190905514 **MOHAMMAD TOFIK**

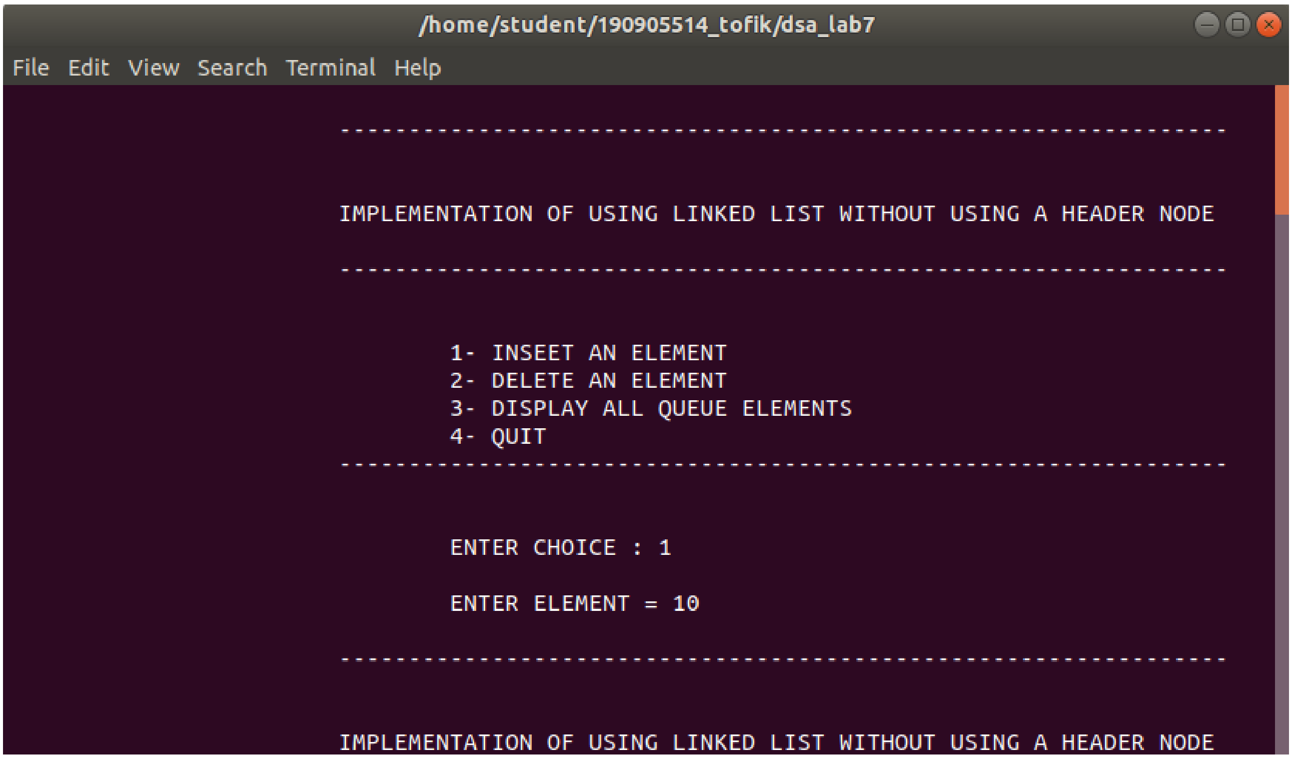
WEEK 7 LAB 7 :

1.Implement a queue using singly linked list without header node.

