Tutorial 6: Recursion

ELEC 278: Fundamentals of Information Structures

The learning goals for Tutorial 6 are:

- Write recursive functions for linked list operations.

Problem 1. Below are several iterative functions for linked list operations. Rewrite the operations using recursive functions.

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

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

// Function to insert a node at the beginning
void insertAtBeginning(struct Node** head, int newData) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    newNode->data = newData;
    newNode->next = *head;
    *head = newNode;
}

// Function to delete a node
void deleteNode(struct Node** head, int key) {
    struct Node* temp = *head, *prev = NULL;
    if (temp != NULL && temp->data == key) {
        *head = temp->next;
        free(temp);
        return;
    }
    while (temp != NULL && temp->data != key) {
        prev = temp;
        temp = temp->next;
    }
    if (temp == NULL) return;
    prev->next = temp->next;
    free(temp);
}

// Function to search for a node
int searchNode(struct Node* head, int key) {
    struct Node* current = head;
    while (current != NULL) {
        if (current->data == key)
            return 1;
        current = current->next;
    }
    return 0;
}

// Function to reverse the linked list
void reverseList(struct Node** head) {
    struct Node* prev = NULL;
    struct Node* current = *head;
    struct Node* next = NULL;
    while (current != NULL) {
        next = current->next;
        current->next = prev;
        prev = current;
        current = next;
    }
    *head = prev;
}

// Function to calculate the length of the linked list
int listLength(struct Node* head) {
    int length = 0;
    struct Node* current = head;
    while (current != NULL) {
        length++;
        current = current->next;
    }
    return length;
}

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

int main() {
    struct Node* head = NULL;
    insertAtBeginning(&head, 1);
    insertAtBeginning(&head, 2);
    insertAtBeginning(&head, 3);
    printf("Linked list:\n");
    printList(head);
    printf("Length: %d\n", listLength(head));
    printf("Searching for 2: %s\n", searchNode(head, 2) ? "Found" : "Not Found");
    reverseList(&head);
    printf("Reversed linked list:\n");
    printList(head);
    deleteNode(&head, 2);
    printf("Linked list after deletion of 2:\n");
    printList(head);
    return 0;
}
```

Solution.

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

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

// Recursive function to insert a node at the beginning
void insertAtBeginning(struct Node** head, int newData) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    newNode->data = newData;
    newNode->next = *head;
    *head = newNode;
}

// Recursive function to delete a node
struct Node* deleteNode(struct Node* head, int key) {
    if (head == NULL) return head;
    if (head->data == key) {
        struct Node* temp = head->next;
        free(head);
        return temp;
    }
    head->next = deleteNode(head->next, key);
    return head;
}

// Recursive function to search for a node
int searchNode(struct Node* head, int key) {
    if (head == NULL) return 0;
    if (head->data == key) return 1;
    return searchNode(head->next, key);
}

// Recursive function to reverse the linked list
struct Node* reverseList(struct Node* head) {
    if (head == NULL || head->next == NULL) return head;
    struct Node* rest = reverseList(head->next);
    head->next->next = head;
    head->next = NULL;
    return rest;
}

// Recursive function to calculate the length of the linked list
int listLength(struct Node* head) {
    if (head == NULL) return 0;
    return 1 + listLength(head->next);
}

// Recursive function to display the linked list
void printList(struct Node* head) {
    if (head == NULL) {
        printf("NULL\n");
        return;
    }
    printf("%d -> ", head->data);
    printList(head->next);
}

int main() {
    struct Node* head = NULL;
    insertAtBeginning(&head, 1);
    insertAtBeginning(&head, 2);
    insertAtBeginning(&head, 3);
    printf("Linked list:\n");
    printList(head);
    printf("Length: %d\n", listLength(head));
    printf("Searching for 2: %s\n", searchNode(head, 2) ? "Found" : "Not Found");
    head = reverseList(head);
    printf("Reversed linked list:\n");
    printList(head);
    head = deleteNode(head, 2);
    printf("Linked list after deletion of 2:\n");
    printList(head);
    return 0;
}
```