Assignment 8

Q)

Code:

#include <stdio.h>

#include <stdlib.h>

struct node

{

int data;

struct node \*next;

};

struct node \*head=NULL;

struct node \*tail=NULL;

void insert();

void display();

int count(int ele);

int main()

{

int ele;

int n;

printf("\n Enter the no of elements that you wan to insert in the linked list: ");

scanf("%d",&n);

int i;

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

{

insert();

}

printf("\n The elements present in the linked list are: ");

display();

printf("\n Enter the element whoe occurance has to be counted in the linked list: ");

scanf("%d",&ele);

int occurance=count(ele);

printf("\n %d has occured %d times in the linked list.",ele,occurance);

return 0;

}

void insert()

{

//struct node \*tail=head;

int data;

printf("\n Enter the element in the linked list");

scanf("%d",&data);

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

newnode->data=data;

newnode->next=NULL;

if(head==NULL)

{

head=tail=newnode;

}

else

{

tail->next=newnode;

tail=tail->next;

}

}

void display()

{

struct node \*temp=head;

printf("\n");

while(temp!=NULL)

{

printf("%d ",temp->data);

temp=temp->next;

}

}

int count(int ele)

{

int ctr=0;

struct node \*temp=head;

while(temp!=NULL)

{

if(temp->data==ele)

{

ctr++;

}

temp=temp->next;

}

return ctr;

}

Output:

 Enter the no of elements that you wan to insert in the linked list: 9

 Enter the element in the linked list2

 Enter the element in the linked list12

 Enter the element in the linked list12

 Enter the element in the linked list34

 Enter the element in the linked list56

 Enter the element in the linked list18

 Enter the element in the linked list23

 Enter the element in the linked list45

 Enter the element in the linked list1

 The elements present in the linked list are:

2 12 12 34 56 18 23 45 1

 Enter the element whoe occurance has to be counted in the linked list: 12

 12 has occured 2 times in the linked list.

**...Program finished with exit code 0**

**Press ENTER to exit console.**

Q)

Code:

#include <stdio.h>

#include <stdlib.h>

struct node

{

int data;

struct node \*next;

};

struct node \*head=NULL;

struct node \*tail=NULL;

void insert();

void display();

struct node \*reverse();

int main()

{

//int ele;

int n;

printf("\n Enter the no of elements that you wan to insert in the linked list: ");

scanf("%d",&n);

int i;

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

{

insert();

}

printf("\n The elements present in the linked list are: ");

display();

printf("\n The reversed linked list is: ");

head=reverse();

display();

return 0;

}

void insert()

{

//struct node \*tail=head;

int data;

printf("\n Enter the element in the linked list");

scanf("%d",&data);

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

newnode->data=data;

newnode->next=NULL;

if(head==NULL)

{

head=tail=newnode;

}

else

{

tail->next=newnode;

tail=tail->next;

}

}

void display()

{

struct node \*temp=head;

printf("\n");

while(temp!=NULL)

{

printf("%d ",temp->data);

temp=temp->next;

}

}

struct node \*reverse()

{

struct node \*curr=head;

struct node \*prev=NULL;

struct node \*n=curr->next;

while(curr!=NULL)

{

curr->next=prev;

prev=curr;

curr=n;

if(curr)

n=curr->next;

}

return prev;

}

Output:

Enter the no of elements that you wan to insert in the linked list: 9

 Enter the element in the linked list23

 Enter the element in the linked list45

 Enter the element in the linked list12

 Enter the element in the linked list34

 Enter the element in the linked list56

 Enter the element in the linked list76

 Enter the element in the linked list90

 Enter the element in the linked list12

 Enter the element in the linked list34

 The elements present in the linked list are:

23 45 12 34 56 76 90 12 34

 The reversed linked list is:

34 12 90 76 56 34 12 45 23

**...Program finished with exit code 0**

**Press ENTER to exit console.**

Q3)

Code:

