01/07/2021

Annapoorneswari D
MCA batch A
Roll No:20
Reg.No: AJC20MCA-2020

DATA STRUCTURE LAB (20MCA135)

1. Implement two singly linked list and implement SET operations

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

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

struct node * Union(struct node * L1, struct node * L2){
  struct node * output = NULL;
  struct node * outTail = NULL;
  while(L1&&L2){
    struct node * newNode = (struct node *) malloc(sizeof(struct
node));
    newNode->next = NULL;
    if(L1->data<L2->data){
      newNode->data = L1->data;
      L1 = L1->next;
    }
    else if(L1->data>L2->data){
      newNode->data = L2->data;
      L2 = L2->next;
    }
    else{
      int data = L1->data;
      newNode->data = data;
      while(L1 && L2 && L1->data == data && L2->data == data){
        L1 = L1->next;
        L2 = L2->next;
      }
    }
    if(!output)
      output = outTail = newNode;
    else{
      outTail->next = newNode;
      outTail = outTail->next;
    }
  }
  while(L1){
    outTail->next = (struct node *) malloc(sizeof(struct node));
    outTail = outTail->next;
    outTail->data = L1->data;
    L1 = L1->next;
  }
  while(L2){
    outTail->next = (struct node *) malloc(sizeof(struct node));
    outTail = outTail->next;
    outTail->data = L2->data;
    L2 = L2->next;
  }
  outTail->next = NULL;
  return output;
}

struct node * intersection(struct node * L1, struct node* L2){
  if(L1 == NULL || L2 == NULL)
    return NULL;
  struct node * output = NULL;
  struct node * outTail = NULL;
  while(L1&&L2){
    if(L1->data<L2->data){
      L1 = L1->next;
    }
    else if(L2->data<L1->data){
      L2 = L2->next;
    }
    else{
      int data = L1->data;
      struct node * newNode = (struct node *) malloc(sizeof(struct node));
      newNode->data = data;
      newNode->next = NULL;
      if(output == NULL){
        outTail = output = newNode;
      }
      else{
        outTail->next = newNode;
        outTail = outTail->next;
      }
      while(L1 && L2 && L1->data == data && L2->data == data){
        L1 = L1->next;
        L2 = L2->next;
      }
    }
  }
  return output;
}

struct node * createList(int listNum){
  struct node * list = NULL;
  struct node * list_tail = NULL;
  printf("Enter elements of List %d in increasing order\n",listNum);
  char ch = 'y';
  do{
    int data;
    printf("Enter your element : ");
    scanf("%d",&data);
    struct node * newNode = (struct node *) malloc(sizeof(struct node));
    newNode->data = data;
    newNode->next = NULL;
    if(list == NULL){
      list = list_tail = newNode;
    }
    else{
      list_tail->next = newNode;
      list_tail = list_tail->next;
    }
    printf("Would you like to insert another element [Y/N] : ");
    scanf(" %c",&ch);
  }while(ch == 'y' || ch == 'Y');

  return list;
}

void print(struct node * list){
  if(list == NULL){
    printf("Empty List\n");
    return;
  }
  while(list!=NULL){
    printf("%d ",list->data);
    list = list->next;
  }
  printf("\n");
}

int main() {
  struct node * L1 = NULL;
  struct node * L2 = NULL;
  struct node * L3 = NULL;
  struct node * L4 = NULL;

  L1 = createList(1);
  L2 = createList(2);
  printf("List 1 : ");
  print(L1);
  printf("List 2 : ");
  print(L2);
  printf("Union : ");
  L3 = Union(L1, L2);
  print(L3);
  printf("Intersection : ");
  L4 = intersection(L1, L2);
  print(L4);

