**EXPERIMENT 3**

**Shantanu swaraj**

**Batch-41**

**SAP ID-500097021**

**Using single linked list, implement its basic operations like insert, delete, and display**

#include <stdio.h>

#include <stdlib.h>

void push();

void pop();

void display();

struct node

{

int value;

struct node \*next;

};

struct node \*head;

void main()

{

int choose;

L1: printf("1:Push into stack , 2:Pop out of stack , 3:Display the stack\n");

printf("Enter the task to do with stack: ");

scanf("%d",&choose);

switch(choose)

{

case 1:

{

push();

break;

}

case 2:

{

pop();

break;

}

case 3:

{

display();

break;

}

default:

{

printf("Program closed...\n");

exit(EXIT\_SUCCESS);

}

};

goto L1;

}

void push()

{

int value;

struct node \*ptr = (struct node\*)malloc(sizeof(struct node));

if(ptr==NULL)

{

printf("Cannot push into stack..\n");

}

else

{

printf("Enter the value:");

scanf("%d",&value);

if(head == NULL)

{

ptr->value = value;

ptr->next = NULL;

head = ptr;

}

else

{

ptr->value = value;

ptr->next =head;

head = ptr;

}

printf("Value pushed successfully...\n");

}

}

void pop()

{

int element;

struct node \*ptr;

if(head==NULL)

{

printf("Stack is empty \nNo element of pop out...\n ");

}

else

{

element = head->value;

ptr = head;

head = head->next;

free(ptr);

printf("Last Element successfully popped/deleted from the stack...\n");

}

}

void display()

{

int i;

struct node \*ptr;

ptr = head;

if(ptr==NULL)

{

printf("The stack is empty.. Nothing to display...\n");

}

else

{

while(ptr!=0)

