Rajalakshmi Engineering College

Name: Nitin Aakash
Email: 240701370@rajalakshmi.edu.in
Roll no: 240701370
Phone: 9498349045
Branch: REC
Department: I CSE FD
Batch: 2028
Degree: B.E - CSE

Scan to verify results

NeoColab_REC_CS23231_DATA STRUCTURES

REC_DS using C_Week 3_COD_Question 5

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : Coding

1. Problem Statement

Milton is a diligent clerk at a school who has been assigned the task of managing class schedules. The school has various sections, and Milton needs to keep track of the class schedules for each section using a stack-based system.

He uses a program that allows him to push, pop, and display class schedules for each section. Milton's program uses a stack data structure, and each class schedule is represented as a character. Help him write a program using a linked list.

Input Format

The input consists of integers corresponding to the operation that needs to be performed:

Choice 1: Push the character onto the stack. If the choice is 1, the following input is a space-separated character, representing the class schedule to be pushed onto the stack.

Choice 2: Pop class schedule from the stack

Choice 3: Display the class schedules in the stack.

Choice 4: Exit the program.

Output Format

The output displays messages according to the choice and the status of the stack:

- If the choice is 1, push the given class schedule to the stack and display the following: "Adding Section: [class schedule]"
- If the choice is 2, pop the class schedule from the stack and display the following: "Removing Section: [class schedule]"
- If the choice is 2, and if the stack is empty without any class schedules, print "Stack is empty. Cannot pop."
- If the choice is 3, print the class schedules in the stack in the following: "Enrolled Sections: " followed by the class schedules separated by space.
- If the choice is 3, and there are no class schedules in the stack, print "Stack is empty"
- If the choice is 4, exit the program and display the following: "Exiting the program"
- If any other choice is entered, print "Invalid choice"

Refer to the sample output for the exact format.

Sample Test Case

Input: 1 d
1 h
3
2

```
3
4
Output: Adding Section: d
Adding Section: h
Enrolled Sections: h d
Removing Section: h
Enrolled Sections: d
Exiting program
```

Answer

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

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

struct Node* top = NULL;
```

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

typedef struct Node {
   char schedule;
   struct Node* next;
} Node;

typedef struct Stack {
   Node* top;
} Stack;

Stack* createStack() {
   Stack* stack = (Stack*)malloc(sizeof(Stack));
   stack->top = NULL;
   return stack;
}

void push(Stack* stack, char schedule) {
   Node* newNode = (Node*)malloc(sizeof(Node));
   newNode->schedule = schedule;
```

```
    newNode->next = stack->top;
    stack->top = newNode;
    printf("Adding Section: %c\n", schedule);
}

void pop(Stack* stack) {
    if (stack->top == NULL) {
        printf("Stack is empty. Cannot pop.\n");
        return;
    }
    Node* temp = stack->top;
    char schedule = temp->schedule;
    stack->top = stack->top->next;
    free(temp);
    printf("Removing Section: %c\n", schedule);
}
void display(Stack* stack) {
    if (stack->top == NULL) {
        printf("Stack is empty\n");
        return;
    }
    Node* current = stack->top;
    printf("Enrolled Sections: ");
    while (current != NULL) {
        printf("%c ", current->schedule);
        current = current->next;
    }
    printf("\n");
}

void freeStack(Stack* stack) {
    Node* current = stack->top;
    Node* nextNode;
    while (current != NULL) {
        nextNode = current->next;
        free(current);
        current = nextNode;
    }
    free(stack);
}

int main() {
    Stack* stack = createStack();
    int choice;
    char schedule;

    while (1) {
        printf("");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                printf("");
                scanf(" %c", &schedule);
                if (isalpha(schedule)) {
                    push(stack, schedule);
                } else {
                    printf("Invalid character. Please enter an alphabetic character.\n");
                }
                break;
            case 2:
                pop(stack);
                break;
            case 3:
                display(stack);
                break;
            case 4:
                printf("Exiting program\n");
                freeStack(stack);
                return 0;
            default:
                printf("Invalid choice\n");
                break;
        }
    }

    return 0;
}
int main() {
    int choice;
    char value;
    do {
        scanf("%d", &choice);
        switch (choice) {
```

```
        case 1:
            scanf(" %c", &value);
            push(value);
            break;
        case 2:
            pop();
            break;
        case 3:
            displayStack();
            break;
        case 4:
            printf("Exiting program\n");
            break;
        default:
            printf("Invalid choice\n");
        }
    } while (choice != 4);

    return 0;
}
```

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