```
#include<stdio.h>
#include <stdlib.h>
typedef struct node *NODEPTR;
struct node
{
int info;
NODEPTR link;
};
NODEPTR getNode()
{
NODEPTR temp;
temp = (NODEPTR)malloc(sizeof(struct node));
if (temp == NULL)
{
printf("\n\t\t\t\tNO MEMEORY\n\n");
exit(0);
}
return temp;
}
void enqueue(NODEPTR *front, NODEPTR *rear, int data)
{
NODEPTR temp = getNode();
temp->link = NULL;
temp->info = data;
if (*front == NULL)
{
*front = *rear = temp;
}
else
{
(*rear)->link = temp;
(*rear) = temp;
}
}
int dequeue(NODEPTR *front, NODEPTR *rear)
{
NODEPTR temp;
if (*front == NULL)
{
printf("\n\t\t\t\tEMPTY\n");
return -1;
}
temp = (*front);
if (*front == *rear)
{
*front = *rear = NULL;
}
else
{
*front = temp->link;
}
int x = temp->info;
free(temp);
return x;
}
```

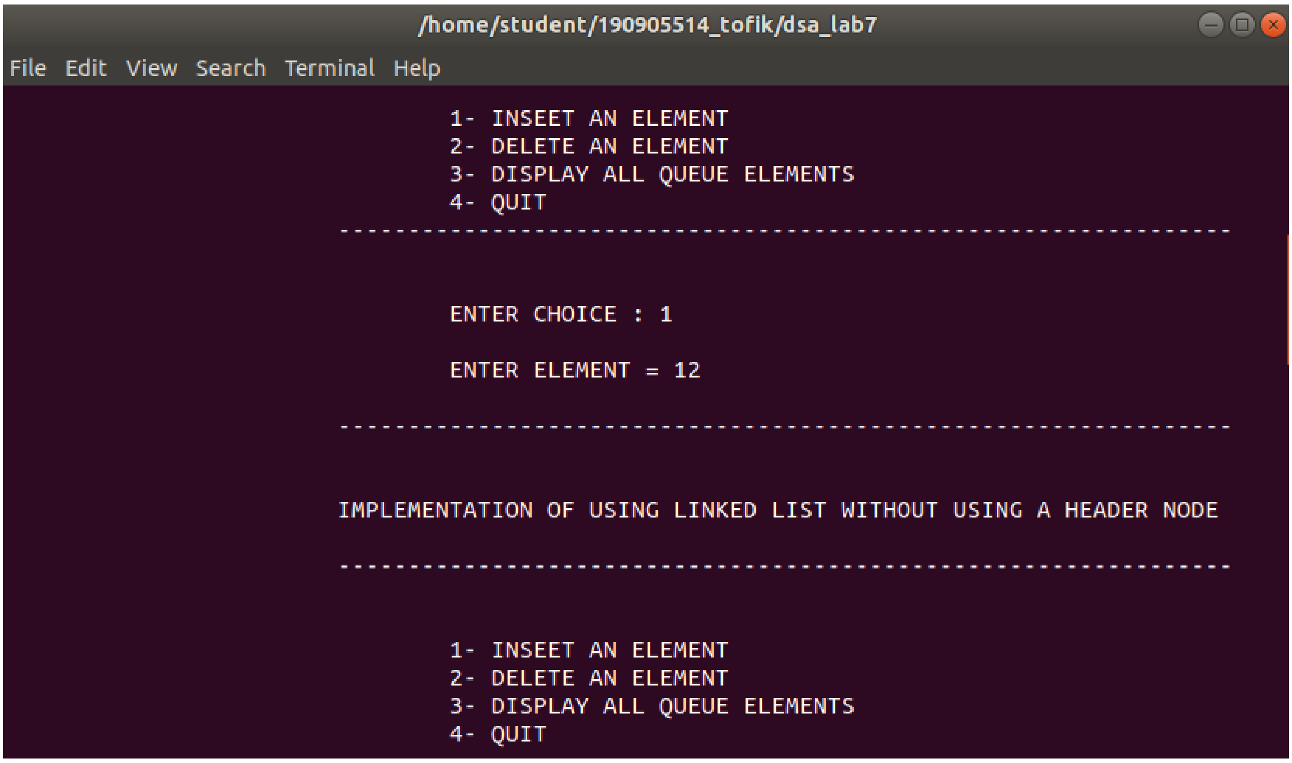
```c
void display(NODEPTR *front, NODEPTR *rear)
{
NODEPTR temp;
if (*front == NULL)
{
printf("\n\t\t\t\tEMPTY\n");
return;
}
temp = (*front);
printf("\n\t\t\t\tQUEUE CONTAINS  ");
for (; temp != *rear; temp = temp->link)
{
printf("%d ", temp->info);
}
printf("\n\t\t\t\t%d ", temp->info);
printf("\n\n");
}
int main()
{
NODEPTR front = NULL;
NODEPTR rear = NULL;
int choice, x;
while (1)
{
printf("\n\t\t\t----------------------------------------------------------------\n\n");
printf("\n\t\t\tIMPLEMENTATION OF USING LINKED LIST WITHOUT USING A HEADER NODE \n");
printf("\n\t\t\t----------------------------------------------------------------\n\n");
printf("\n\t\t\t\t1- INSEET AN ELEMENT");
printf("\n\t\t\t\t2- DELETE AN ELEMENT");
printf("\n\t\t\t\t3- DISPLAY ALL QUEUE ELEMENTS");
printf("\n\t\t\t\t4- QUIT");
