DAY_21_ASSIGNMENT

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

struct Node
{
   int data;
   struct Node*next;
}*head=NULL;
void display(struct Node*p);
void rdisplay(struct Node*p);
void revdisplay(struct Node*p);
int Ncount(struct Node*p);
int rcount(struct Node*p);
int sumof(struct Node*p);
int sumoff(struct Node*p);
int maxelement(struct Node*p);
int remax(struct Node*p);
struct Node* Lsearch(struct Node *p,int key);
void insert(struct Node*p,int index,int val);
void create(int *,int n);


int main(){
   struct Node *temps;
   int  A[]={10,20,30,40,50};
   create(A,5);
   // struct Node *first;

   // first =(struct Node*)malloc(sizeof(struct Node));
   // first->data=10;
   // struct Node *second;
   // second =(struct Node*)malloc(sizeof(struct Node));
   // second->data=40;
   //  struct Node *third;
   // third =(struct Node*)malloc(sizeof(struct Node));
   // third->data=30;

   // first->next=second;
   // second->next=third;
   // third->next=NULL;
```

```
        // display(first);
        // printf("\n");
        // printf("using recursion\n");
        // rdisplay(first);
        // printf("\n");
        // revdisplay(first);
        // printf("\n");
        // int n=Ncount(first);
        // printf("the total number of node is %d",n);
        // printf("\n");
        // int m= rcount(first);
        // printf("the total number of node is %d\n",m);
        // int sum =sumof(first);
        // printf("the sum of all the element %d\n",sum);
        // int res=sumoff(first);
        // printf("the sum of using recusion is %d\n",res);
        // int max=maxelement(first);
        // printf("the maximum element is %d\n",max);
        // int re=remax(first);
        // printf("the maximum element using recursion%d",re);
        // temps = Lsearch(first, 10);
        // if (temps != NULL) {
        //     printf("Element %d found at address: \n", temps->data);
        // } else {
        //     printf("Element not found in the list.\n");
        // }
        // insert(first,1,5);
        display(head);
        printf("\n");
        create(A,5);
        display(head);

}
void display(struct Node*p){
    //p=0x200;
    while(p!=NULL){

        printf("%d->",p->data);
        p=p->next;
        //p = NULL
    }
}

// display all element using recursion
```

```
void rdisplay(struct Node*p){
    if(p!=NULL){

        printf("%d->",p->data);
         rdisplay(p->next);

    }

}
void revdisplay(struct Node*p){
    if(p!=NULL){
        rdisplay(p->next);
        printf("%d->",p->data);

    }

}
int Ncount(struct Node*p){
    int c=0;
    while (p!=NULL)
    {
        c++;
        p=p->next;
    }
    return c;


}
// implement this count using recursion allso

int rcount(struct Node*p){
    if(p==NULL){
        return 0;
    }
    else{
        return rcount(p->next)+1;
        //7
    }
}

// sum of all the elements
int sumof(struct Node*p){
    int total=0;
    while(p!=NULL){
```

```
        total=total+p->data;
        p=p->next;
    }
    return total;
}
// sum of all element using recursion

int sumoff(struct Node*p){
    if(p==NULL){
        return 0;
    }
    else{
        return p->data+sumoff(p->next);
    }
}
// maximum element
int maxelement(struct Node*p){
    int m=-32768;
    while(p!=NULL){
        if(p->data>m){
            m=p->data;
        }
        p=p->next;
    }
    return m;
}
// using recursion
int remax(struct Node*p){
    int x=0;
    if(p==NULL){
        return -32768;
    }
    else{
        x=remax(p->next);
        if(x>p->data){  //8 3 7 12 9
            return x; //3>8
        }
        else{
            return p->data;
        }

    }
}
// searching a element from a linkedlist
```

```
struct Node* Lsearch(struct Node *p,int key){

while (p!=NULL)
{
   if(key==p->data){
    return p;
   }
   p=p->next;
}
return NULL;

}
void insert(struct Node*p,int index,int val){
    struct Node *t;
    int i;
    if(index<0||index>Ncount(p)){
        printf("invalid position");
    }
    t=(struct Node*)malloc(sizeof(struct Node));
    t->data=val;

    if(index==0){
        t->next=head;
        head=t;
        //p->next=t;
    }
    else{
        for(int i=0;i<index-1;i++){
            p=p->next;
        }
        t->next=p->next;
        p->next=t;

    }
}
void create(int A[],int n){
    struct Node*new,*temp;//last mean temp
    head=(struct Node*)malloc(sizeof(struct Node));
    head->data=A[0];
    head->next=NULL;
    temp=head;
    for(int i=1;i<n;i++){
        new=(struct Node*)malloc(sizeof(struct Node));
        new->data=A[i];
```

```
        new->next=NULL;
        temp->next=new;
        temp=new;
    }
}
```