  printf("\nProgram exit successfully!");
  return 0;
}
```

OUTPUT

```
Enter elements of List 1 in increasing order
Enter your element : 4
Would you like to insert another element [Y/N] : Y
Enter your element : 8
Would you like to insert another element [Y/N] : Y
Enter your element : 12
Would you like to insert another element [Y/N] : Y
Enter your element : 14
Would you like to insert another element [Y/N] : N
Enter elements of List 2 in increasing order
Enter your element : 2
Would you like to insert another element [Y/N] : Y
Enter your element : 13
Would you like to insert another element [Y/N] : Y
```

Enter your element : 25

Would you like to insert another element [Y/N] : Y

Enter your element : 30

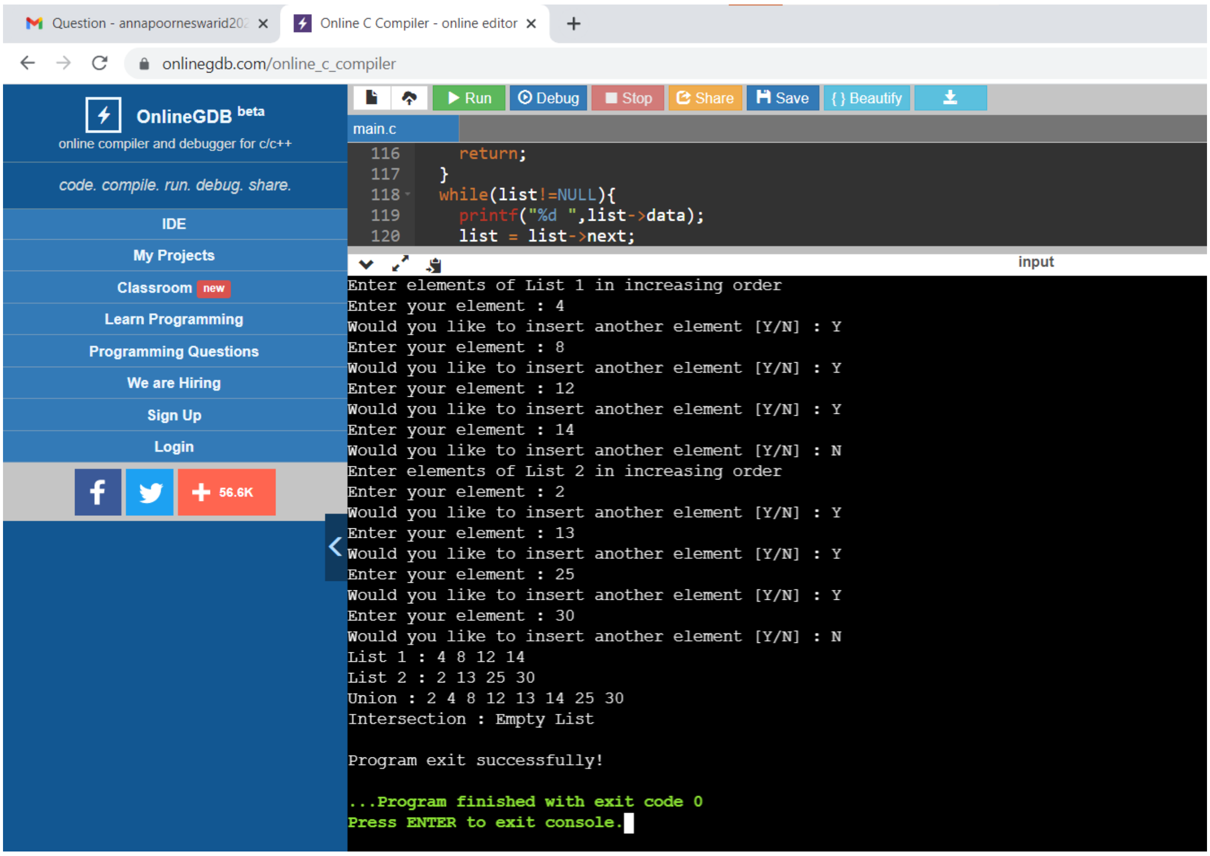
Would you like to insert another element [Y/N] : N

List 1 : 4 8 12 14

List 2 : 2 13 25 30

Union : 2 4 8 12 13 14 25 30

Intersection : Empty List


Question - annapoorneswarid20
Online C Compiler - online editor
onlinegdb.com/online_c_compiler
Run
Debug
Stop
Share
Save
{ } Beautify
OnlineGDB beta
online compiler and debugger for c/c++
code. compile. run. debug. share.
IDE
My Projects
Classroom new
Learn Programming
Programming Questions
We are Hiring
Sign Up
Login
56.6K
main.c
input

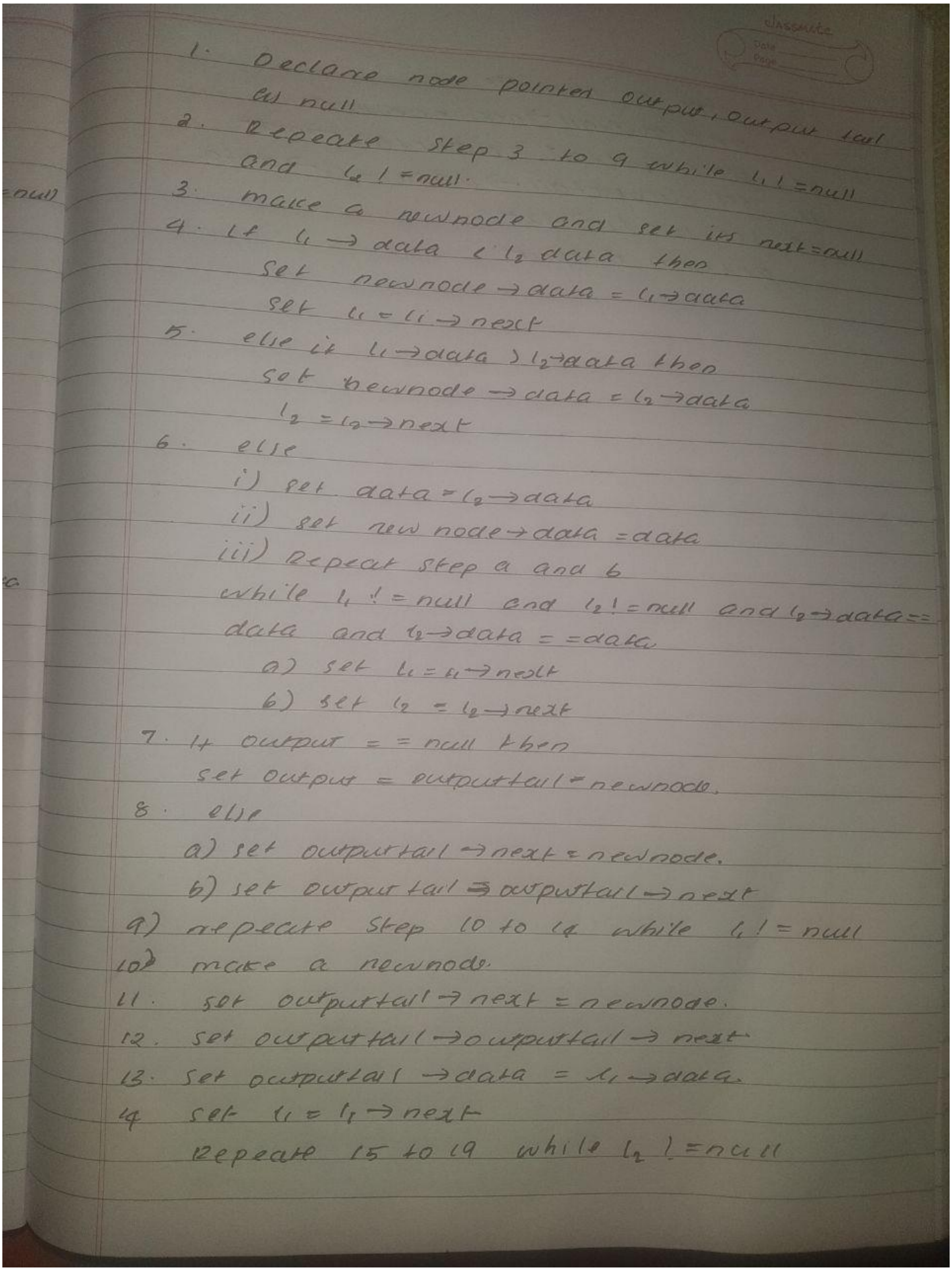
```
    return;
  }
  while(list!=NULL){
    printf("%d ",list->data);
    list = list->next;
```

