WEEK 9:DOUBLY LINKED LISTS

PROGRAM

```
#include<stdio.h>
#include<stdlib.h>
struct node
{
        int info;
        struct node *llink;
        struct node *rlink;
        };
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;
        }
void freenode(NODE x)
{
        free(x);
}
NODE dinsert_front(int item,NODE head)
{
NODE temp,cur;
temp=getnode();
```

```
temp->info=item;
cur=head->rlink;
head->rlink=temp;
temp->llink=head;
temp->rlink=cur;
cur->llink=temp;
return head;
}
NODE dinsert_rear(int item,NODE head)
{
NODE temp,cur;
temp=getnode();
temp->info=item;
cur=head->llink;
head->llink=temp;
temp->rlink=head;
temp->llink=cur;
cur->rlink=temp;
return head;
}
NODE ddelete_front(NODE head)
{
NODE cur,next;
if(head->rlink==head)
{
printf("Dq empty\n");
return head;
}
cur=head->rlink;
next=cur->rlink;
head->rlink=next;
```

```
next->llink=head;
printf("The node deleted is %d",cur->info);
freenode(cur);
return head;
}
NODE ddelete_rear(NODE head)
{
NODE cur,prev;
if(head->rlink==head)
{
printf("Dq empty\n");
return head;
}
cur=head->llink;
prev=cur->llink;
head->llink=prev;
prev->rlink=head;
printf("The node deleted is %d",cur->info);
freenode(cur);
return head;
}
NODE insert_leftpos(int item,NODE head)
{
NODE temp,cur,prev;
if(head->rlink==head)
{
printf("List empty\n");
return head;
}
cur=head->rlink;
while(cur!=head)
```

```
{
if(item==cur->info)break;
cur=cur->rlink;
}
if(cur==head)
{
 printf("Key not found\n");
 return head;
 }
 prev=cur->llink;
 printf("Enter towards left of %d=",item);
 temp=getnode();
 scanf("%d",&temp->info);
 prev->rlink=temp;
 temp->llink=prev;
 cur->llink=temp;
 temp->rlink=cur;
 return head;
}
NODE delete_all_key(int item,NODE head)
{
NODE prev,cur,next;
int count;
  if(head->rlink==head)
   {
    printf("LE");
    return head;
    }
count=0;
cur=head->rlink;
while(cur!=head)
```

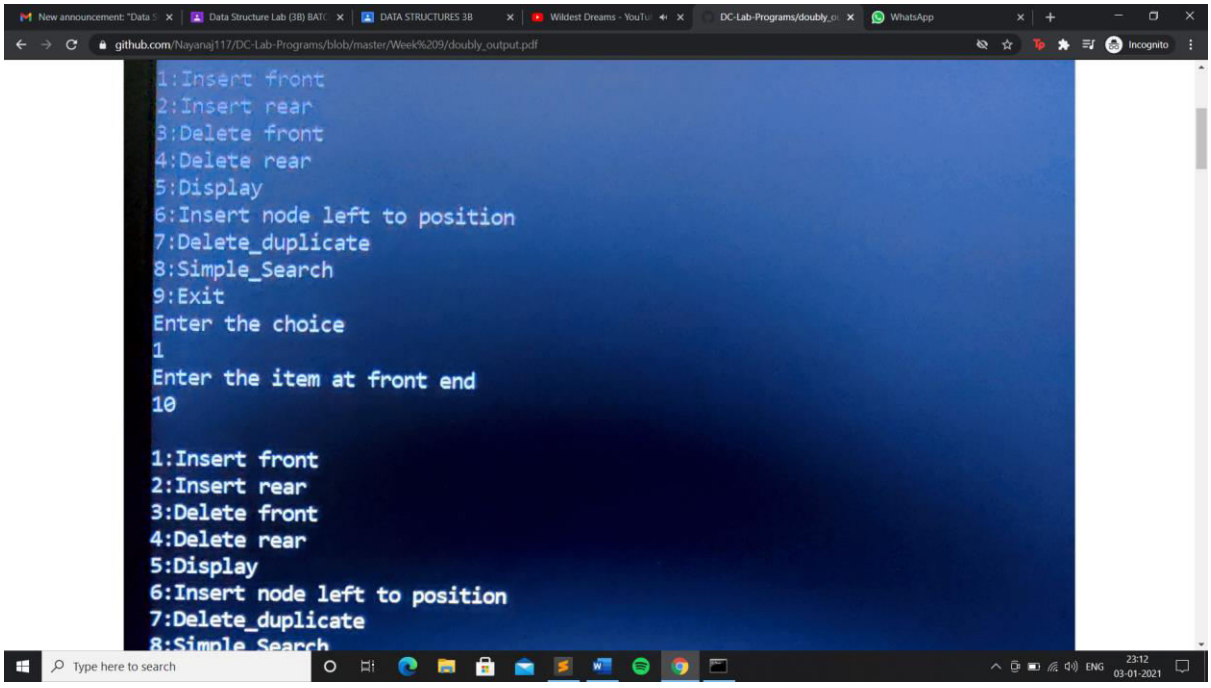
```
{
 if(item!=cur->info)
 cur=cur->rlink;
 else
{
 count++;
 prev=cur->llink;
 next=cur->rlink;
 prev->rlink=next;
 next->llink=prev;
 freenode(cur);
 cur=next;
 }
}

if(count==0)
 printf("Key not found");
 else
printf("Key found at %d positions and are deleted\n", count);

return head;
}
void search(int item,NODE head){
   NODE cur;
   if(head->rlink==head)
   {
    printf("List Empty");
    return ;
    }
    cur=head->rlink;
    while(cur!=head)
```

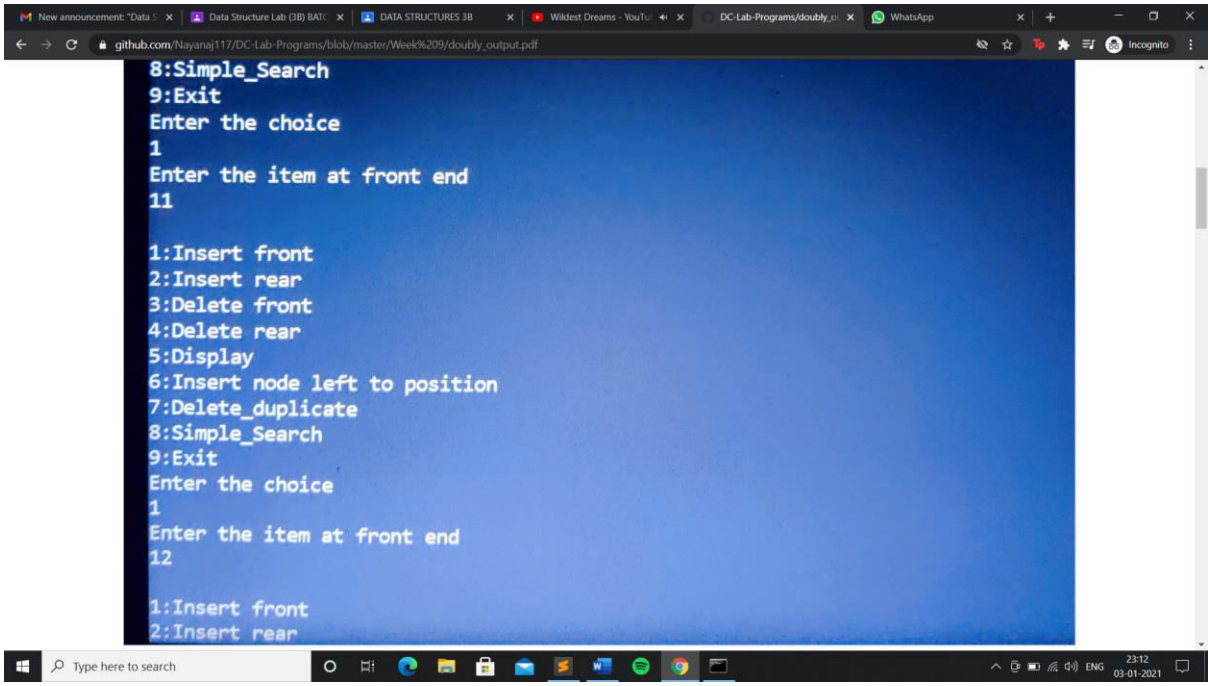
```
    {
    if(item==cur->info)break;
    cur=cur->rlink;
    }
    if(cur==head)
    {printf("Search unsuccessfull\n");
       return;
    }
    printf("Search successful\n");

}

void display(NODE head)
{
NODE temp;
if(head->rlink==head)
{
printf("Dq empty\n");
return;
}
printf("Contents of dq\n");
temp=head->rlink;
while(temp!=head)
{
printf("%d\n",temp->info);
temp=temp->rlink;
}
printf("\n");
}
void main()
{
```