printf("\n\t\t\t----------------------------------------------------------------\n\n");
printf("\n\t\t\t\tENTER CHOICE : ");
scanf("%d", &choice);
switch (choice)
{
case 1:
printf("\n\t\t\t\tENTER ELEMENT = ");
scanf("%d", &x);
enqueue(&front, &rear, x);
break;
case 2:
x = dequeue(&front, &rear);
if (x != -1)
printf("\n\t\t\t\t DELETED ELEMENT are = %d", x);
printf("\n\n");
break;
case 3:
display(&front, &rear);
break;
case 4:
printf("\n\t\t\t\tEXITITNG\n\n");
exit(0);
default:
printf("\n\t\t\t\tEXITING\n\n");
return 0;
}
}
return 0;
}
```

OUTPUT :



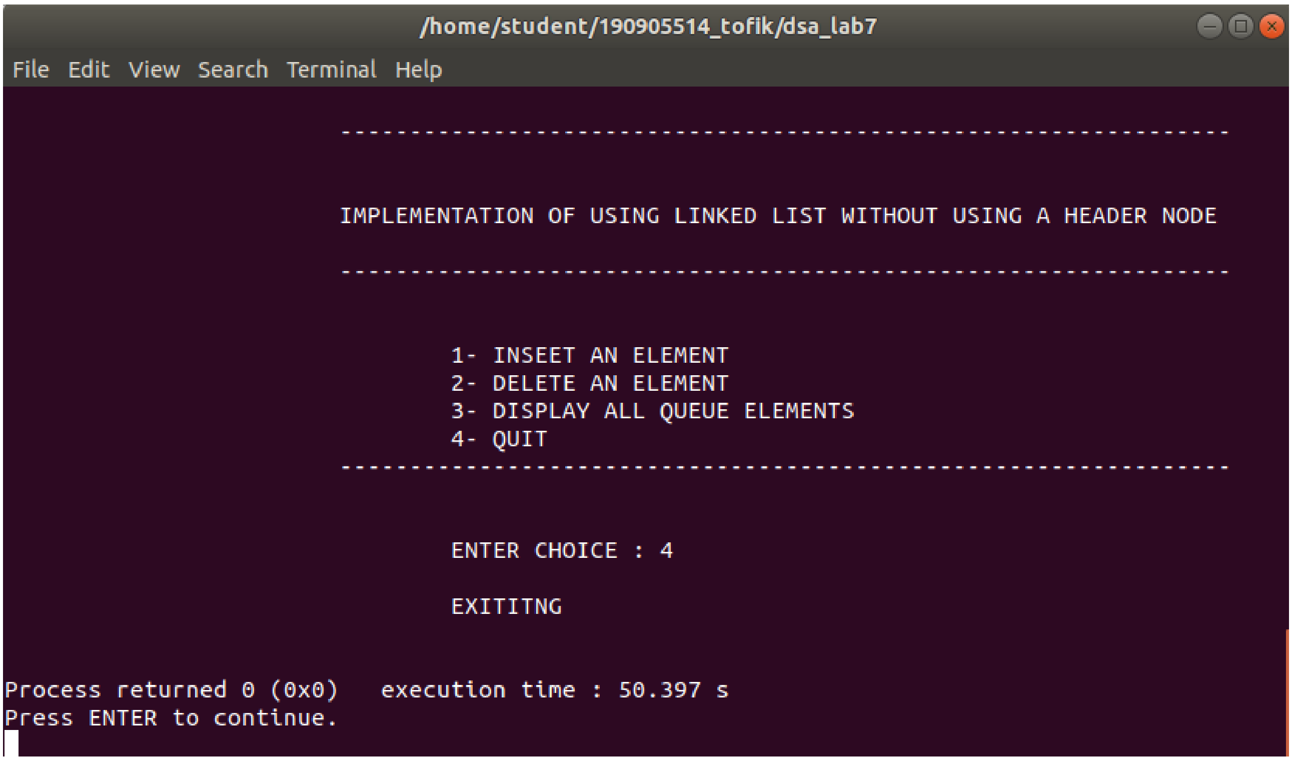
/home/student/190905514_tofik/dsa_lab7
File Edit View Search Terminal Help

```
                        -----------------------------------------------------------------


                        IMPLEMENTATION OF USING LINKED LIST WITHOUT USING A HEADER NODE

                        -----------------------------------------------------------------


                                1- INSEET AN ELEMENT
                                2- DELETE AN ELEMENT
                                3- DISPLAY ALL QUEUE ELEMENTS
                                4- QUIT
                        -----------------------------------------------------------------


                                ENTER CHOICE : 1

                                ENTER ELEMENT = 10

                        -----------------------------------------------------------------


                        IMPLEMENTATION OF USING LINKED LIST WITHOUT USING A HEADER NODE
```



/home/student/190905514_tofik/dsa_lab7
File Edit View Search Terminal Help

```
                                1- INSEET AN ELEMENT
                                2- DELETE AN ELEMENT
                                3- DISPLAY ALL QUEUE ELEMENTS
                                4- QUIT
                        -----------------------------------------------------------------


                                ENTER CHOICE : 1

                                ENTER ELEMENT = 12

                        -----------------------------------------------------------------


                        IMPLEMENTATION OF USING LINKED LIST WITHOUT USING A HEADER NODE

                        -----------------------------------------------------------------


                                1- INSEET AN ELEMENT
                                2- DELETE AN ELEMENT
                                3- DISPLAY ALL QUEUE ELEMENTS
                                4- QUIT
```


/home/student/190905514_tofik/dsa_lab7
File Edit View Search Terminal Help

```
                        ---------------------------------------------------------------

                        IMPLEMENTATION OF USING LINKED LIST WITHOUT USING A HEADER NODE

                        ---------------------------------------------------------------


                                1- INSEET AN ELEMENT
                                2- DELETE AN ELEMENT
                                3- DISPLAY ALL QUEUE ELEMENTS
                                4- QUIT
                        ---------------------------------------------------------------


                                ENTER CHOICE : 4

                                EXITITNG


Process returned 0 (0x0)   execution time : 50.397 s
Press ENTER to continue.
