## Using pointer to array

#include <stdio.h>

#include <string.h>

#define cols 20

#define rows 5

int cnt = 0;

void print\_menu()

{

    printf("\*\*\*Name Data Base \*\*\*\n"

           "i : input\n"

           "d : delete\n"

           "p : print\n"

           "s : sort\n"

           "q : quit\n"

           "...Enter choice: ");

}

void input(char (\*p)[cols])

{

    if (cnt == rows)

    {

        printf("Database is full. Cannot add more names.\n");

        return;

    }

    printf("Enter name: ");

    fgets(p[cnt], cols, stdin);

    p[cnt][strcspn(p[cnt], "\n")] = '\0'; // Remove the trailing newline

    printf("Name inserted successfully...!!!\n");

    cnt++;

}

void delete(char (\*p)[cols])

{

    if (cnt == 0)

    {

        printf("Database is empty. Nothing to delete.\n");

        return;

    }

    int index;

    printf("Enter index to delete: ");

    scanf("%d", &index);

    if (index < 0 || index >= cnt)

    {

        printf("Invalid index. No name to delete.\n");

        return;

    }

    for (int i = index; i < cnt - 1; i++)

    {

        strcpy(p[i], p[i + 1]);

    }

    cnt--;

    printf("Name deleted successfully...!!!\n");

}

void print(char (\*p)[cols])

{

    if (cnt == 0)

    {

        printf("Database is empty. No names to print.\n");

        return;

    }

    for (int i = 0; i < cnt; i++)

    {

        printf("Name %d: %s\n", i, p[i]);

    }

}

void sort(char (\*p)[cols])

{

    for (int i = 0; i < cnt - 1; i++)

    {

        for (int j = 0; j < cnt - i - 1; j++)

        {

            if (strcmp(p[j], p[j + 1]) > 0)

            {

                char temp[cols];

                strcpy(temp, p[j]);

                strcpy(p[j], p[j + 1]);

                strcpy(p[j + 1], temp);

            }

        }

    }

    printf("Sorted the Data Base...\n");

}

int main()

{

    char names[rows][cols];

    char choice;

    while (1)

    {

        print\_menu();

        scanf("%c", &choice);

        fflush(stdin); // Clear the input buffer

        switch (choice)

        {

        case 'i':

            input(names);

            break;

        case 'd':

            delete (names);

            break;

        case 'p':

            print(names);

            break;

        case 's':

            sort(names);

            break;

        case 'q':

            printf("\*\*\*Thanks for using name database\*\*\*\n");

            return 0;

        default:

            printf("Invalid choice...!!!\n");

        }

    }

    return 0;

}

## Using Dynamic Memory

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

int cnt = 0;

void print\_menu()

{

    printf("\*\*\*Name Data Base \*\*\*\n"

           "i : input\n"

           "d : delete\n"

           "p : print\n"

           "s : sort\n"

           "q : quit\n"

           "...Enter choice: ");

}

void \*input(char \*names[])

{

    char \*newName = (char \*)malloc(sizeof(char) \* 20);

    printf("Enter name: ");

    fgets(newName, 20, stdin);

    newName[strcspn(newName, "\n")] = '\0'; // Remove the trailing newline

    names = (char \*\*)realloc(names, sizeof(char \*) \* (cnt + 1));

    names[cnt++] = newName;

    printf("Name inserted successfully...!!!\n");

    return names;

}

void \*delete(char \*names[])

{

    if (cnt == 0)

    {

        printf("Database is empty. Nothing to delete.\n");

        return names;

    }

    int index;

    printf("Enter index to delete: ");

    if (scanf("%d", &index) != 1)

    {

        printf("Invalid input. Aborting...\n");

        // Clear the input buffer

        int c;

        while ((c = getchar()) != '\n' && c != EOF)

            ;

        return names;

    }

    if (index < 0 || index >= cnt)

    {

        printf("Invalid index. No name to delete.\n");

        return names;

    }

    free(names[index]);                   // remove element at index

    for (int i = index; i < cnt - 1; i++) // reload arr list

    {

        names[i] = names[i + 1];

    }

    names = (char \*\*)realloc(names, sizeof(char \*) \* (--cnt));

    printf("Name deleted successfully...!!!\n");

    return names;

}

void sort(char \*names[])

{

    for (int i = 0; i < cnt - 1; i++)

    {

        for (int j = 0; j < cnt - i - 1; j++)

        {

            if (strcmp(names[j], names[j + 1]) > 0)

            {

                char \*temp = names[j];

                names[j] = names[j + 1];

                names[j + 1] = temp;

            }

        }

    }

    printf("Sorted the Data Base...\n");