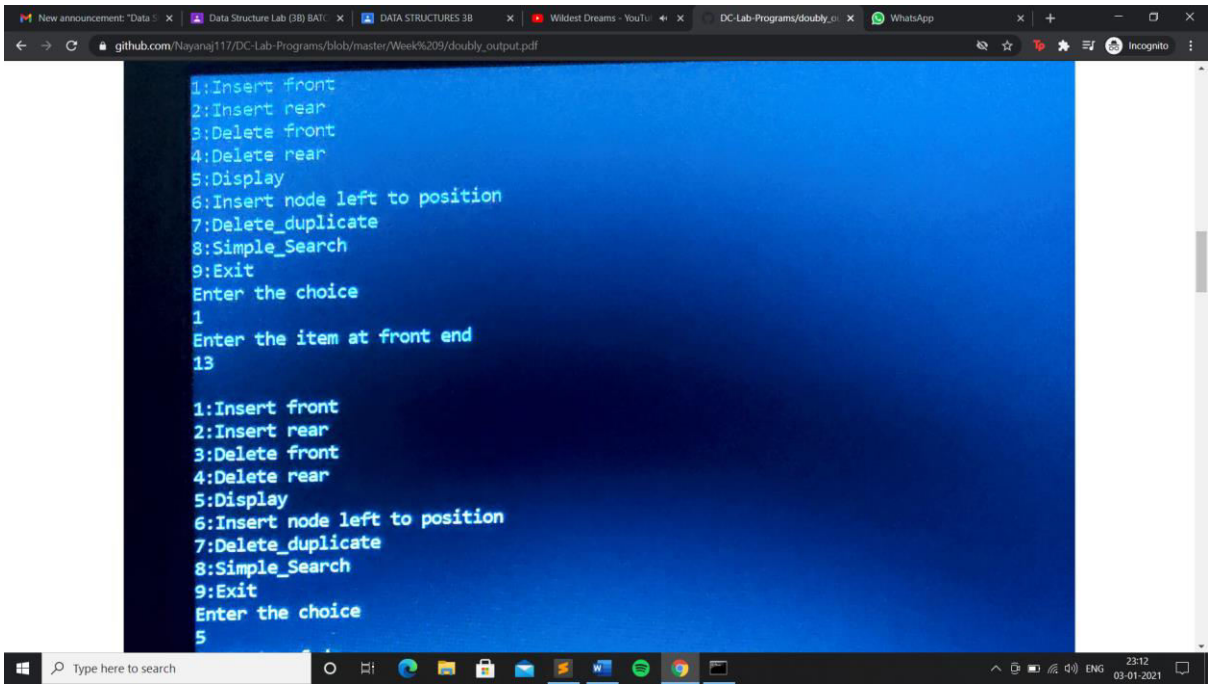
```
NODE head,last;
int item, choice;
head=getnode();
head->rlink=head;
head->llink=head;

for(;;)
{
        printf("\n1:Insert front\n2:Insert rear\n3:Delete front\n4:Delete rear\n5:Display\n6:Insert node left to position\n7:Delete_duplicate\n8:Simple_Search\n9:Exit\n");
        printf("Enter the choice\n");
        scanf("%d",&choice);
        switch(choice)
        {
                case 1: printf("Enter the item at front end\n");
                        scanf("%d",&item);
                        last=dinsert_front(item,head);
                        break;
                case 2: printf("Enter the item at rear end\n");
                        scanf("%d",&item);
                        last=dinsert_rear(item,head);
                        break;
                case 3:last=ddelete_front(head);
                        break;
                case 4: last=ddelete_rear(head);
                        break;
                case 5: display(head);
                        break;
                        case 6:printf("Enter the key item\n");
                 scanf("%d",&item);
                 head=insert_leftpos(item,head);
```

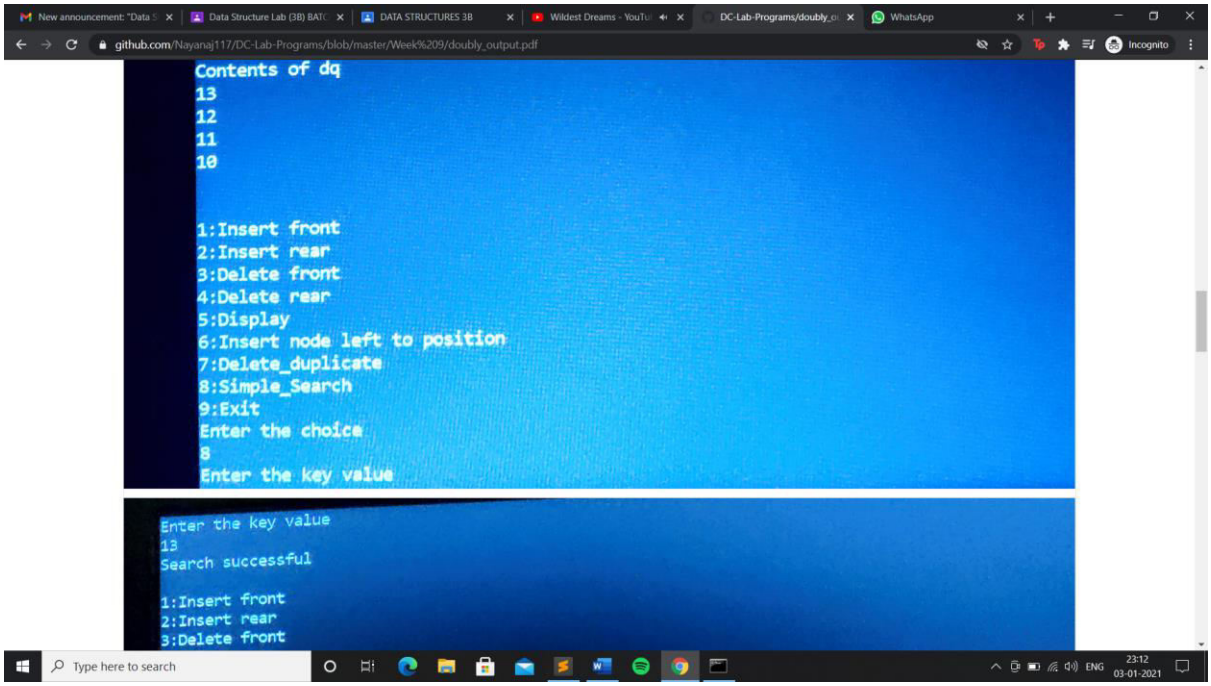
```
                    break;
                    case 7:printf("Enter the key value\n");
                    scanf("%d",&item);
                    delete_all_key(item,head);
                    break;
                    case 8:printf("Enter the key value\n");
                    scanf("%d",&item);
                    search(item,head);
                    break;
                    case 9:exit(0);

                   default:printf("Invalid choice\n");
                   }
          }
}
```

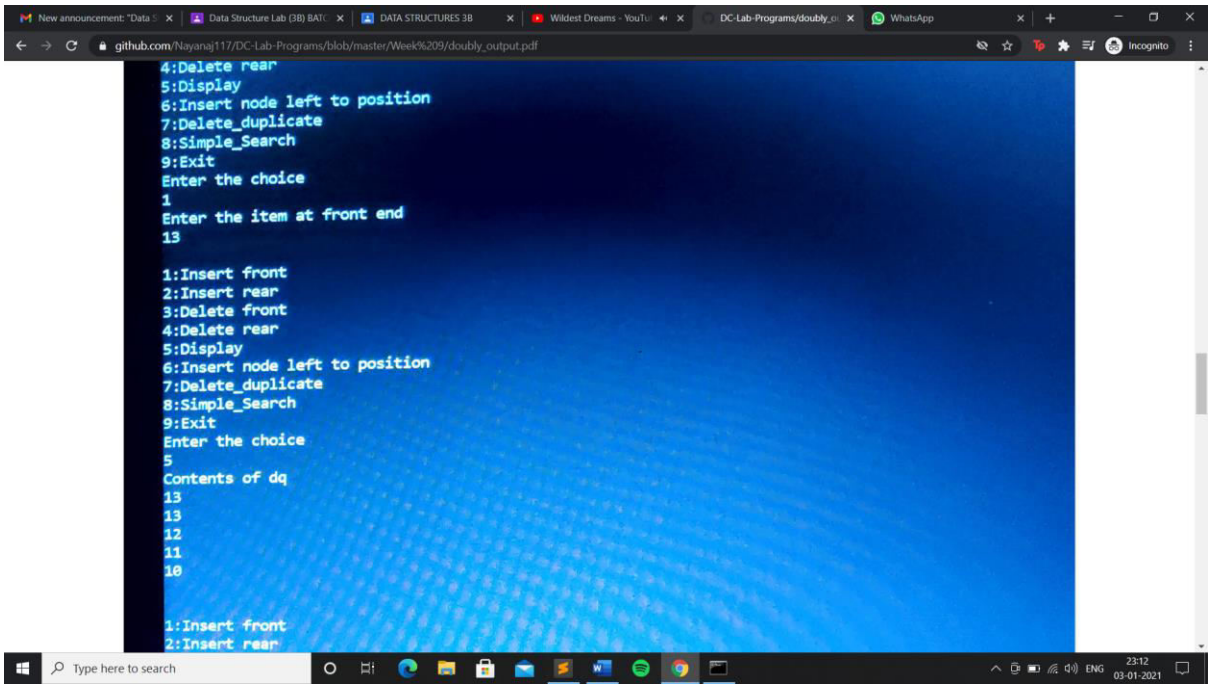
OUTPUT:

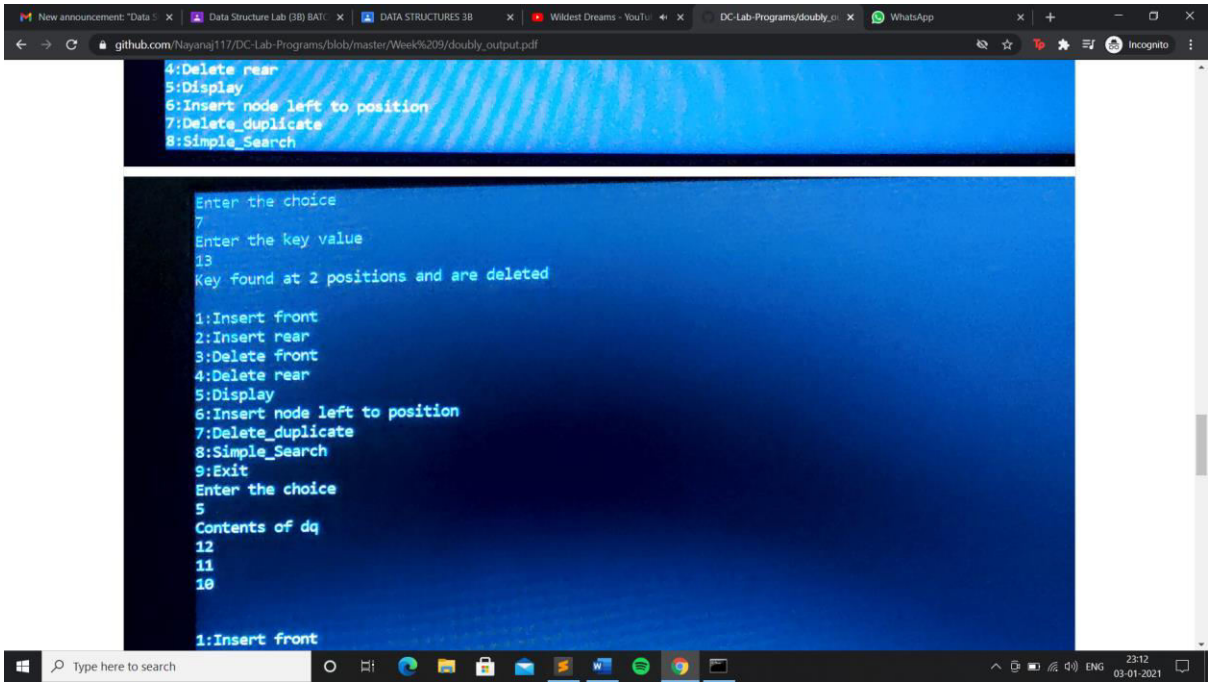


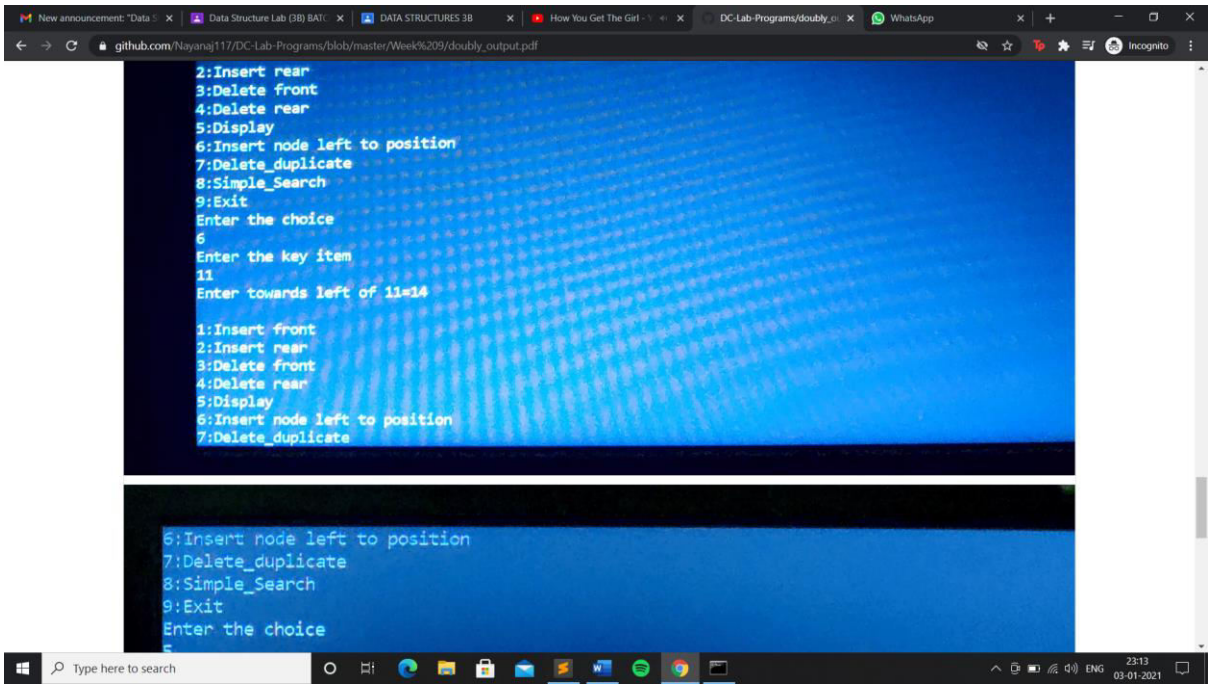

github.com/Nayanaj117/DC-Lab-Programs/blob/master/Week%209/doubly_output.pdf
8:Simple_Search
9:Exit
Enter the choice
1
Enter the item at front end
11
1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Display
6:Insert node left to position
7:Delete_duplicate
8:Simple_Search
9:Exit
Enter the choice
1
Enter the item at front end
12
1:Insert front
2:Insert rear
Type here to search

