Rajalakshmi Engineering College

Name: MITHUN CHAKRAVARTHY
Email: 241801158@rajalakshmi.edu.in
Roll no: 241801158
Phone: 9360341516
Branch: REC
Department: I AI & DS AF
Batch: 2028
Degree: B.E - AI & DS

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 2_MCQ_Updated

Attempt : 1
Total Mark : 20
Marks Obtained : 18

Section 1 : MCQ

1. 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***

2. 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***

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

Answer

2

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

4. 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 head

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

5. 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***

6. 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 TwoWayLinkedList)); list->head = NULL; list->tail = NULL;

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

7. 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***

8. 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***

9. 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***

10. 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->Bwd->Fwd = X->Fwd; X->Fwd->Bwd = X->Bwd;

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

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 information is stored in a doubly-linked list's nodes?

Answer

All of the mentioned options

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

13. 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***

14. How do you reverse a doubly linked list?

Answer

By swapping the next and previous pointers of each node

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

15. 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***

16. 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***

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

Answer

prev

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

18. 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***

19. 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

6 <--> 5 <--> 4 <--> 3 <--> 2 <--> 1.

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

20. 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->next = head; if (head != NULL) { head->prev = newNode; } head = newNode; }

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

Rajalakshmi Engineering College

Name: MITHUN CHAKRAVARTHY
Email: 241801158@rajalakshmi.edu.in
Roll no: 241801158
Phone: 9360341516
Branch: REC
Department: I AI & DS AF
Batch: 2028
Degree: B.E - AI & DS

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;
   newNode -> prev = NULL;
   if(*head == 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 = head;
    while(temp != NULL)
    {
       struct Node* nextNode = temp -> next;
       free(temp);
```

```
        temp = nextNode;
    }
    head = NULL;
}

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: MITHUN CHAKRAVARTHY
Email: 241801158@rajalakshmi.edu.in
Roll no: 241801158
Phone: 9360341516
Branch: REC
Department: I AI & DS AF
Batch: 2028
Degree: B.E - AI & DS

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 2_CY

Attempt : 1
Total Mark : 30
Marks Obtained : 30

Section 1 : Coding

1. Problem Statement

Krishna needs to create a doubly linked list to store and display a sequence of integers. Your task is to help write a program to read a list of integers from input, store them in a doubly linked list, and then display the list.

Input Format

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

The second line of input consists of n space-separated integers.

Output Format

The output prints a single line displaying the integers in the order they were added to the doubly linked list, separated by spaces.

If nothing is added (i.e., the list is empty), it will display "List is empty".

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 5
1 2 3 4 5

Output: 1 2 3 4 5

Answer

```
// You are using GCC
#include<stdio.h>
#include<stdlib.h>
struct Node {
    int data;
    struct Node* prev;
    struct Node* next;
};
struct Node* createNode(int data){
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    newNode->data = data;
    newNode->prev = NULL;
    newNode->next = NULL;
    return newNode;
}
void insertEnd(struct Node** head,int data){
    struct Node* newNode = createNode(data);
    if(*head == NULL){
        *head = newNode;
        return;
    }
    struct Node* temp = *head;
    while(temp->next != NULL){
        temp = temp->next;
    }
    temp->next = newNode;
    newNode->prev = temp;
}
```

```
void printList (struct Node* head){
    struct Node* temp = head;
    if(temp == NULL){
        printf("List is empty\n");
        return;
    }
    while(temp != NULL){
        printf("%d",temp->data);
        temp = temp->next;
    }
    printf("\n");
}
int main(){
    int n,data;
    struct Node* head = NULL;
    scanf("%d",&n);
    for(int i = 0;i < n;i++){
        scanf("%d", &data);
        insertEnd(&head,data);
    }
    printList(head);
    return 0;
}
```

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

2. Problem Statement

Aarav is working on a program to analyze his test scores, which are stored in a doubly linked list. He needs a solution to input scores into the list and determine the highest score.

Help him by providing code that lets users enter test scores into the doubly linked list and find the maximum score efficiently.