}

void print(char \*names[])

{

    if (cnt == 0)

    {

        printf("Database is empty. No names to print.\n");

        return;

    }

    for (int i = 0; i < cnt; i++)

    {

        printf("Name %d: %s\n", i, names[i]);

    }

}

int main()

{

    char \*\*names = NULL;

    char choice;

    while (1)

    {

        print\_menu();

        scanf("%c", &choice);

        fflush(stdin); // Clear the input buffer

        switch (choice)

        {

        case 'i':

            names = input(names);

            break;

        case 'd':

            names = delete (names);

            break;

        case 'p':

            print(names);

            break;

        case 's':

            sort(names);

            break;

        case 'q':

            printf("\*\*\*Thanks for using name database\*\*\*\n");

            for (int i = 0; i < cnt; i++)

            {

                free(names[i]);

            }

            free(names);

            return 0;

        default:

            printf("Invalid choice...!!!\n");

        }

    }

    return 0;

}

## Using Double Pointer

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

int cnt = 0;

void print\_menu()

{

    printf("\*\*\*Name Data Base \*\*\*\n"

           "i : input\n"

           "d : delete\n"

           "p : print\n"

           "s : sort\n"

           "f : find\n"

           "q : quit\n"

           "...Enter choice: ");

}

char \*getString(char \*str, int maxLen)

{

    if (fgets(str, maxLen, stdin) == NULL)

    {

        return NULL;

    }

    str[strcspn(str, "\n")] = '\0'; // Remove the trailing newline

    return str;

}

void \*input(char \*\*names)

{

    char \*newName = (char \*)malloc(sizeof(char) \* 20);

    printf("Enter name: ");

    if (getString(newName, 20) == NULL)

    {

        free(newName);

        return names;

    }

    names = (char \*\*)realloc(names, sizeof(char \*) \* (cnt + 1));

    names[cnt++] = newName;

    printf("Name inserted successfully...!!!\n");

    return names;

}

void print(char \*\*names)

{

    if (cnt == 0)

    {

        printf("Database is empty. No names to print.\n");

        return;

    }

    for (int i = 0; i < cnt; i++)

    {

        printf("Name %d: %s\n", i, names[i]);

    }

}

void \*delete(char \*\*names)

{

    if (cnt == 0)

    {

        printf("Database is empty. Nothing to delete.\n");

        return names;

    }

    int index;

    printf("Enter index to delete: ");

    if (scanf("%d", &index) != 1)

    {

        printf("Invalid input. Aborting...\n");

        // Clear the input buffer

        int c;

        while ((c = getchar()) != '\n' && c != EOF)

            ;

        return names;

    }

    if (index < 0 || index >= cnt)

    {

        printf("Invalid index. No name to delete.\n");

        return names;

    }

    free(names[index]);

    for (int i = index; i < cnt - 1; i++)

    {

        names[i] = names[i + 1];

    }

    names = (char \*\*)realloc(names, sizeof(char \*) \* (--cnt));

    printf("Name deleted successfully...!!!\n");

    return names;

}

void sort(char \*\*names)

{

    for (int i = 0; i < cnt - 1; i++)

    {

        for (int j = 0; j < cnt - i - 1; j++)

        {

            if (strcmp(names[j], names[j + 1]) > 0)

            {

                char \*temp = names[j];

                names[j] = names[j + 1];

                names[j + 1] = temp;

            }

        }

    }

    printf("Sorted the Data Base...\n");

}

void find(char \*\*names)

{

    char \*searchName = (char \*)malloc(sizeof(char) \* 20);

    printf("Enter name to search: ");

    if (getString(searchName, 20) == NULL)

    {

        free(searchName);

        return;

    }

    for (int i = 0; i < cnt; i++)

    {

        if (strcmp(names[i], searchName) == 0)

        {

            printf("Name found at index %d\n", i);

            free(searchName);

            return;

        }

    }

    printf("Name not found in the database.\n");

    free(searchName);

}

int main()

{

    char \*\*names = NULL;

    char choice;

    while (1)

    {

        print\_menu();

        scanf("%c", &choice);

        fflush(stdin); // Clear the input buffer

        switch (choice)

        {

        case 'i':

            names = input(names);

            break;

        case 'd':

            names = delete (names);

            break;

        case 'p':

            print(names);

            break;

        case 's':

            sort(names);

            break;

        case 'f':

            find(names);

            break;

        case 'q':

            printf("\*\*\*Thanks for using name database\*\*\*\n");

            for (int i = 0; i < cnt; i++)

            {

                free(names[i]);

            }

            free(names);

            return 0;

        default:

            printf("Invalid choice...!!!\n");

        }

    }

    return 0;

}