The Time-Traveller's Messenger

locked

Mr. X, a time traveller from the past, has just landed in the modern world. Fascinated by how people communicate, he quickly realizes that with everyone's jam-packed schedules, texting is the new calling! Determined to stay connected with his contacts, Mr. X seeks your help to create a basic chat messenger.

Features your messenger must include:

Add a new chat: So Mr. X can start new conversations.

Pin a chat: Prioritize important contacts.

Display all chats.

Unpin a chat: When a chat is no longer at the forefront.

Input Format

As per your discretion.

Constraints

No constraints

Output Format

As per your discretion.

Submissions: 66
Max Score: 0
Difficulty: Medium

Rate This Challenge:
☆☆☆☆☆

More

C

```c
#include<stdio.h>
#include<stdlib.h>
#include<ctype.h>
#include<strings.h>
#include<math.h>

typedef struct node{
    char name[50];
    char text[50];
    struct node *prev;
    struct node *next;
}chat;

```

```c
void disp(chat *headp, chat *head);
chat *addchat(chat *headp, chat *head, char *n, char*t);
void pinchat(chat **headp, chat **head, chat *p);
void unpin(chat **headp, chat **head, chat *p);

int main()
{
    chat* head = NULL;//pointer to first node before unpinned chats
    chat* headp = NULL;//pointer to first node amongst pinned chats
    int out = 0;
    printf("***************************************************************\n");
    printf("YOUR CHAT MESSENGER\n");
    printf("***************************************************************\n");

    while(out==0)
    {
        printf("Pick your action:\n");
        printf("To DISPLAY all chats enter 0\n");
        printf("To ADD a chat enter 1\n");
        printf("To PIN a chat print 2\n");
        printf("To UNPIN a chat enter 3\n");
        int op = 0; //takes an option
        printf("Enter option:");
        scanf("%d",&op);
        if(op==1)
        {
            char n[50] =""; //stores name of chat
            char t[50] =""; //stores text in chat
            printf("Enter name:");
            getchar();
            scanf("%[^\n]s",&n);
            getchar();
            printf("Enter text:");
            scanf("%[^\n]s",&t);
            head = addchat(headp,head,n,t);
        }
        else if(op==2)
        {
            printf("Enter name of contact from unpinned chats:\n");
            getchar();
            char cont[50] ="";//name of contact
            disp(headp,head);
            scanf("%[^\n]s",&cont);
            chat *p = head;
            while(strcmp(p->name,cont)!=0)//moves the pointer to the specific chat
            {
                p = p->next;
            }

            pinchat(&headp,&head,p);
        }

        else if(op==3)
        {
            printf("From the pinned chats enter a chat that you want to unpin\n");
            disp(headp,head);
            char in[50] ="";
            getchar();
            scanf("%[^\n]s",&in);
            chat* point2 = headp;
            while(strcmp(point2->name,in)!=0)
            {
                point2 = point2->next;
            }
            unpin(&headp,&head,point2);
        }
        else if(op==0)
        {
            disp(headp,head);
        }
        else
        {
            printf("Wrong Input\n");
            }
        }

        return 0;
}

void disp(chat *headp, chat *head){
    chat *hp=headp;
    if(head==NULL){
        printf("Empty\n");
    }
    else{
        if(headp!=NULL){
            printf("Pinned Chats\n");
            while (headp!=head->next && headp!=NULL){
                printf("Name: %s\n",headp->name);
                printf("Text: %s\n",headp->text);
                headp=headp->next;
            }
        }
    }
    printf("Unpinned Chats\n");
    if(hp!=NULL){
        head=head->next;
    }
    while(head!=NULL){
        printf("Name: %s\n",head->name);
        printf("Text: %s\n",head->text);
        head=head->next;
    }
}

chat *addchat(chat *headp, chat *head, char *n, char*t){
    chat *p=(chat*)malloc(sizeof(chat));
    p->prev=NULL;
    p->next=NULL;
    strcpy(p->name, n);
    strcpy(p->text, t);

    if(head==NULL){
        head=p;
    }
    else if (headp==NULL && head!=NULL){
        p->next=head;
        head->prev=p;
        head=p;
    }
    else if(headp!=NULL){
        if(head->next==NULL){
            p->prev=head;
            p->next=NULL;
            head->next=p;
        }
        else{
            p->prev=head;
            p->next=head->next;
            head->next=p;
            p->next->prev=p;
        }
    }
    return head;
}

void pinchat(chat **headp, chat **head, chat *p){
    //first detach nodes to pin
    if(p->next==NULL){
        if(p->prev==NULL){
            *head=NULL;
        }
        else{
            p->prev->next=NULL;
        }
    }
    else if (p->prev==NULL){
        *head=p->next;
        (*head)->prev=NULL;
    }
    else{
        p->prev->next=p->next;
        p->next->prev=p->prev;
    }
    p->prev=NULL;
    p->next=NULL;

    //attach the nodes
    if(*headp==NULL){
        if(head==NULL){
            *head=p;
            *headp=p;
        }
        else{
            p->next=(*head);
            p->prev=NULL;
            (*head)->prev=p;
            (*head)=p;
            (*head)->prev=NULL;
            (*headp)=p;
        }
    }
    else{
        if((*head)->next==NULL){
            (*head)->next=p;
            p->next=NULL;
            p->prev=(*head);
            (*head)=p;
        }
        else{
            p->next=(*head)->next;
            p->prev=(*head);
            (*head)->next=p;
            p->next->prev=p;
            (*head)=p;
        }
    }
}

void unpin(chat **headp, chat **head, chat *p){
    //Pinned chat p is unpinned and placed after head
    //if first pinned chat is chosen
    if(p==(*headp)){
        if((*headp)==(*head)){
            (*headp)=NULL;
        }
        else{
            //move headp to next pinned chat
            (*headp)=p->next;
            p->next=NULL;
            (*headp)->next=NULL;
            //if last pinned chat
            if((*head)->next==NULL){
                (*head)->next=p;
                p->prev=(*head);
            }
            else{
                p->next=(*head)->next;
                p->prev=(*head);
                (*head)->next=p;
                p->next->prev=p;
            }
        }
    }
    //if p is last pinned chat
    else if (p==(*head)){
        (*head)=(*head)->prev;
    }
    else{
        //remove p from linked list
        p->prev->next=p->next;
        p->next->prev=p->prev;
        //if all chats are pinned
        if((*head)->next==NULL){
            (*head)->next=p;
            p->prev=(*head);
        }
        else{
            p->next=(*head)->next;
            p->prev=(*head);
            (*head)->next=p;
            p->next->prev=p;
        }
    }
}
```

Line: 1 Col: 1

Upload Code as File □ Test against custom input Run Code Submit Code

Interview Prep | Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy |