Lab 7:--

Program:-

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

struct node
 {
   int info;
   struct node *link;
 };
typedef struct node *NODE;
NODE getnode()
{
NODE x;
x=(NODE)malloc(sizeof(struct node));
if(x==NULL)
 {
   printf("mem full\n");
   exit(0);
 }
 return x;
}
NODE insert_rear(NODE first,int item)
{
NODE temp,cur;
```

```
temp=getnode();
temp->info=item;
temp->link=NULL;
if(first==NULL)
  return temp;
cur=first;
while(cur->link!=NULL)
  cur=cur->link;
cur->link=temp;
return first;
}
void display(NODE first)
{
  NODE temp;
  if(first==NULL)
    printf("list empty");

  for(temp=first;temp!=NULL;temp=temp->link)
    {
    printf("%d\n",temp->info);
    }
}
NODE concat(NODE first,NODE second)
{
  NODE cur;
```

```
  if(first==NULL)
    return second;
  if(second==NULL)
    return first;
  cur=first;
  while(cur->link!=NULL)
    cur=cur->link;
  cur->link=second;
  return first;
}
NODE reverse(NODE first)
  {
  NODE cur,temp;
  cur=NULL;
  while(first!=NULL)
    {
      temp=first;
      first=first->link;
      temp->link=cur;
      cur=temp;
    }
  return cur;
}
```

```
NODE asc(NODE first)
{
        NODE prev=first;
        NODE cur=NULL;
                        int temp;

if(first== NULL) {
                  return 0;
                  }
else {
                 while(prev!= NULL) {

                  cur = prev->link;

                  while(cur!= NULL) {

                 if(prev->info > cur->info) {
                  temp = prev->info;
                  prev->info = cur->info;
                  cur->info = temp;
                    }
                    cur = cur->link;
                  }
                  prev= prev->link;
                                  }
```

```
                    }
                    return first;
                }


NODE des(NODE first)
{
        NODE prev=first;
        NODE cur=NULL;
                    int temp;

if(first==NULL) {
            return 0;
             }
            else {
               while(prev!= NULL) {

               cur = prev->link;

                 while(cur!= NULL) {

              if(prev->info < cur->info) {
                 temp = prev->info;
                 prev->info = cur->info;
                 cur->info = temp;
```

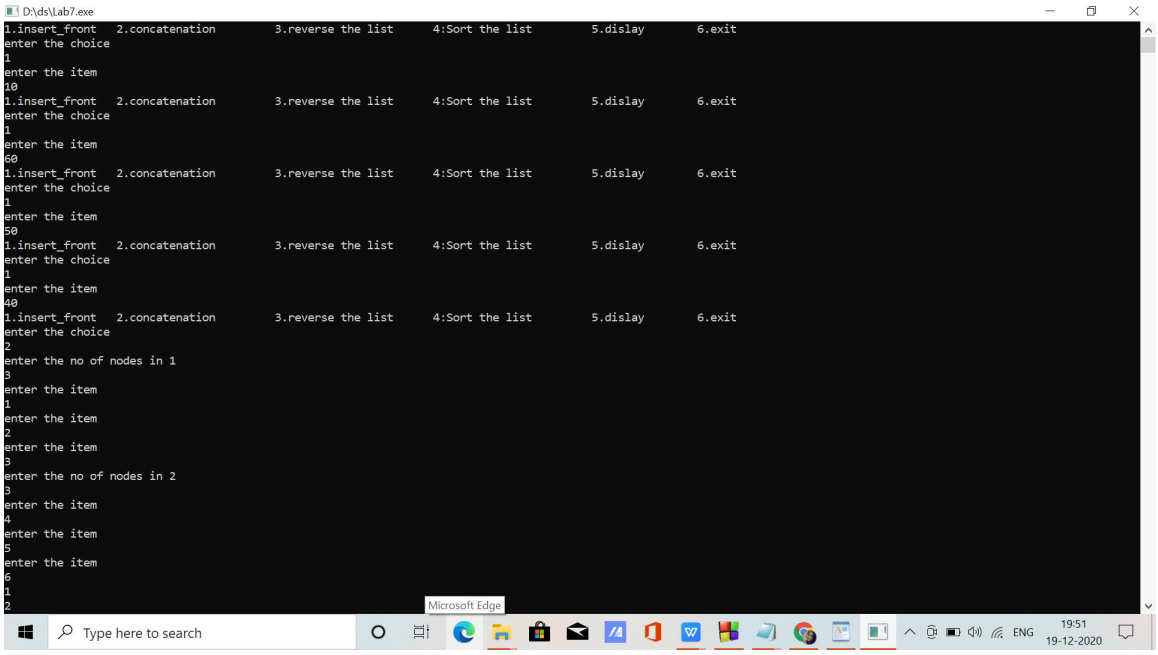
```
                    }
                    cur = cur->link;
                    }
                    prev= prev->link;
                     }
                     }
                     return first;
                }

