Answer

-

Status : Skipped ***Marks : 0/10***

3. Problem Statement

Ravi is given an array of integers and is tasked with sorting it in a unique way. He needs to sort the elements in such a way that the elements at odd positions are in descending order, and the elements at even positions are in ascending order. Ravi decided to use the Insertion Sort algorithm for this task.

Your task is to help ravi, to create even_odd_insertion_sort function to sort the array as per the specified conditions and then print the sorted array.

Example

Input:

10

25 36 96 58 74 14 35 15 75 95

Output:

96 14 75 15 74 36 35 58 25 95

Input Format

The first line of input consists of a single integer, N, which represents the size of the array.

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

Output Format

The output displays the sorted array using the even-odd insertion sort algorithm and prints the sorted array.

Refer to the sample output for formatting specifications.

Sample Test Case

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

Answer

```
#include<stdio.h>
void isorta(int arr[],int size){
   for(int i =1;i<size;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 isortd(int arr[],int size){
   for(int i=1;i<size;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;
    }
}

int main(){
    int n;
    scanf("%d",&n);
    int arr[n];
    for(int i =0;i<n;i++) scanf("%d",&arr[i]);
    int oddcount = (n+1)/2;
    int evencount = n/2;
    int oddpos[oddcount],evenpos[evencount];
    int oddidx = 0,evenidx =0;
    for(int i =0;i<n;i++){
        if((i+1)%2==1)oddpos[oddidx++] = arr[i];
        else evenpos[evenidx++] = arr[i];
    }
    isortd(oddpos,oddcount);
    isorta(evenpos,evencount);
    oddidx =0,evenidx = 0;
    for(int i=0;i<n;i++){
        if((i+1)%2==0)arr[i]=oddpos[oddidx++];
        else arr[i] = evenpos[evenidx++];
    }
    for(int i=0;i<n;i++)printf("%d ",arr[i]);
    printf("\n");
}
```

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