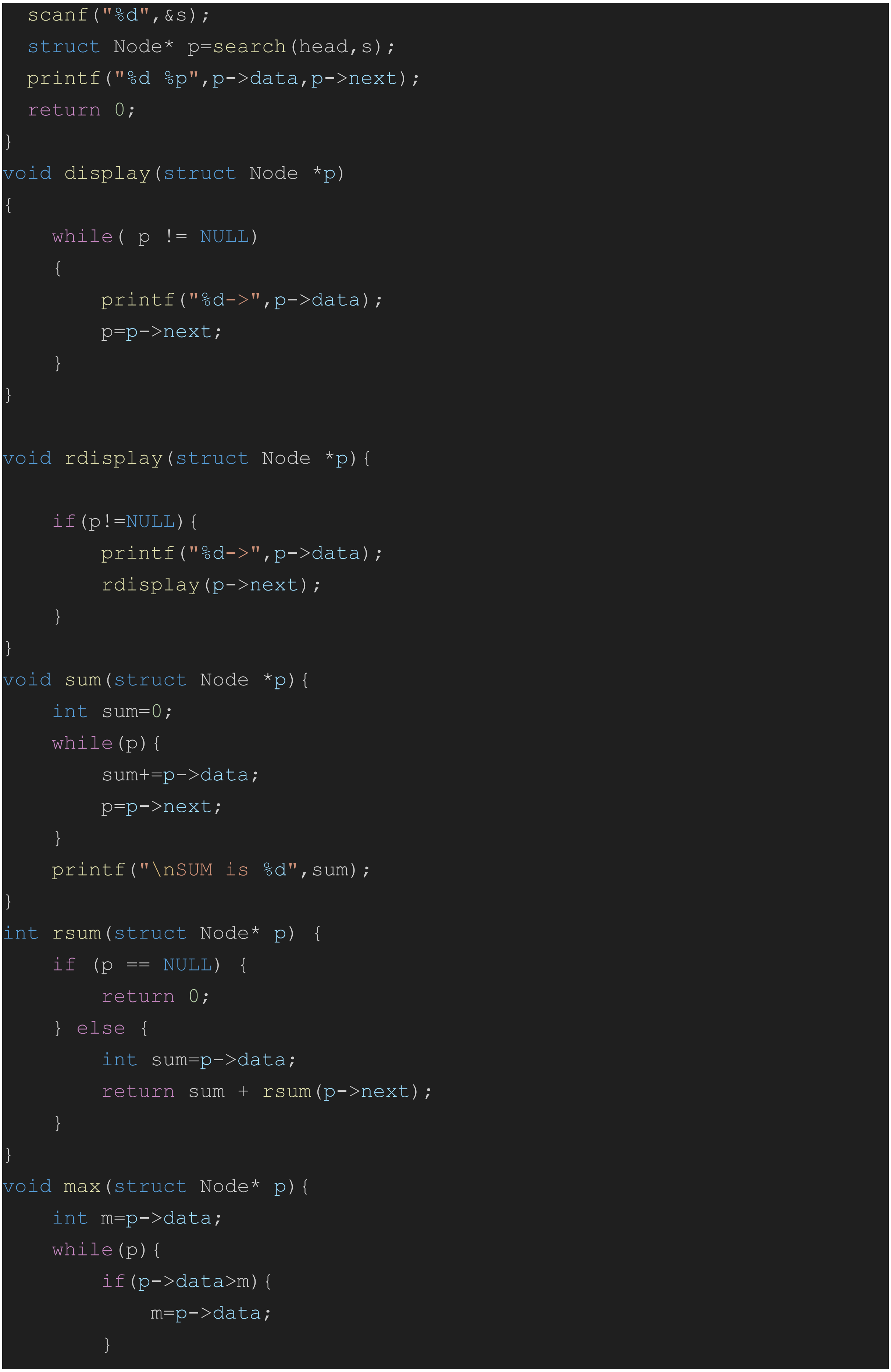
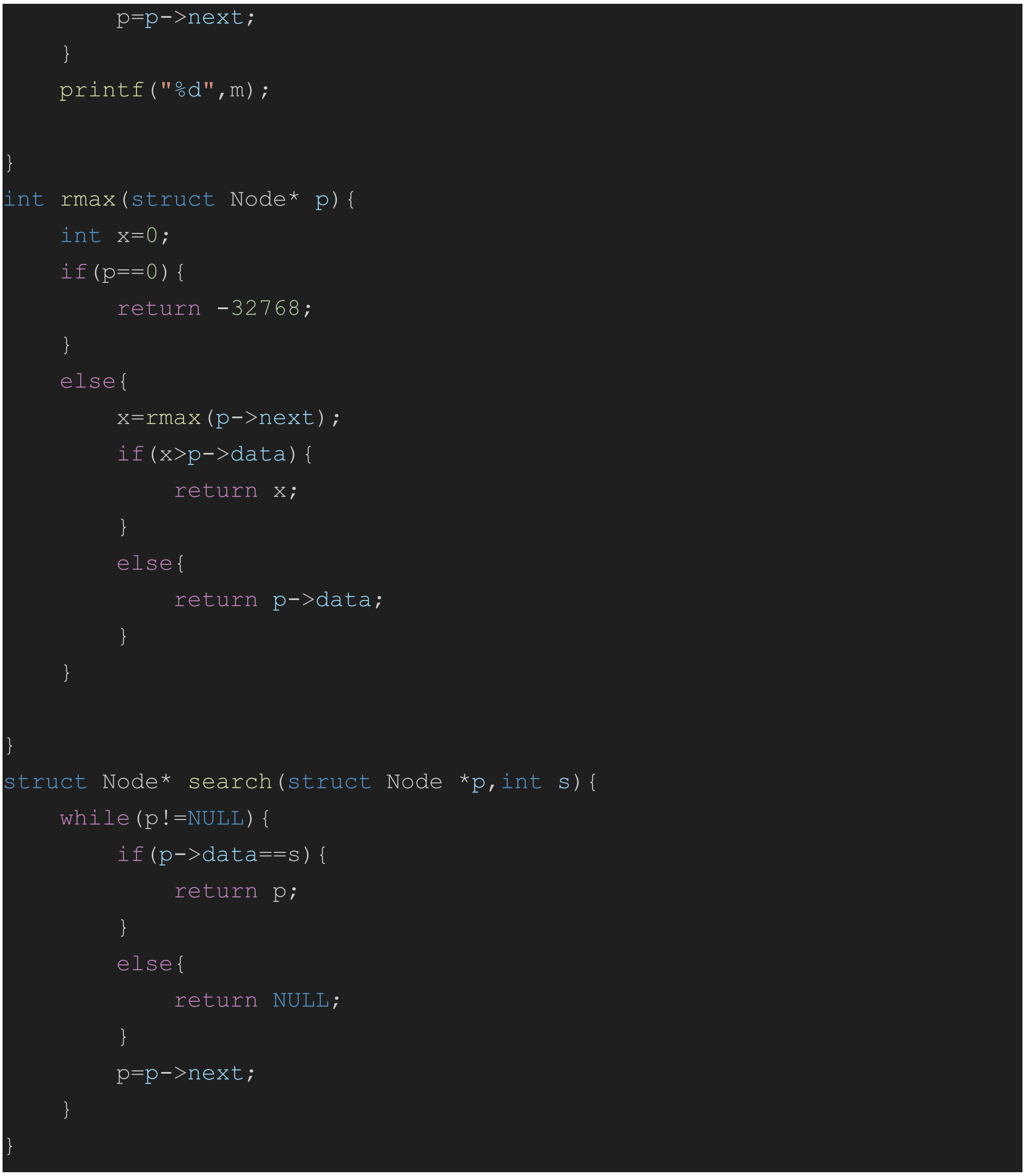
```
#include <stdio.h>
#include<stdlib.h>

struct Node
{
    int data;
    struct Node *next;
};

void display(struct Node*);
void rdisplay(struct Node*);
void sum(struct Node *);
int rsum(struct Node *);
void max(struct Node*);
int rmax(struct Node*);
struct Node* search(struct Node*,int);
int main()
{
  struct Node *head,*n1,*n2;
  head=(struct Node *)malloc(sizeof(struct Node));
  n1=(struct Node *)malloc(sizeof(struct Node));
  n2=(struct Node *)malloc(sizeof(struct Node));


  head->data=10;
  head->next=n1;
  n1->data=20;
  n1->next=n2;
  n2->data=40;
  n2->next=NULL;

  display(head);
  printf("\nRECURSION DISPLAY::\n");
  rdisplay(head);
  sum(head);
  int sum=rsum(head);
  printf("\nRECURSION SUM::%d\n",sum);
  printf("Maximum element::");
  max(head);
  int max=rmax(head);
  printf("\nMaximum using recursion::%d",max);
  int s;
  printf("\nEnter element to search::");
```



```
  scanf("%d",&s);
  struct Node* p=search(head,s);
  printf("%d %p",p->data,p->next);
  return 0;
}
void display(struct Node *p)
{
    while( p != NULL)
    {
        printf("%d->",p->data);
        p=p->next;
    }
}

void rdisplay(struct Node *p){

    if(p!=NULL){
        printf("%d->",p->data);
        rdisplay(p->next);
    }
}
void sum(struct Node *p){
    int sum=0;
    while(p){
        sum+=p->data;
        p=p->next;
    }
    printf("\nSUM is %d",sum);
}
int rsum(struct Node* p) {
    if (p == NULL) {
        return 0;
    } else {
        int sum=p->data;
        return sum + rsum(p->next);
    }
}
void max(struct Node* p){
    int m=p->data;
    while(p){
        if(p->data>m){
            m=p->data;
        }
```



```
        p=p->next;
    }
    printf("%d",m);

}
int rmax(struct Node* p){
    int x=0;
    if(p==0){
        return -32768;
    }
    else{
        x=rmax(p->next);
        if(x>p->data){
            return x;
        }
        else{
            return p->data;
        }
    }

}
struct Node* search(struct Node *p,int s){
    while(p!=NULL){
        if(p->data==s){
            return p;
        }
        else{
            return NULL;
        }
        p=p->next;
    }
}
```