//Stack Implementation Using Linked List

#include <stdio.h>

#include <stdlib.h>

struct stack

{

int data;

struct stack \*next;

};

struct stack \*top=NULL;

//struct node \*tail=NULL;

void push();

void pop();

void peek();

void display();

//struct node \*reverse();

int main()

{

int choice;

while(1)

{

printf("\n Main Menu");

printf("\n 1.Push");

printf("\n 2.Pop");

printf("\n 3.Peek");

printf("\n 4.Display");

printf("\n 5.Exit");

printf("\n Enter your choice: ");

scanf("%d",&choice);

switch(choice)

{

case 1:push();

display();

break;

case 2:pop();

display();

break;

case 3:peek();

break;

case 4:display();

break;

case 5:exit(0);

break;

default: printf("Invalid Entry");

}

}

return 0;

}

void push()

{

//struct node \*tail=head;

int data;

printf("\n Enter the element in the linked list.");

scanf("%d",&data);

struct stack \*newele=(struct stack\*)malloc(sizeof(struct stack));

newele->data=data;

newele->next=top;

top=newele;

}

void pop()

{

if(top==NULL)

{

printf("\n Stack is overflowing.");

return;

}

struct stack \*temp=top;

top=temp->next;

temp->next=NULL;

free(temp);

}

void peek()

{

if(top==NULL)

{

printf("\n Stack is overflowing.");

return;

}

printf("\n The element present on the top of the stack is: %d",top->data);

}

void display()

{

if(top==NULL)

{

printf("\n Stack is overflowing.");

return;

}

struct stack \*temp=top;

printf("\n");

while(temp!=NULL)

{

printf("%d ",temp->data);

temp=temp->next;

}

}

Output:

 Main Menu

 1.Push

 2.Pop

 3.Peek

 4.Display

 5.Exit

 Enter your choice: 1

 Enter the element in the linked list.23

23

 Main Menu

 1.Push

 2.Pop

 3.Peek

 4.Display

 5.Exit

 Enter your choice: 1

 Enter the element in the linked list.23

23 23

 Main Menu

 1.Push

 2.Pop

 3.Peek

 4.Display

 5.Exit

 Enter your choice: 1

 Enter the element in the linked list.345

345 23 23

 Main Menu

 1.Push

 2.Pop

 3.Peek

 4.Display

 5.Exit

 Enter your choice: 1

 Enter the element in the linked list.56

56 345 23 23

 Main Menu

 1.Push

 2.Pop

 3.Peek

 4.Display

 5.Exit

 Enter your choice: 1

 Enter the element in the linked list.94

94 56 345 23 23

 Main Menu

 1.Push

 2.Pop

 3.Peek

 4.Display

 5.Exit

 Enter your choice: 1

 Enter the element in the linked list.48

48 94 56 345 23 23

 Main Menu

 1.Push

 2.Pop

 3.Peek

 4.Display

 5.Exit

 Enter your choice: 1

 Enter the element in the linked list.721

721 48 94 56 345 23 23

 Main Menu

 1.Push

 2.Pop

 3.Peek

 4.Display

 5.Exit

 Enter your choice: 2

48 94 56 345 23 23

 Main Menu

 1.Push

 2.Pop

 3.Peek

 4.Display

 5.Exit

 Enter your choice: 2

94 56 345 23 23

 Main Menu

 1.Push

 2.Pop

 3.Peek

 4.Display

 5.Exit

 Enter your choice: 3

 The element present on the top of the stack is: 94

 Main Menu

 1.Push

 2.Pop

 3.Peek

 4.Display

 5.Exit

 Enter your choice: 4

94 56 345 23 23

 Main Menu

 1.Push

 2.Pop

 3.Peek

 4.Display

 5.Exit

 Enter your choice: 5

**...Program finished with exit code 0**

**Press ENTER to exit console.**

Q4)

Code:

//Queue Implementation Using Linked List

#include <stdio.h>

#include <stdlib.h>

struct queue

{

int data;

struct queue \*next;