{

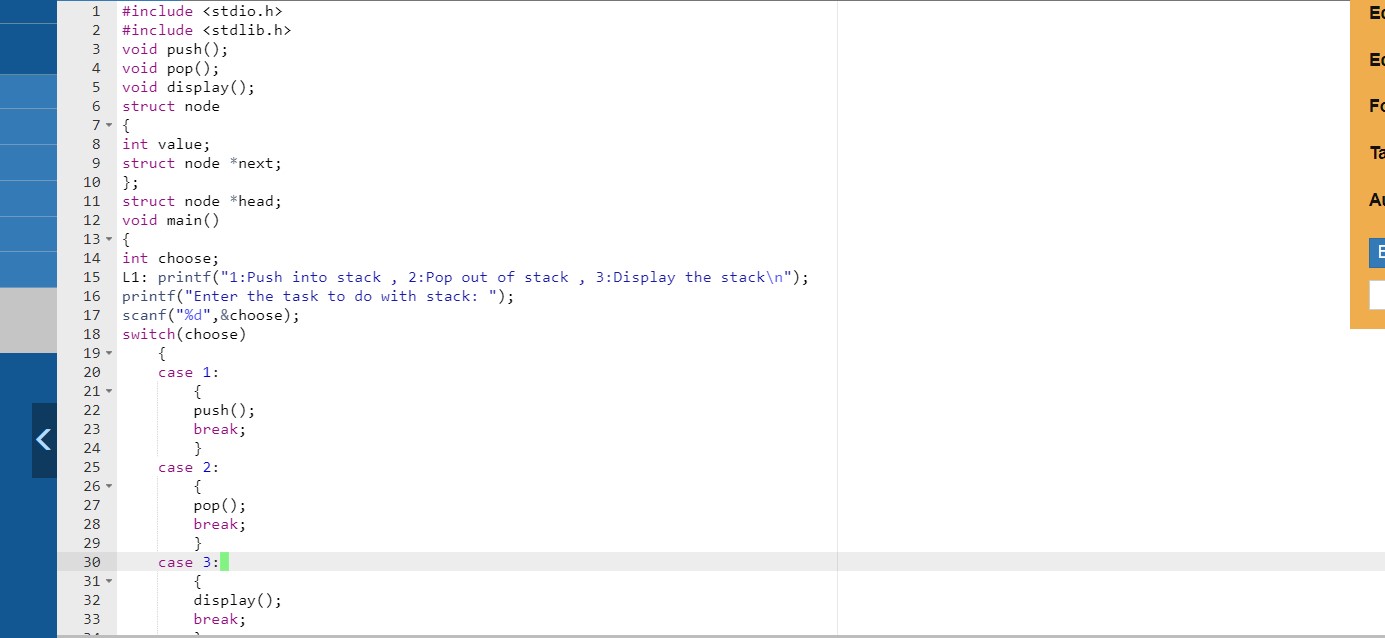
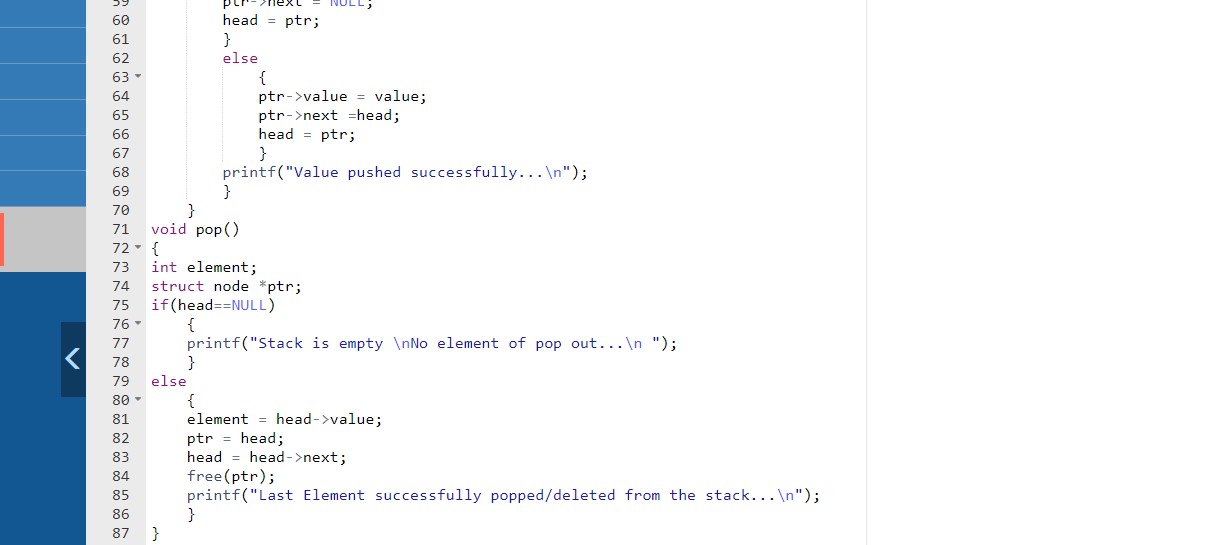
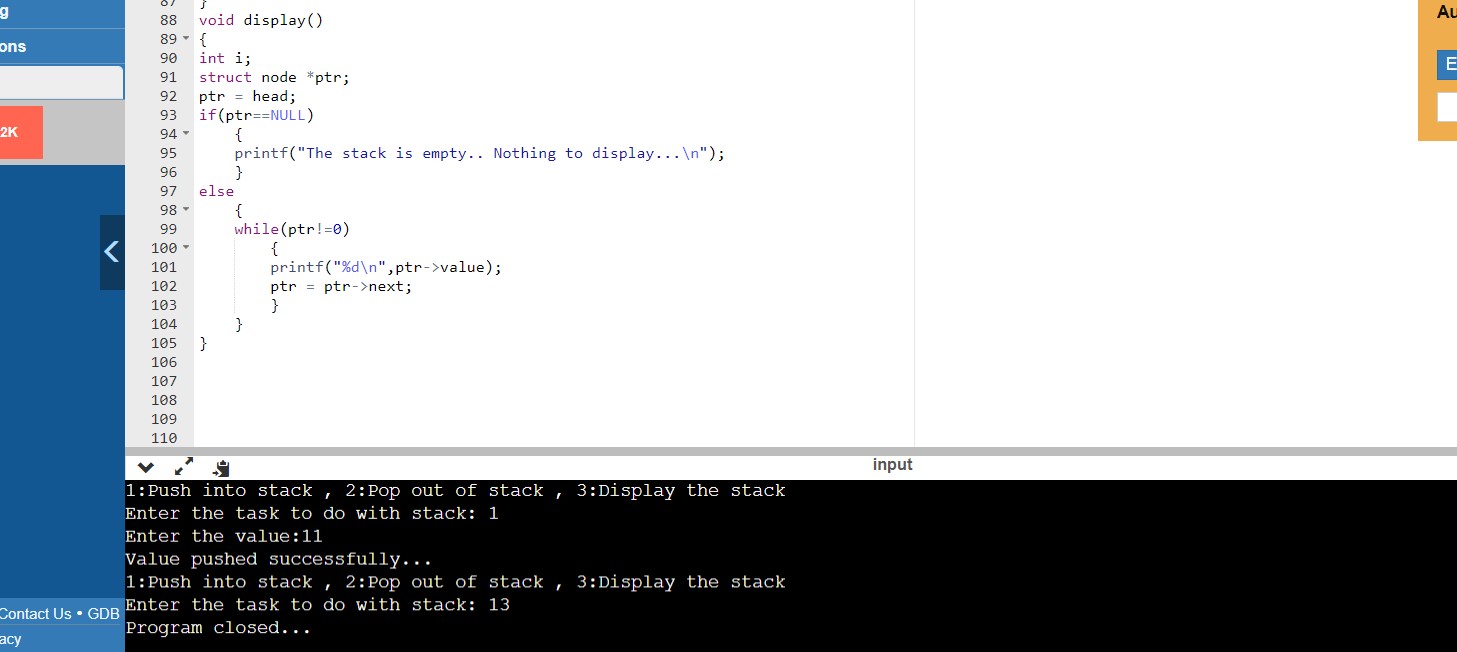
printf("%d\n",ptr->value);