void main()
{
int item,choice,pos,i,n,option;
NODE first=NULL,a,b;
for(;;)
{
printf("1.insert_front\t 2.concatenation\t 3.reverse the list\t 4:Sort the list\t 5.dislay\t 6.exit\n");
printf("enter the choice\n");
scanf("%d",&choice);
switch(choice)
 {
   case 1:printf("enter the item\n");
                    scanf("%d",&item);
                    first=insert_rear(first,item);
                    break;
   case 2:printf("enter the no of nodes in 1\n");
```

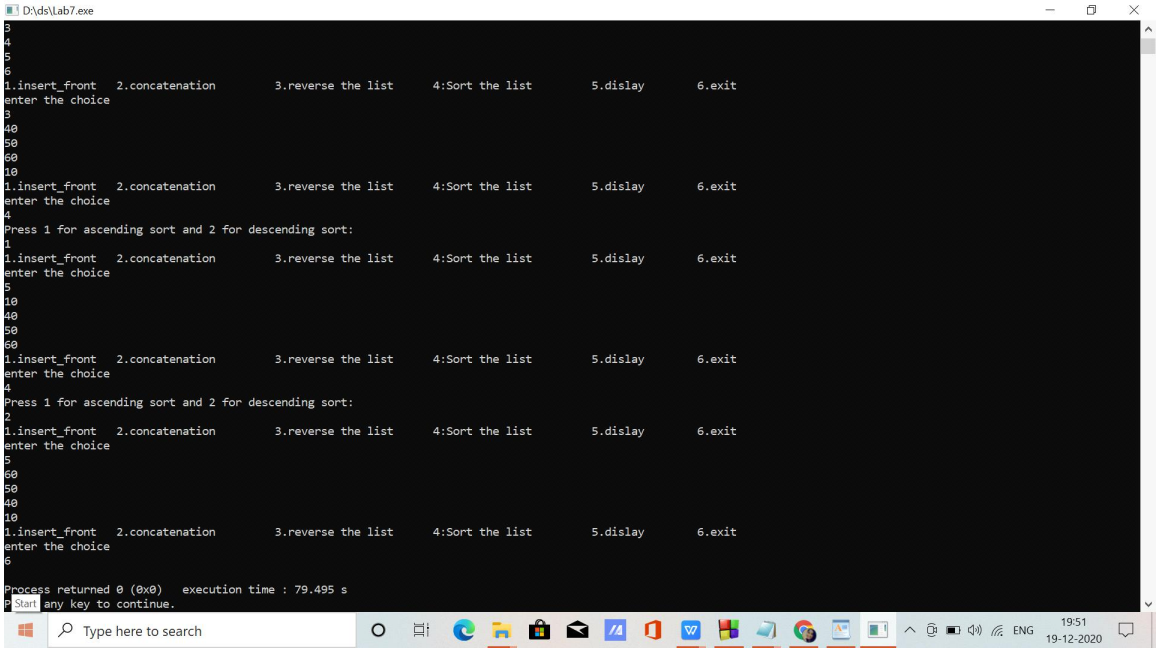
```
                    scanf("%d",&n);
                    a=NULL;
                    for(i=0;i<n;i++)
                     {
                       printf("enter the item\n");
                       scanf("%d",&item);
                       a=insert_rear(a,item);
                     }
                     printf("enter the no of nodes in 2\n");
                    scanf("%d",&n);
                    b=NULL;
                    for(i=0;i<n;i++)
                     {
                       printf("enter the item\n");
                       scanf("%d",&item);
                       b=insert_rear(b,item);
                     }
                     a=concat(a,b);
                     display(a);
                    break;
  case 3:first=reverse(first);
                    display(first);
                    break;
case 4:printf("Press 1 for ascending sort and 2 for descending sort:\n");
            scanf("%d",&option);
```

```
            if(option==1)
            first=asc(first);
            if(option==2)
            first=des(first);
            break;
    case 5:display(first);
                    break;
    default:exit(0);
    }
    }
    }
```

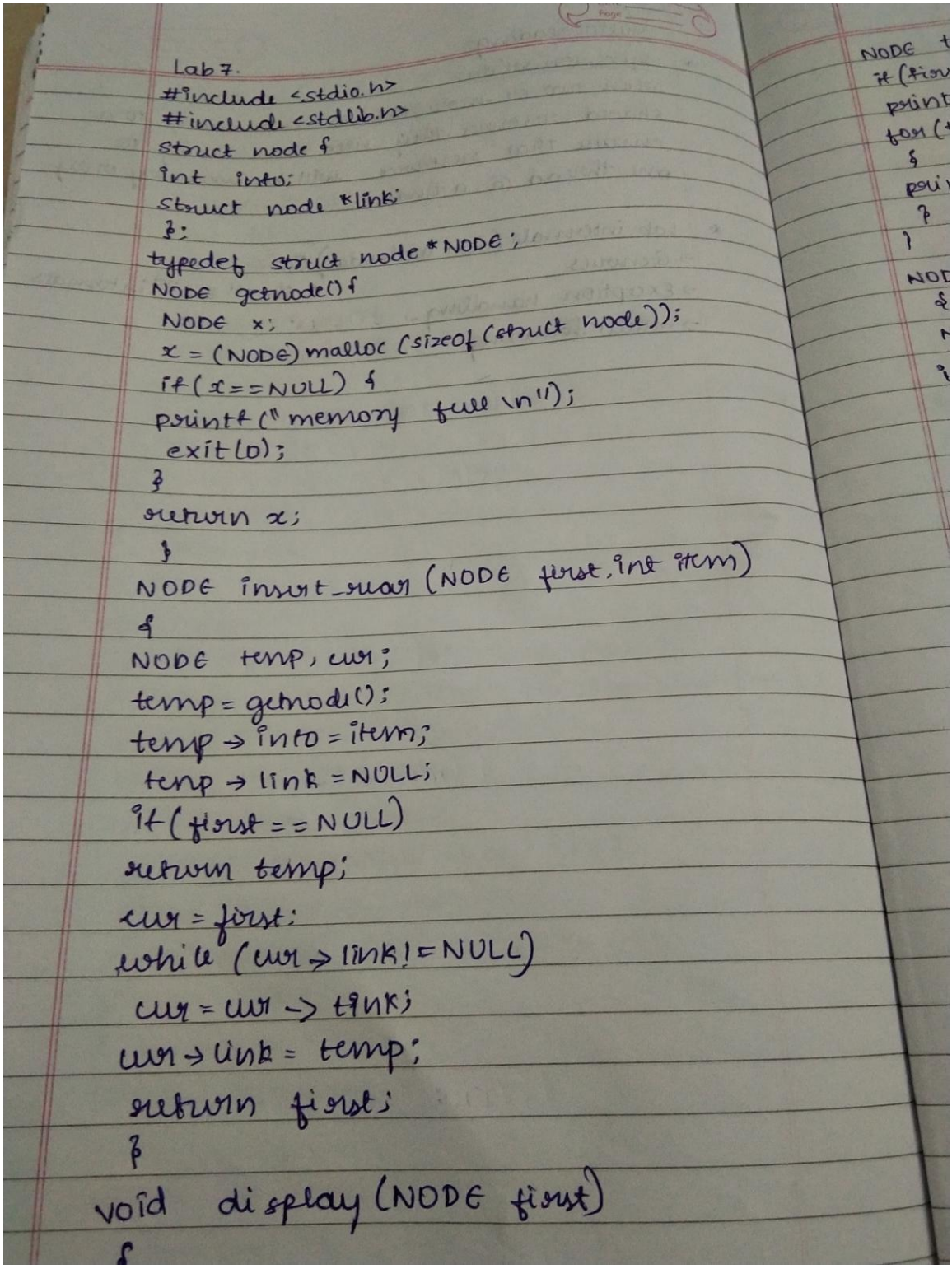
Output screenshot:-



Written Picture:-


D:\ds\Lab7.exe
3
4
5
6
1.insert_front 2.concatenation 3.reverse the list 4:Sort the list 5.dislay 6.exit
enter the choice
3
40
50
60
10
1.insert_front 2.concatenation 3.reverse the list 4:Sort the list 5.dislay 6.exit
enter the choice
4
Press 1 for ascending sort and 2 for descending sort:
1
1.insert_front 2.concatenation 3.reverse the list 4:Sort the list 5.dislay 6.exit
enter the choice
5
10
40
50
60
1.insert_front 2.concatenation 3.reverse the list 4:Sort the list 5.dislay 6.exit
enter the choice
4
Press 1 for ascending sort and 2 for descending sort:
2
1.insert_front 2.concatenation 3.reverse the list 4:Sort the list 5.dislay 6.exit
enter the choice
5
60
50
40
10
1.insert_front 2.concatenation 3.reverse the list 4:Sort the list 5.dislay 6.exit
enter the choice
6
Process returned 0 (0x0) execution time : 79.495 s
Press any key to continue.
Start
Type here to search
ENG
19:51
19-12-2020

Written:--



```
Lab 7.
#include <stdio.h>
#include <stdlib.h>
struct node {
int info;
struct node *link;
};
typedef struct node *NODE;
NODE getnode(){
 NODE x;
x = (NODE) malloc (sizeof (struct node));
if(x==NULL) {
printf("memory full \n");
 exit(0);
}
return x;
 }
NODE insert_rear (NODE first, int item)
{
NODE temp, cur;
temp = getnode();
temp -> info = item;
 temp -> link = NULL;
if(first==NULL)
return temp;
cur = first;
while (cur -> link!=NULL)
 cur = cur -> link;
cur -> link = temp;
return first;
}
void display (NODE first)
{
```

```
NODE asc(NODE first)
{ NODE prev = first;
  NODE cur = NULL;
  int temp;
if (first == NULL){
 return 0;
 }
else {
while (prev != NULL){
 cur = prev -> link;
 while (cur != NULL){
if (prev -> info > cur -> info){
temp = prev -> info;
prev -> info = cur -> info;
cur -> info = temp;
 }
 cur = cur -> link;
 }
prev = prev -> link; }
return first; }

NODE dsc(NODE first){
NODE prev = first;
 NODE cur = NULL;
int temp;
if (first == NULL){
 return 0;
 }
else { while (prev != NULL){
cur = prev -> link;
while (cur != NULL){
if (prev -> info < cur -> info){
```

```
temp = prev->info;
prev->info = cur->info;
cur->info = temp; }
cur = cur->link; }
prev = prev->link; }
}
return first;
}
void main() {
int item, choice, pos, i, n, option;
NODE first = NULL, a, b;
for(;;)
{
printf("1. insert-front \t 2. concatenation \t 3.
reverse the list \t 4: sort the list \t 5. display
\t 6. exit \n");
printf("enter the choice \n");
scanf("%d", &choice);
switch(choice)
{
case 1: printf("enter the item \n");
    scanf("%d", &item);
    first = insert_rear(first, item);
    break;
case 2: printf(" enter the no of nodes in 1 \n");
scanf("%d", &n);
  a = NULL;
for(i=0; i<n; i++) {
printf(" enter the item \n");
scanf("%d", &item);
a = insert_rear(a, item);
}
```

```
printf("enter the no of nodes in 2\n");
scanf("%d",&n);
b = NULL;
for (i=0;i<n;i++){
printf("enter the item\n");
scanf("%d",&item);
b = insert_rear(b,item); }
a = concat(a,b);
display(a);
break;
case 3: first = reverse(first);
    display(first); break;
case 4: prf("Press 1 for asc and 2 for desc\n");
  sf("%d",&option);
if(option==1)
 first = asc(first);
if(option==2)
first = dis(first);
 break;
case 5: display(first); break;
default: exit(0);
}
}
}
```