```

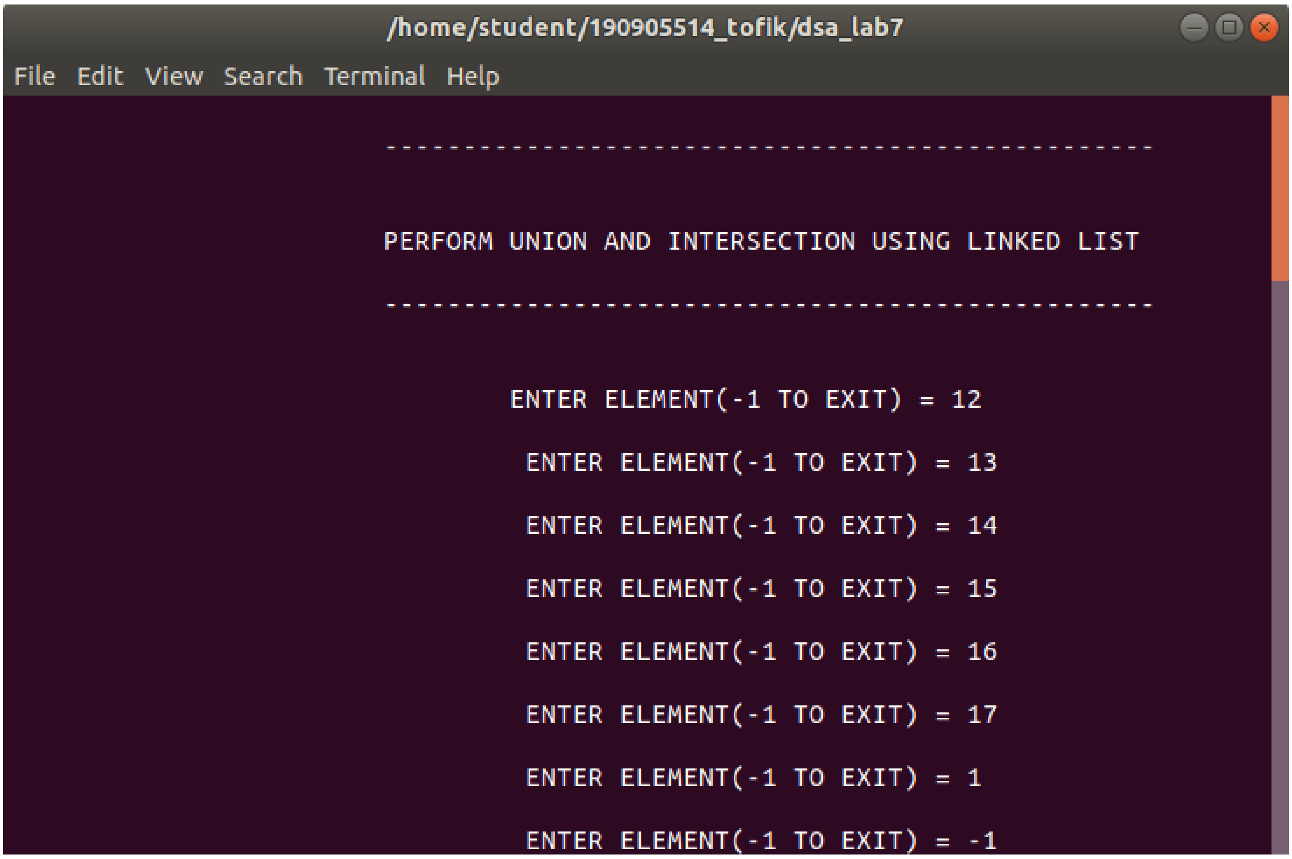
2.Perform UNION and INTERSECTION set operations on singly linked lists with header node.