ptr = ptr->next;

}

}

}



Implement a single Linked List data structure and its operations like insert and delete in the beginning/end and nth position of the list, and display the items stored in the linked list

//insertion in beginning

#include <stdio.h>

#include <stdlib.h>

void insertbegin();

void display();

struct node

{

int no;

struct node \*next;

}\*new, \*first=0, \*ptr;

int main()

{

insertbegin();

display();

return 0;

}

void insertbegin()

{

int choice = 1;

while(choice == 1)

{

new = (struct node\*) malloc(sizeof(struct node));

printf("Enter a number: ");

scanf("%d", &new->no);

new->next = first;

first = new;

printf("Enter '1' to insert another block in the end of the list\n");

scanf("%d",&choice);

}

}

void display()

{

if(first==0)

{

printf("List is empty...");

}

else

{

for(ptr=first;ptr!=0;ptr=ptr->next)

{

printf("Data: %d\n",ptr->no);

}

}

}

//insertion in middle

#include <stdio.h>

#include <stdlib.h>

struct node

{

int no;

struct node \*next;

}\*new, \*first=0,\*ptr, \*temp;

void insertend();

void insertbegin();

void insertmiddle();

void createblock();

void display();

int count=0;

int main()

{

int choice;

L1 : printf("1:Insert in Beginning , 2:Insert at end , 3:Insert in middle ,

4:Display, Any: Exit\n");

printf("Enter your choice: ");

scanf("%d",&choice);

switch(choice)

{

case 1:

insertbegin();

break;

case 2:

insertend();

break;

case 3:

insertmiddle();

break;

case 4:

display();

break;

default:

printf("close program...\n");

exit(EXIT\_SUCCESS);

}

goto L1;

return 0;

}

void createblock()

{

new = (struct node\*)malloc(sizeof(struct node));

printf("Enter the data: ");

scanf("%d",&new->no);

new->next=0;

count++;

}

void insertend()

{

createblock();

if(first==0)

{

first=new;

}

else

{

for(ptr=first; ptr!=0; ptr=ptr->next);

ptr->next=new;

}

}

void insertbegin()

{

createblock();

if(first==0)

{

first = new;

}

else

{

new->next=first;

first = new;

}

}

void display()

{

if(first==0)

{

printf("The linked list is empty...");

}

else

{

printf("List %d elements: ", count);

for(ptr=first;ptr!=0;ptr=ptr->next)