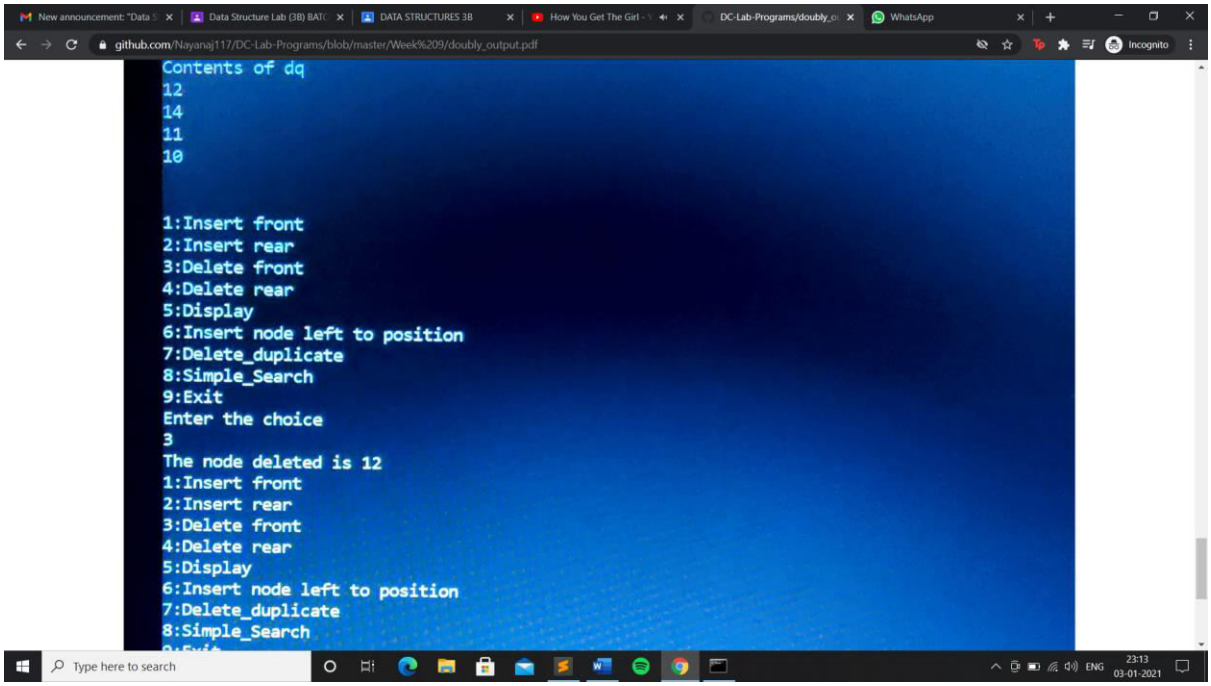

github.com/Nayanaj117/DC-Lab-Programs/blob/master/Week%209/doubly_output.pdf
1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Display
6:Insert node left to position
7:Delete_duplicate
8:Simple_Search
9:Exit
Enter the choice
1
Enter the item at front end
13
1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Display
6:Insert node left to position
7:Delete_duplicate
8:Simple_Search
9:Exit
Enter the choice
5
Type here to search