```
#include <stdio.h>
#include <stdlib.h>
typedef struct node *NODEPTR;
struct node
{
int info;
NODEPTR link;
};
NODEPTR getNode()
{
NODEPTR temp;
temp = (NODEPTR)malloc(sizeof(struct node));
if (temp == NULL)
{
printf("\n\t\t\t\tNO MEMEORY");
exit(0);
}
return temp;
}
NODEPTR insertlast(NODEPTR last, int data)
{
NODEPTR temp = getNode();
temp->link = NULL;
temp->info = data;
last->link = temp;
last = temp;
return last;
}
NODEPTR createlist()
{
NODEPTR first = getNode();
NODEPTR last = first;
int x;
printf("\n\t\t\t\tENTER ELEMENT(-1 TO EXIT) = ");
```

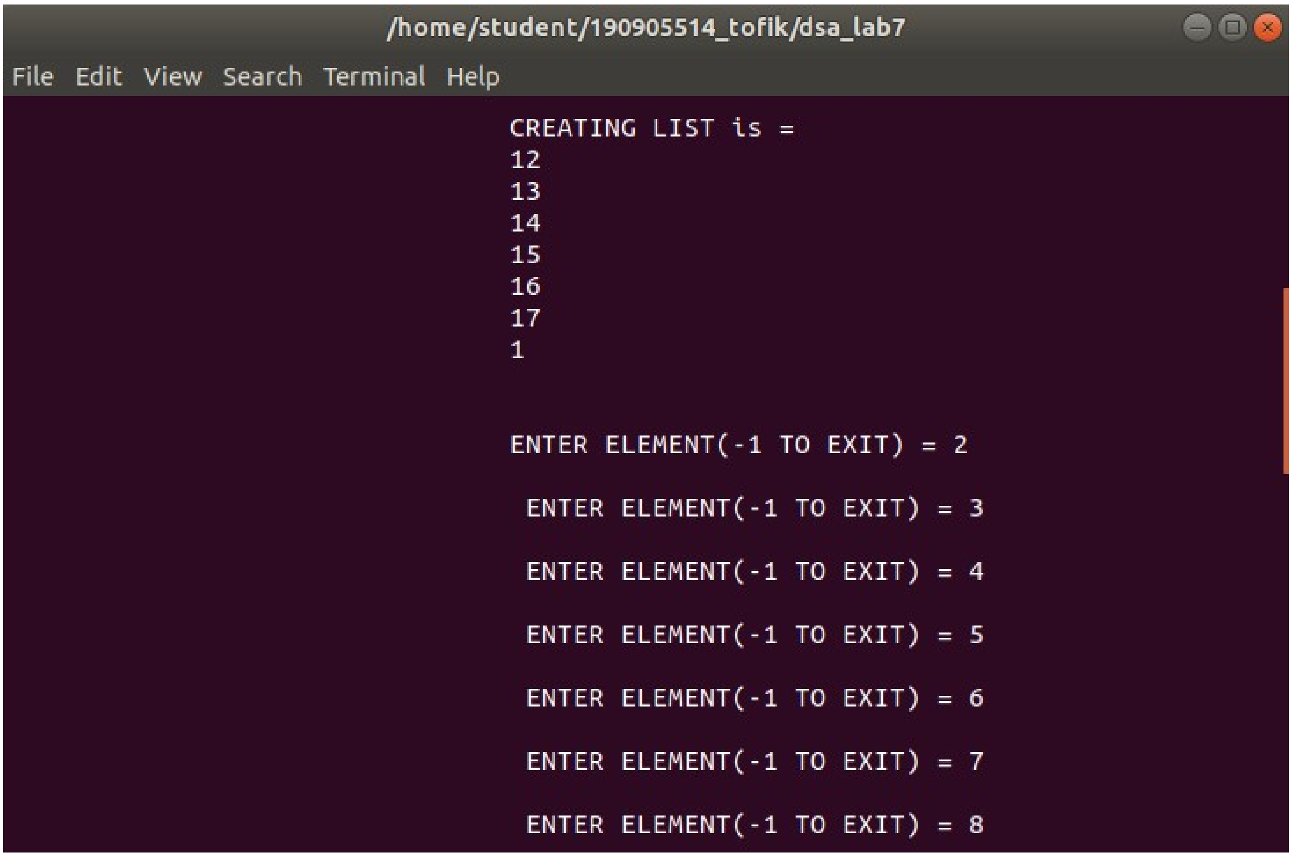
```
scanf("%d", &x);
while (x != -1)
{
last = insertlast(last, x);
printf("\n\t\t\t\t ENTER ELEMENT(-1 TO EXIT) = ");
scanf("%d", &x);
}
NODEPTR temp = first;
first = temp->link;
free(temp);
return first;
}
void display(NODEPTR front)
{
NODEPTR temp;
temp = (front);
printf("\n\t\t\t\tCREATING LIST is = ");
for (; temp != NULL; temp = temp->link)
{
printf("\n\t\t\t\t%d ", temp->info);
}
printf("\n\n");
}
int ismember(NODEPTR a, int x)
{
NODEPTR temp = a;
while (temp != NULL)
{
if (temp->info == x)
{
return 1;
}
temp = temp->link;
}
return 0;
}
NODEPTR dunion(NODEPTR a, NODEPTR b)
{
NODEPTR c = getNode();