};

struct queue \*front=NULL;

struct queue \*rear=NULL;

void Enqueue();

void Dequeue();

//void peek();

void display();

//struct node \*reverse();

int main()

{

int choice;

while(1)

{

printf("\n Main Menu");

printf("\n 1.Enqueue");

printf("\n 2.Dequeue");

printf("\n 3.Display");

printf("\n 4.Exit");

printf("\n Enter your choice: ");

scanf("%d",&choice);

switch(choice)

{

case 1:Enqueue();

display();

break;

case 2:Dequeue();

display();

break;

case 3:display();

break;

case 4:exit(0);

break;

default: printf("Invalid Entry");

}

}

return 0;

}

void Enqueue()

{

//struct node \*tail=head;

int data;

printf("\n Enter the element in the linked list.");

scanf("%d",&data);

struct queue \*newele=(struct queue\*)malloc(sizeof(struct queue));

newele->data=data;

newele->next=NULL;

if(front==NULL)

{

front=newele;

rear=newele;

}

else

{

//tail=head;

rear->next=newele;

rear=rear->next;

}

}

void Dequeue()

{

if(front==NULL)

{

printf("\n Queue is underflowing.");

return;

}

struct queue \*temp=front;

front=temp->next;

temp->next=NULL;

free(temp);

}

void display()

{

if(front==NULL)

{

printf("\n Queue is underflowing.");

return;

}

struct queue \*temp=front;

printf("\n");

while(temp!=NULL)

{

printf("%d ",temp->data);

temp=temp->next;

}

}

Output:

 Main Menu

 1.Enqueue

 2.Dequeue

 3.Display

 4.Exit

 Enter your choice: 1

 Enter the element in the linked list.2

2

 Main Menu

 1.Enqueue

 2.Dequeue

 3.Display

 4.Exit

 Enter your choice: 1

 Enter the element in the linked list.34

2 34

 Main Menu

 1.Enqueue

 2.Dequeue

 3.Display

 4.Exit

 Enter your choice: 1

 Enter the element in the linked list.498

2 34 498

 Main Menu

 1.Enqueue

 2.Dequeue

 3.Display

 4.Exit

 Enter your choice: 1

 Enter the element in the linked list.347

2 34 498 347

 Main Menu

 1.Enqueue

 2.Dequeue

 3.Display

 4.Exit

 Enter your choice: 1

 Enter the element in the linked list.267

2 34 498 347 267

 Main Menu

 1.Enqueue

 2.Dequeue

 3.Display

 4.Exit

 Enter your choice: 1

 Enter the element in the linked list.529

2 34 498 347 267 529

 Main Menu

 1.Enqueue

 2.Dequeue

 3.Display

 4.Exit

 Enter your choice: 1

 Enter the element in the linked list.54

2 34 498 347 267 529 54

 Main Menu

 1.Enqueue

 2.Dequeue

 3.Display

 4.Exit

 Enter your choice: 1

 Enter the element in the linked list.3

2 34 498 347 267 529 54 3

 Main Menu

 1.Enqueue

 2.Dequeue

 3.Display

 4.Exit

 Enter your choice: 2

34 498 347 267 529 54 3

 Main Menu

 1.Enqueue

 2.Dequeue

 3.Display

 4.Exit

 Enter your choice: 2

498 347 267 529 54 3

 Main Menu

 1.Enqueue

 2.Dequeue

 3.Display

 4.Exit

 Enter your choice: 3

498 347 267 529 54 3

 Main Menu

 1.Enqueue

 2.Dequeue

 3.Display

 4.Exit

 Enter your choice: 4

**...Program finished with exit code 0**

**Press ENTER to exit console.**

Q5)

Code:

#include <stdio.h>

#include <stdlib.h>

struct polynomial

{

int coefficient;

int degree;

struct polynomial \*next;

};

struct polynomial \*pt=NULL;

struct polynomial \*insert(struct polynomial \*p);

void display(struct polynomial \*p);

struct polynomial \*polysum(struct polynomial \*p1,struct polynomial \*p2);

int main()

