Rajalakshmi Engineering College

Name: Sidharth P
Email: 240701514@rajalakshmi.edu.in
Roll no: 240701514
Phone: 7695968308
Branch: REC
Department: I CSE FE
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 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

```
// You are using GCC
#include <stdio.h>
#include <stdlib.h>

// Define a Node structure
struct Node {
   float gpa;
   struct Node* next;
};

// Function to insert a GPA at the front of the list
void insertAtFront(struct Node** head, float gpa) {
   struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
   newNode->gpa = gpa;
   newNode->next = *head;
   *head = newNode;
}

// Function to delete a node at a given position (1-based index)
void deleteAtPosition(struct Node** head, int position) {
   if (*head == NULL) return;

   struct Node* temp = *head;

   // If deleting the head node
   if (position == 1) {
       *head = temp->next;
       free(temp);
       return;
   }

   // Find the previous node of the node to be deleted
   struct Node* prev = NULL;
   for (int i = 1; temp != NULL && i < position; i++) {
       prev = temp;
       temp = temp->next;
   }

   // If position is out of bounds
   if (temp == NULL) return;

   prev->next = temp->next;
   free(temp);
}

// Function to print the GPA list
void printList(struct Node* head) {
   struct Node* temp = head;
   while (temp != NULL) {
       printf("GPA: %.1f\n", temp->gpa);  // Print GPA rounded to one decimal place
       temp = temp->next;
   }
}

int main() {
   struct Node* head = NULL;
   int n, position;
   float gpa;

   // Read number of students
   scanf("%d", &n);
   // Read GPA values and insert them at the front
   for (int i = 0; i < n; i++) {
       scanf("%f", &gpa);
       insertAtFront(&head, gpa);
   }

   // Read the position to delete
   scanf("%d", &position);

   // Delete the node at the given position
   deleteAtPosition(&head, position);

   // Print the updated list
   printList(head);

   // Free memory
   struct Node* temp;
   while (head != NULL) {
       temp = head;
       head = head->next;
       free(temp);
   }

   return 0;
}
```

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