NODEPTR x = c;
NODEPTR temp = a;
while (temp != NULL)
{
x = insertlast(x, temp->info);
temp = temp->link;
}
temp = b;
while (temp != NULL)
{
if (ismember(a, temp->info) == 0)
x = insertlast(x, temp->info);
temp = temp->link;
}
temp = c;
c = temp->link;
free(temp);
return c;
}
NODEPTR inter(NODEPTR a, NODEPTR b)
{
NODEPTR c = getNode();
NODEPTR x = c;
NODEPTR temp = b;
```

```
while (temp != NULL)
{
if (ismember(a, temp->info) == 1)
x = insertlast(x, temp->info);
temp = temp->link;
}
temp = c;
c = temp->link;
free(temp);
return c;
}
int main()
{
printf("\n\t\t\t--------------------------------------------------\n\n");
printf("\n\t\t\tPERFORM UNION AND INTERSECTION USING LINKED LIST\n");
printf("\n\t\t\t--------------------------------------------------\n\n");

NODEPTR a = createlist();
display(a);
NODEPTR b = createlist();
display(b);
printf("\n\t\t\t\tAfter union = ");
NODEPTR c = dunion(a, b);
display(c);
printf("\n\t\t\t\tAfter intersection = ");
NODEPTR d = inter(a, b);
display(d);
return 0;
}
```

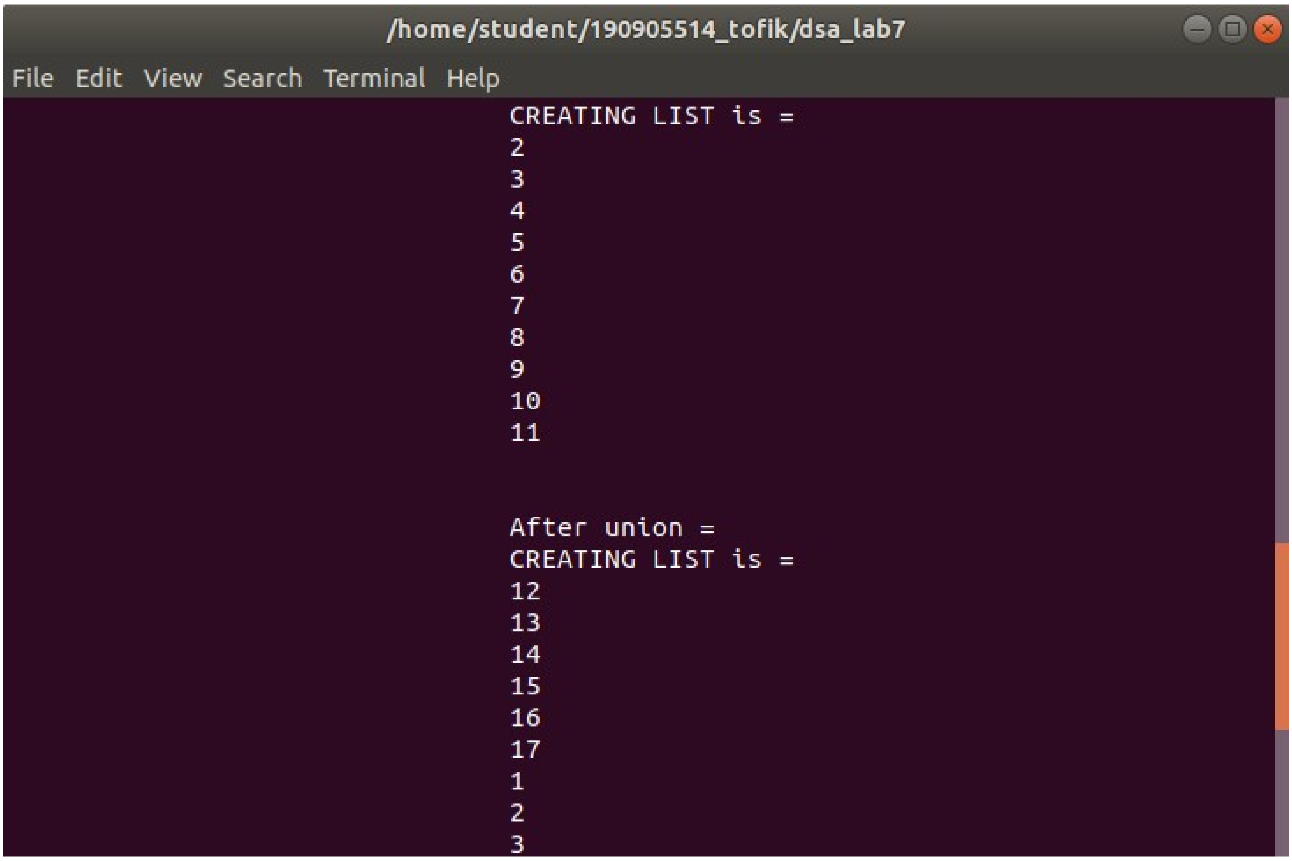
OUTPUT :


/home/student/190905514_tofik/dsa_lab7
File Edit View Search Terminal Help

```
                        --------------------------------------------------