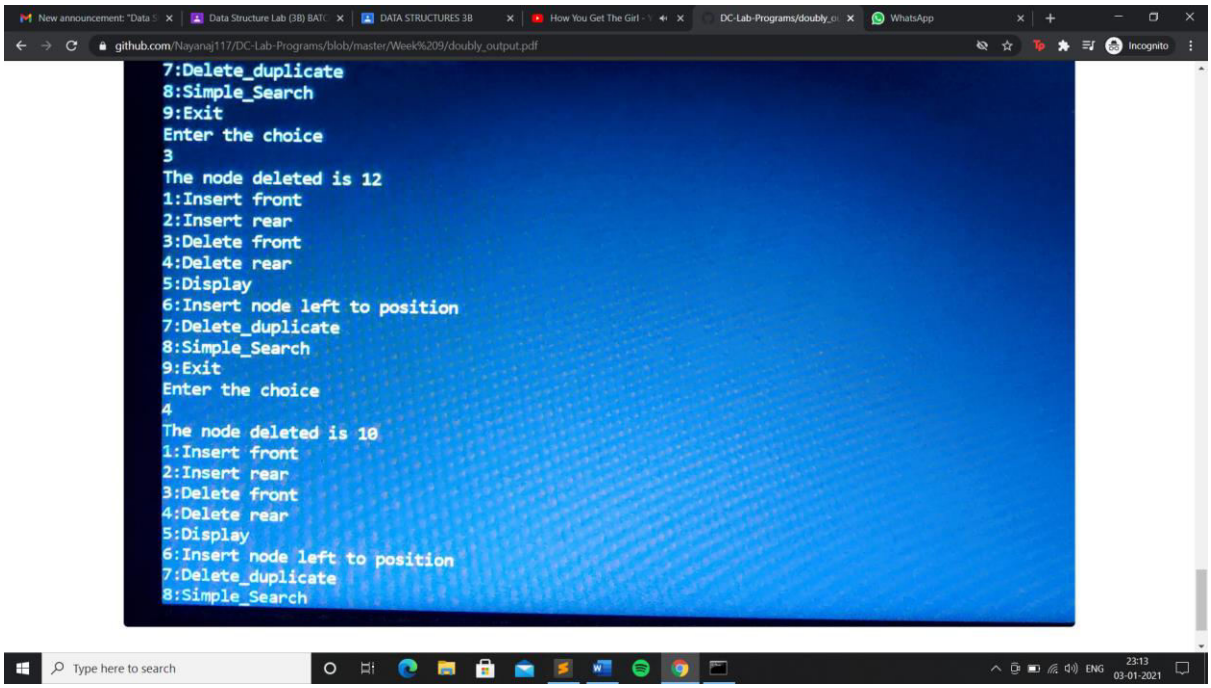

github.com/Nayanaj117/DC-Lab-Programs/blob/master/Week%209/doubly_output.pdf
Contents of dq
13
12
11
10
1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Display
6:Insert node left to position
7:Delete_duplicate
8:Simple_Search
9:Exit
Enter the choice
8
Enter the key value
Enter the key value
13
Search successful
1:Insert front
2:Insert rear
3:Delete front
Type here to search
ENG
23:12
03-01-2021


github.com/Nayanaj117/DC-Lab-Programs/blob/master/Week%209/doubly_output.pdf
4:Delete rear
5:Display
6:Insert node left to position
7:Delete_duplicate
8:Simple_Search
9:Exit
Enter the choice
1
Enter the item at front end
13
1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Display
6:Insert node left to position
7:Delete_duplicate
8:Simple_Search
9:Exit
Enter the choice
5
Contents of dq
13
13
12
11
10
1:Insert front
2:Insert rear
Type here to search
ENG
23:12
03-01-2021


github.com/Nayanaj117/DC-Lab-Programs/blob/master/Week%209/doubly_output.pdf
4:Delete rear
5:Display
6:Insert node left to position
7:Delete_duplicate
8:Simple_Search
Enter the choice
7
Enter the key value
13
Key found at 2 positions and are deleted
1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Display
6:Insert node left to position
7:Delete_duplicate
8:Simple_Search
9:Exit
Enter the choice
5
Contents of dq
12
11
10
1:Insert front
Type here to search
ENG
23:12
03-01-2021


github.com/Nayanaj117/DC-Lab-Programs/blob/master/Week%209/doubly_output.pdf
2:Insert rear
3:Delete front
4:Delete rear
5:Display
6:Insert node left to position
7:Delete_duplicate
8:Simple_Search
9:Exit
Enter the choice
6
Enter the key item
11
Enter towards left of 11=14
1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Display
6:Insert node left to position
7:Delete_duplicate
6:Insert node left to position
7:Delete_duplicate
8:Simple_Search
9:Exit
Enter the choice
Type here to search
ENG
23:13
03-01-2021


github.com/Nayanaj117/DC-Lab-Programs/blob/master/Week%209/doubly_output.pdf
Contents of dq
12
14
11
10
1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Display
6:Insert node left to position
7:Delete_duplicate
8:Simple_Search
9:Exit
Enter the choice
3
The node deleted is 12
1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Display
6:Insert node left to position
7:Delete_duplicate
8:Simple_Search
Type here to search
ENG
23:13
03-01-2021


github.com/Nayanaj117/DC-Lab-Programs/blob/master/Week%209/doubly_output.pdf
7:Delete_duplicate
8:Simple_Search
9:Exit
Enter the choice
3
The node deleted is 12
1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Display
6:Insert node left to position
7:Delete_duplicate
8:Simple_Search
9:Exit
Enter the choice
4
The node deleted is 10
1:Insert front
2:Insert rear
3:Delete front
4:Delete rear
5:Display
6:Insert node left to position
7:Delete_duplicate
8:Simple_Search
Type here to search
ENG
23:13
03-01-2021