Institute of Computer Technology

B. Tech. Computer Science and Engineering

Sub: DS

Course Code: 2CSE302

Practical – 5

Name: Jaymin Gondaliya

Enrollment No: 23162171007

Sem - 3

Branch: CS

Class: A

Batch: 32

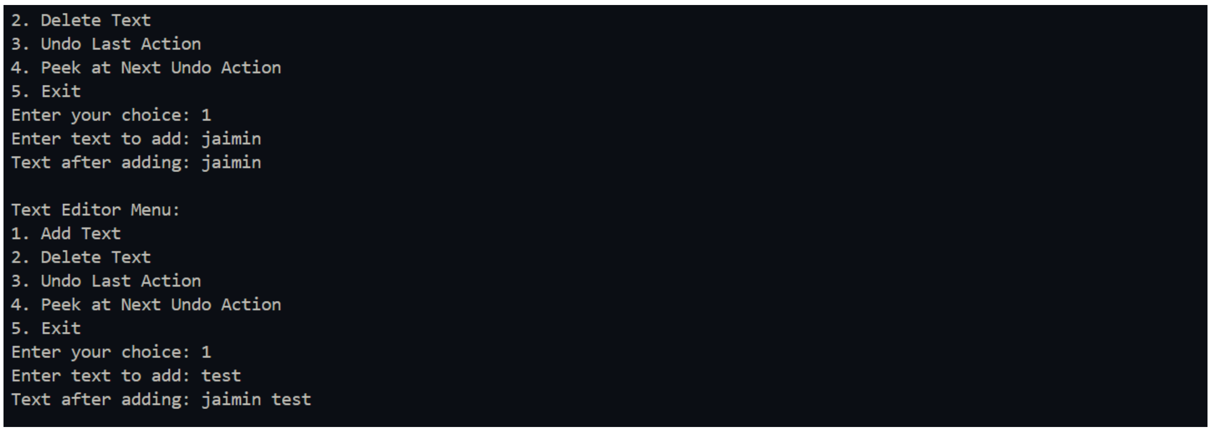
Problem Definition-1: Implement an Undo Mechanism in a Text Editor

Code:

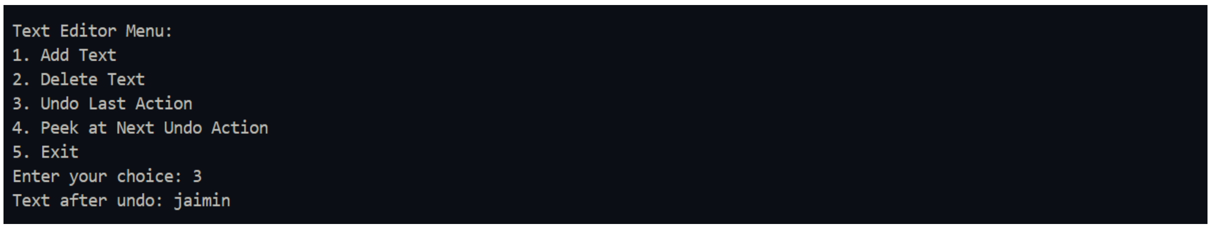
```
#include <stdio.h>
#include <string.h>

#define MAX 100

// Define a structure to store text editing actions
typedef struct {
    char action[MAX]; // Type of action: "add", "delete"
    char text[MAX];   // The text associated with the action
} Action;

// Stack to store actions
Action undoStack[MAX];
int top = -1; // Initialize top of stack

// Function to push an action onto the stack
void push(Action action) {
    if (top < MAX - 1) {
        undoStack[++top] = action;
    } else {
        printf("Stack is full. Cannot push more actions.\n");
    }
}

// Function to pop the most recent action from the stack
Action pop() {
    if (top >= 0) {
        return undoStack[top--];
    } else {
        printf("Stack is empty. No actions to undo.\n");
        Action empty = {"", ""}; // Return an empty action if the stack is
empty
        return empty;
    }
}

// Function to peek at the most recent action without removing it
Action peek() {
    if (top >= 0) {
        return undoStack[top];
    } else {
        printf("Stack is empty. No actions to peek.\n");
        Action empty = {"", ""}; // Return an empty action if the stack is
empty
        return empty;
    }
}

// Simulate adding text
void addText(char editorText[], char text[]) {
    strcat(editorText, text); // Add text to the editor
    printf("Text after adding: %s\n", editorText);

    Action action;
    strcpy(action.action, "add");
    strcpy(action.text, text);
    push(action); // Push the add action onto the stack
}

// Simulate deleting text
void deleteText(char editorText[], int length) {
    int len = strlen(editorText);
    char deletedText[MAX];
    strncpy(deletedText, editorText + len - length, length);
    deletedText[length] = '\0';

    editorText[len - length] = '\0'; // Delete text from the editor
    printf("Text after deleting: %s\n", editorText);
```

```c
    Action action;
    strcpy(action.action, "delete");
    strcpy(action.text, deletedText);
    push(action); // Push the delete action onto the stack
}

// Function to undo the last action
void undo(char editorText[]) {
    Action lastAction = pop(); // Get the last action

    if (strcmp(lastAction.action, "add") == 0) {
        // If the last action was add, delete the added text
        int len = strlen(lastAction.text);
        editorText[strlen(editorText) - len] = '\0';
    } else if (strcmp(lastAction.action, "delete") == 0) {
        // If the last action was delete, add the deleted text back
        strcat(editorText, lastAction.text);
    }

    printf("Text after undo: %s\n", editorText);
}

// Function to display the next action to be undone
void displayNextUndoAction() {
    Action nextAction = peek();
    if (strcmp(nextAction.action, "") != 0) {
        printf("Next undo action: %s \"%s\"\n", nextAction.action,
nextAction.text);
    }
}

int main() {
    char editorText[MAX] = ""; // Initialize the text editor content
    int choice;
    char inputText[MAX];
    int deleteLength;

    while (1) {
        printf("\nText Editor Menu:\n");
        printf("1. Add Text\n");
        printf("2. Delete Text\n");
        printf("3. Undo Last Action\n");
        printf("4. Peek at Next Undo Action\n");
        printf("5. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);
        getchar(); // To consume the newline character left by scanf
        switch (choice) {
            case 1:
                printf("Enter text to add: ");
                fgets(inputText, MAX, stdin);
                inputText[strcspn(inputText, "\n")] = '\0'; // Remove newline
character
                addText(editorText, inputText);
                break;
            case 2:
                printf("Enter number of characters to delete: ");
                scanf("%d", &deleteLength);
                deleteText(editorText, deleteLength);
                break;
            case 3:
                undo(editorText);
                break;
            case 4:
                displayNextUndoAction();
                break;
            case 5:
                printf("Exiting the editor.\n");
                return 0;
            default:
                printf("Invalid choice. Please try again.\n");
        }
    }

    return 0;
}
```

Output-



```
2. Delete Text
3. Undo Last Action
4. Peek at Next Undo Action
5. Exit
Enter your choice: 1
Enter text to add: jaimin
Text after adding: jaimin

Text Editor Menu:
1. Add Text
2. Delete Text
3. Undo Last Action
4. Peek at Next Undo Action
5. Exit
Enter your choice: 1
Enter text to add: test
Text after adding: jaimin test
```



```
Text Editor Menu:
1. Add Text
2. Delete Text
3. Undo Last Action
4. Peek at Next Undo Action
5. Exit
Enter your choice: 3
Text after undo: jaimin
```