Input Format

The first line consists of an integer N, representing the number of elements to be initially inserted into the doubly linked list.

The second line consists of N space-separated integers, denoting the score to be inserted.

Output Format

The output prints an integer, representing the highest score present in the list.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 4
89 71 2 70
Output: 89

Answer

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

// Structure for a node in the doubly linked list
struct Node {
    int score;
    struct Node* next;
    struct Node* prev;
};

// Function to create a new node
struct Node* createNode(int score) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    newNode->score = score;
    newNode->next = NULL;
    newNode->prev = NULL;
    return newNode;
}

// Function to insert a score at the end of the doubly linked list
void insertScore(struct Node** head, int score) {
    struct Node* newNode = createNode(score);
    if (*head == NULL) {
        *head = newNode;
```

```
    } else {
        struct Node* temp = *head;
        while (temp->next != NULL) {
            temp = temp->next;
        }
        temp->next = newNode;
        newNode->prev = temp;
    }
}

// Function to find the maximum score in the doubly linked list
int findMaxScore(struct Node* head) {
    int maxScore = head->score;
    struct Node* temp = head;

    while (temp != NULL) {
        if (temp->score > maxScore) {
            maxScore = temp->score;
        }
        temp = temp->next;
    }
    return maxScore;
}

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

    struct Node* head = NULL;

    // Input scores and insert them into the doubly linked list
    for (int i = 0; i < N; i++) {
        int score;
        scanf("%d", &score);
        insertScore(&head, score);
    }

    // Find and print the maximum score
    int maxScore = findMaxScore(head);
    printf("%d\n", maxScore);

    return 0;
```

}

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

3. Problem Statement

Imagine you're managing a store's inventory list, and some products were accidentally entered multiple times. You need to remove the duplicate products from the list to ensure each product appears only once.

You have an unsorted doubly linked list of product IDs. Some of these product IDs may appear more than once, and your goal is to remove any duplicates.

Input Format

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

The second line of input consists of n space-separated integers representing the list elements.

Output Format

The output prints the final after removing duplicate nodes, separated by a space.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 10
12 12 10 4 8 4 6 4 4 8
Output: 8 4 6 10 12

Answer

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

// Structure for a node in the doubly linked list
struct Node {
    int data;
    struct Node* next;
    struct Node* prev;
};

// Function to create a new node
struct Node* createNode(int data) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    newNode->data = data;
    newNode->next = NULL;
    newNode->prev = NULL;
    return newNode;
}


void insertNode(struct Node** head,int data){
    struct Node* nn=createNode(data);
    if(*head==NULL){
        *head=nn;
        return;
    }
    nn->next=*head;
    (*head)->prev=nn;
    *head=nn;
}

// Function to remove duplicate elements from the doubly linked list
void removeDuplicates(struct Node** head) {
    struct Node* current = *head;
    struct Node* temp;

    while (current != NULL) {
        temp = current->next;

        while (temp != NULL) {
            if (current->data == temp->data) {
                // Remove the duplicate node
```

```
            if (temp->next != NULL) {
                temp->next->prev = temp->prev;
            }
            if (temp->prev != NULL) {
                temp->prev->next = temp->next;
            }
            struct Node* toDelete = temp;
            temp = temp->next;  // Move temp to next node
            free(toDelete);     // Free memory of the duplicate node
        } else {
            temp = temp->next; // Continue checking next nodes
        }
    }
    current = current->next; // Move to the next node
  }
}

// Function to print the list
void printList(struct Node* head) {
    struct Node* temp = head;
    while (temp != NULL) {
        printf("%d ", temp->data);
        temp = temp->next;
    }
    printf("\n");
}

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

    struct Node* head = NULL;

    // Input the list elements and insert them into the doubly linked list
    for (int i = 0; i < n; i++) {
        int data;
        scanf("%d", &data);
        insertNode(&head, data);
    }

    // Remove duplicates from the doubly linked list
    removeDuplicates(&head);
```

```
    // Print the final list after removing duplicates
    printList(head);

    return 0;
}
```

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