Rajalakshmi Engineering College

Name: Manisha S
Email: 240801191@rajalakshmi.edu.in
Roll no: 240801191
Phone: 8667375390
Branch: REC
Department: I ECE AF
Batch: 2028
Degree: B.E - ECE

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. 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***

2. 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 -> 16 -> 22 -> 45 -> 16

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

3. 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 -> 16 -> 6

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

4. 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**

5. 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***

6. 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***

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

Answer

Binary search

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

8. 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**

9. 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**

10. 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***

Rajalakshmi Engineering College

Name: Manisha S
Email: 240801191@rajalakshmi.edu.in
Roll no: 240801191
Phone: 8667375390
Branch: REC
Department: I ECE AF
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 1_COD_Question 1

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

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

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

Input Format

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

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

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

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

Output Format

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

Sample Test Case

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

Output: 18

Answer

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

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

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

Rajalakshmi Engineering College

Name: Manisha S
Email: 240801191@rajalakshmi.edu.in
Roll no: 240801191
Phone: 8667375390
Branch: REC
Department: I ECE AF
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 1_COD_Question 2

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

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

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

Input Format

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

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

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

Position starts from 1.

Output Format

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

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

Refer to the sample output for formatting specifications.

Sample Test Case

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

Answer

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

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

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

void insert(int value)
{
   struct node* newNode=(struct node*)malloc(sizeof(struct node));
   newNode->data=value;
   newNode->next=NULL;

   if(head==NULL)
   {
      head=newNode;
      tail=newNode;
```

```
    }
    else{
        tail->next=newNode;
        tail=newNode;
    }
}
void deleteNode(int position)
{
    if(head==NULL)
    {
        printf("Invalid position.DE11etion not possible.\n");
        return;
    }
    struct node* temp=head;

    if(position ==1)
    {
        head=temp->next;
        free(temp);
        display_List();
        return;
    }
    for(int i=1;temp != NULL && i<position - 1;i++)
    {
        temp=temp->next;
    }
    if(temp==NULL || temp->next==NULL)
    {
        printf("Invalid position.  Deletion not possible.\n");
        return;
    }
    struct node* nodeToDelete =temp->next;
    temp->next = temp->next->next;
    free(nodeToDelete);

    display_List();
}
void display_List()
{
    if(head==NULL){
        return;
    }
```

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


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: Manisha S
Email: 240801191@rajalakshmi.edu.in
Roll no: 240801191
Phone: 8667375390
Branch: REC
Department: I ECE AF
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 4_COD_Question 3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

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

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

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

Input Format

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

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

Option 2: Delete an element from the queue.

Option 3: Display the elements in the queue.

Output Format

For option 1 (insertion):-

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

For option 2 (deletion):-

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

For option 3 (display):-

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

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

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 1 10

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

Answer

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

#define max 5

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

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

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

void display() {
   if (front == -1 || front > rear) {
      printf("Queue is empty.\n");
   } else {
```

```
        printf("Elements in the queue are: ");
        for (int i = front; i <= rear; i++) {
            printf("%d ", queue[i]);
        }
        printf("\n");
    }
}

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

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

Rajalakshmi Engineering College

Name: Manisha S
Email: 240801191@rajalakshmi.edu.in
Roll no: 240801191
Phone: 8667375390
Branch: REC
Department: I ECE AF
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 1_COD_Question 3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

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

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

Input Format

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

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

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

which the new character node needs to be inserted.

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

Output Format

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

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

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

Refer to the sample output for formatting specifications.

Sample Test Case

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

Answer

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

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

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

Status : Correct

Marks : 10/10

Rajalakshmi Engineering College

Name: Manisha S
Email: 240801191@rajalakshmi.edu.in
Roll no: 240801191
Phone: 8667375390
Branch: REC
Department: I ECE AF
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 1_COD_Question 4

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

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

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

Input Format

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

The second line consists of N space-separated integers.

Output Format

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

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
78 89 34 51 67

Output: 67 51 34 89 78

Answer

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

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

struct Node *head=NULL,*newnode,*ptr;
void insertAtFront(struct Node **head,int a)
{newnode=(struct Node*)malloc(sizeof(struct Node));
newnode->data=a;
newnode->next=NULL;
newnode->next=*head;
   *head=newnode;

}
void printList(struct Node*head)
{ptr=head;
   while(ptr!=NULL)
   {printf("%d ",ptr->data);
   ptr=ptr->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: Manisha S
Email: 240801191@rajalakshmi.edu.in
Roll no: 240801191
Phone: 8667375390
Branch: REC
Department: I ECE AF
Batch: 2028
Degree: B.E - ECE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 1_COD_Question 5

Attempt : 1
Total Mark : 10
Marks Obtained : 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 gpa
{
   float value;
   struct gpa* next;
}Node;
Node* newnode(float value)
{
   Node* newgpa = (Node*) malloc(sizeof(Node));
   newgpa->value = value;
   newgpa->next = NULL;
   return newgpa;
}
Node* insertAtStart(Node* head, float value)
{
   Node* newgpa = newnode(value);
   newgpa->next = head;
   return newgpa;
}
```

```
void traverse(Node* head)
{
   while(head != NULL)
   {
      printf("GPA: %.1f\n",head->value);
      head = head->next;
   }
}
void deleteAtPosition(Node** head, int pos)
{
   pos -= 1;
   Node* temp = *head;
   if(pos == 0)
   {
      *head = temp->next;
      free(temp);
      return;
   }
   while(--pos)
   {
      temp = temp->next;
   }
   Node* temp1 = temp->next;
   temp->next = temp->next->next;
   free(temp1);
}
int main()
{
   int n,pos;
   float value;
   scanf("%d",&n);
   Node* head = NULL;
   for(int i=0; i<n; i++)
   {
      scanf("%f",&value);
      head = insertAtStart(head, value);
   }
   scanf("%d",&pos);
   deleteAtPosition(&head, pos);
   traverse(head);
}
```

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

Rajalakshmi Engineering College

Name: Manisha S
Email: 240801191@rajalakshmi.edu.in
Roll no: 240801191
Phone: 8667375390
Branch: REC
Department: I ECE AF
Batch: 2028
Degree: B.E - ECE

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(Node));
   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: Manisha S
Email: 240801191@rajalakshmi.edu.in
Roll no: 240801191
Phone: 8667375390
Branch: REC
Department: I ECE AF
Batch: 2028
Degree: B.E - ECE

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)
    {
        len++;
        temp = temp->next;
    }
    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***