{

printf("%d ",ptr->no);

}

printf("\n");

}

}

void insertmiddle()

{

int pos;

printf("Enter the position: ");

scanf("%d ", &pos);

if(pos<1)

{

printf("Wrong input..\n");

}

else if (pos>count+1)

{

printf("Wrong input..\n");

}

else if (pos==1)

{

insertbegin();

}

else

{

int i=1;

for(ptr=first;i<pos-1;ptr=ptr->next,i++);

if(ptr->next==0)

{

insertend();

}

else

{

createblock();

temp = ptr->next;

ptr->next = new;

new->next = temp;

}

}

}

//insertion in end

#include <stdio.h>

#include <stdlib.h>

void insertend();

void display();

struct node

{

int no;

struct node \*next;

}\*new, \*first=0, \*ptr;

int main()

{

insertend();

display();

return 0;

}

void insertend()

{

int choice = 1;

while(choice == 1)

{

new = (struct node\*) malloc(sizeof(struct node));

printf("Enter a number: ");

scanf("%d", &new->no);

new->next = 0;

if(first==0)

{

first = new;

}

else

{

for(ptr=first;ptr->next != 0;ptr=ptr->next);

ptr->next = new;

}

printf("Enter '1' to insert another block in the end

of the list\n");

scanf("%d",&choice);

}

}

void display()

{

if(first==0)

{

printf("List is empty...");

}

else

{

for(ptr=first;ptr!=0;ptr=ptr->next)

{

printf("Data: %d\n",ptr->no);

}

}

}

//deletion in begging, middle and at end

#include <stdio.h>

#include <stdlib.h>

struct node

{

int no;

struct node \*next;

}\*new, \*first=0,\*ptr, \*temp;

void insertend();

void insertbegin();

void insertmiddle();

void deleteend();

void deletebegin();

void deletemiddle();

void createblock();

void display();

int count=0;

int main()

{

int choice;

L1 : printf("1:Insert in Beginning , 2:Insert at end , 3:Insert in middle ,

4:Display, Any: Exit\n");

printf("5:Delete in Beginning , 6:delete at end , 7:Delete in middle ,

4:Display, Any: Exit\n");

printf("Enter your choice: ");

scanf("%d",&choice);

switch(choice)

{

case 1:

insertbegin();

break;

case 2:

insertend();

break;

case 3:

insertmiddle();

break;

case 4:

display();

break;

case 5:

deletebegin();

break;

case 6:

deleteend();

break;

case 7:

deletemiddle();

break;

default:

printf("close program...\n");

exit(EXIT\_SUCCESS);

}

goto L1;

return 0;

}

void createblock()

{

new = (struct node\*)malloc(sizeof(struct node));

printf("Enter the data: ");

scanf("%d",&new->no);

new->next=0;

count++;

}

void insertend()

{

createblock();

if(first==0)

{

first=new;

}

else

{

for(ptr=first; ptr!=0; ptr=ptr->next);

ptr->next=new;

}

}

void insertbegin()

{

createblock();

if(first==0)

{

first = new;

}

else

{

new->next=first;

first = new;

}

}

void display()

{

if(first==0)

{

printf("\nThe linked list is empty...\n");

}

else

{

printf("List %d elements: ", count);

for(ptr=first;ptr!=0;ptr=ptr->next)

{

printf("%d ",ptr->no);

}

printf("\n");

}

}

void insertmiddle()

{

int pos;

printf("Enter the position: ");

scanf("%d ", &pos);

if(pos<1)

{

printf("Wrong input..\n");

}

else if (pos>count+1)

{

printf("Wrong input..\n");

}

else if (pos==1)

{

insertbegin();

}

else

{

int i=1;

for(ptr=first;i<pos-1;ptr=ptr->next,i++);

if(ptr->next==0)

{

insertend();

}

else

{

createblock();

//linking

temp = ptr->next;

ptr->next = new;

new->next = temp;

}

}

}

void deletebegin()

{

//for empty linked list

if(first==0)

{

printf("Linked list has no elements..\n");

}

//one element in the linked list

else if(first->next==0)

{

printf("Data is deleted node: %d\n",first->no);

free(first);

first=0;

count=0;