{

int m,n;

struct polynomial \*p1=NULL;

struct polynomial \*p2=NULL;

struct polynomial \*p=NULL;

printf("\n Enter the no of terms of the first polynomial: ");

scanf("%d",&m);

printf("\n Enter the no of terms of the second polynomial: ");

scanf("%d",&n);

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

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

{

struct polynomial \*pa=insert(p1);

p1=pa;

}

pt=NULL;

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

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

{

struct polynomial \*pb=insert(p2);

p2=pb;

}

printf("\n The first polynomial is: p1(x)= ");

display(p1);

printf("\n The second polynomial is: p2(x)= ");

display(p2);

printf("\n The polynomial sum is: p(x)= ");

p=polysum(p1,p2);

display(p);

return 0;

}

struct polynomial \*insert(struct polynomial \*p)

{

int coefficient;

int degree;

printf("\n Enter the coefficient: ");

scanf("%d",&coefficient);

printf("\n Enter the degree: ");

scanf("%d",&degree);

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

newpoly->coefficient=coefficient;

newpoly->degree=degree;

newpoly->next=NULL;

if(p==NULL)

{

p=pt=newpoly;

}

else

{

pt->next=newpoly;

pt=newpoly;

}

return p;

}

void display(struct polynomial \*p)

{

struct polynomial \*temp=p;

//printf("\n");

while(temp->next!=NULL)

{

printf("%dx%d + ",temp->coefficient,temp->degree);

temp=temp->next;

}

printf("%dx%d",temp->coefficient,temp->degree);

}

struct polynomial \*polysum(struct polynomial \*p1,struct polynomial \*p2)

{

struct polynomial \*p=NULL;

struct polynomial \*t=NULL;

struct polynomial \*t1=p1;

struct polynomial \*t2=p2;

while(t1!=NULL && t2!=NULL)

{

if(t1->degree<t2->degree)

{

if(p==NULL)

{

p=t=t2;

}

else

{

t->next=t2;

t=t2;

}

t2=t2->next;

}

else if(t1->degree>t2->degree)

{

if(p==NULL)

{

p=t=t1;

}

else

{

t->next=t1;

t=t1;

}

t1=t1->next;

}

else

{

struct polynomial \*n=(struct polynomial\*)malloc(sizeof(struct polynomial));

n->degree=t1->degree;

n->coefficient=t1->coefficient+t2->coefficient;

n->next=NULL;

if(p==NULL)

{

p=t=n;

}

else

{

t->next=n;

t=n;

}

t1=t1->next;

t2=t2->next;

}

}

if(t1==NULL)

{

while(t2!=NULL)

{

t->next=t2;

t=t2;

t2=t2->next;

}

}

else

{

while(t1!=NULL)

{

t->next=t1;

t=t1;

t1=t1->next;

}

}

return p;

}

Output:

 Enter the no of terms of the first polynomial: 5

 Enter the no of terms of the second polynomial: 3

 Enter the first polynomial:

 Enter the coefficient: 4

 Enter the degree: 5

 Enter the coefficient: 5

 Enter the degree: 4

 Enter the coefficient: 2

 Enter the degree: 3

 Enter the coefficient: 3

 Enter the degree: 2

 Enter the coefficient: 7

 Enter the degree: 1

 Enter the second polynomial:

 Enter the coefficient: 9

 Enter the degree: 6

 Enter the coefficient: 6

 Enter the degree: 4

 Enter the coefficient: 3

 Enter the degree: 2

 The first polynomial is: p1(x)= 4x5 + 5x4 + 2x3 + 3x2 + 7x1

 The second polynomial is: p2(x)= 9x6 + 6x4 + 3x2

 The polynomial sum is: p(x)= 9x6 + 4x5 + 11x4 + 2x3 + 6x2 + 7x1

**...Program finished with exit code 0**

**Press ENTER to exit console.**