                        PERFORM UNION AND INTERSECTION USING LINKED LIST

                        --------------------------------------------------


                                ENTER ELEMENT(-1 TO EXIT) = 12

                                 ENTER ELEMENT(-1 TO EXIT) = 13

                                 ENTER ELEMENT(-1 TO EXIT) = 14

                                 ENTER ELEMENT(-1 TO EXIT) = 15

                                 ENTER ELEMENT(-1 TO EXIT) = 16

                                 ENTER ELEMENT(-1 TO EXIT) = 17

                                 ENTER ELEMENT(-1 TO EXIT) = 1

                                 ENTER ELEMENT(-1 TO EXIT) = -1
```


/home/student/190905514_tofik/dsa_lab7
File Edit View Search Terminal Help

```
CREATING LIST is =
12
13
14
15
16
17
1


ENTER ELEMENT(-1 TO EXIT) = 2

 ENTER ELEMENT(-1 TO EXIT) = 3

 ENTER ELEMENT(-1 TO EXIT) = 4

 ENTER ELEMENT(-1 TO EXIT) = 5

 ENTER ELEMENT(-1 TO EXIT) = 6

 ENTER ELEMENT(-1 TO EXIT) = 7

 ENTER ELEMENT(-1 TO EXIT) = 8
```


/home/student/190905514_tofik/dsa_lab7
File Edit View Search Terminal Help

```
CREATING LIST is =
2
3
4
5
6
7
8
9
10
11


After union =
CREATING LIST is =
12
13
14
15
16
17
1
2
3
```