}

else

{

printf("Data is deleted node: %d\n",first->no);

temp = first->next;

free(first);

first = temp;

count = count-1;

}

}

void deleteend()

{

//for empty linked list

if(first==0)

{

printf("Linked list has no elements..\n");

}

//one element in the linked list

else if(first->next==0)

{

printf("Data is deleted node: %d\n",first->no);

free(first);

first=0;

count=0;

}

//more than one elements in the linked list

else

{

for (ptr=first; ptr->next->next !=0; ptr=ptr->next);

printf("Data is deleted node: %d\n",ptr->next->no);

free(ptr->next);

ptr->next=0;

count = count-1;

}

}

void deletemiddle()

{

int pos;

printf("Enter the position: ");

scanf("%d ", &pos);

if(pos<1)

{

printf("wrong input...\n");

}

else if(pos>count)

{

printf("Wrong input...\n");

}

else if (pos==1)

{

deletebegin();

}

else if (pos==count)

{

deleteend();

}

else

{

int i=1;

for(ptr=first;i<pos-1;ptr = ptr->next,i++);

printf("Data is deleted node: %d\n",ptr->next->no);

temp = ptr->next;

ptr->next = ptr->next->next;

free(temp);

count = count - 1;

}

}

**Add two polynomials using linked list**

#include <stdio.h>

#include <stdlib.h>

struct node

{

float coeffnt;

int expont;

struct node\* link;

};

struct node\* insert(struct node\* head, float coeff, int expo)

{

struct node\* temp;

struct node\* newpoly = malloc(sizeof(struct node));

newpoly->coeffnt = coeff;

newpoly->expont = expo;

newpoly->link = NULL;

if(head == NULL || expo> head->expont)

{

newpoly->link = head;

head = newpoly;

}

else

{

temp = head;

while(temp->link != NULL && temp->link->expont >= expo)

{

temp = temp->link;

}

newpoly->link = temp->link;

temp->link = newpoly;

}

return head;

}

struct node\* create(struct node\* head)

{

int n,i;

float coeffnt;

int expont;

printf("Enter the number of terms: ");

scanf("%d",&n);

for(i=0;i<n;i++)

{

printf("Enter the coefficient for term %d: ",i+1);

scanf("%f",&coeffnt);

printf("Enter the exponent for term %d: ",i+1);

scanf("%d",&expont);

head = insert(head, coeffnt, expont);

}

return head;

}

void display(struct node\* head)

{

if(head == NULL)

{

printf("No polynomial available..\n");

}

else

{

struct node\* temp = head;

while(temp!=NULL)

{

printf("(%.1fx^%d)",temp->coeffnt,temp->expont);

temp = temp->link;

if(temp!=NULL)

{

printf("+");

}

else

{

printf("\n");

}

}

}

}

void poly\_add(struct node\* head1, struct node\* head2)

{

struct node\* ptr1 = head1;

struct node\* ptr2 = head2;

struct node\* head3 = NULL;

while(ptr1!=NULL && ptr2!=NULL)

{

if(ptr1->expont == ptr2->expont)

{

head3 = insert(head3, ptr1->coeffnt,ptr1->expont);

ptr1 = ptr1->link;

ptr2 = ptr2->link;

}

else if(ptr1->expont > ptr2->expont)

{

head3 = insert(head3, ptr1->coeffnt,ptr1->expont);

ptr1 = ptr1->link;

}

else if(ptr1->expont < ptr2->expont)

{

head3 = insert(head3, ptr2->coeffnt,ptr2->expont);

ptr2 = ptr2->link;

}

while(ptr1!=NULL)

{

head3 = insert(head3, ptr1->coeffnt,ptr1->expont);

ptr1 = ptr1->link;

}

while(ptr2!=NULL)

{

head3 = insert(head3, ptr2->coeffnt,ptr2->expont);

ptr2 = ptr2->link;

}

printf("The added polynomial is: ");

display(head3);

}

}

int main()

{

struct node\* head1 = NULL;

struct node\* head2 = NULL;

printf("Enter the first polynomial\n");

head1 = create(head1);

printf("Enter the second polynomial\n");

head2 = create(head2);

poly\_add(head1,head2);

return 0;

}