```
Enter elements of List 1 in increasing order
Enter your element : 4
Would you like to insert another element [Y/N] : Y
Enter your element : 8
Would you like to insert another element [Y/N] : Y
Enter your element : 12
Would you like to insert another element [Y/N] : Y
Enter your element : 14
Would you like to insert another element [Y/N] : N
Enter elements of List 2 in increasing order
Enter your element : 2
Would you like to insert another element [Y/N] : Y
Enter your element : 13
Would you like to insert another element [Y/N] : Y
Enter your element : 25
Would you like to insert another element [Y/N] : Y
Enter your element : 30
Would you like to insert another element [Y/N] : N
List 1 : 4 8 12 14
List 2 : 2 13 25 30
Union : 2 4 8 12 13 14 25 30
Intersection : Empty List

Program exit successfully!

...Program finished with exit code 0
Press ENTER to exit console.
```

ALGORITHMS



```
1.  Declare node pointer output, output tail
    as null
2.  Repeate step 3 to 9 while l1 != null
    and l2 != null.
3.  make a newnode and set its next=null
4.  If l1 → data < l2 data then
        set newnode → data = l1 → data
        set l1 = l1 → next
5.  else if l1 → data > l2 → data then
        set newnode → data = l2 → data
        l2 = l2 → next
6.  else
    i) set data = l2 → data
    ii) set new node → data = data
    iii) Repeat step a and b
    while l1 != null and l2 != null and l2 → data ==
    data and l2 → data = = data
        a) set l1 = l1 → next
        b) set l2 = l2 → next
7.  If output = = null then
    set output = outputtail = newnode.
8.  else
    a) set outputtail → next = newnode.
    b) set output tail ⇒ outputtail → next
9)  repeate step 10 to 14 while l1 != null
10) make a newnode.
11. set outputtail → next = newnode.
12. set outputtail → outputtail → next
13. set outputtail → data = l1 → data
14  set l1 = l1 → next
    Repeate 15 to 19 while l2 != null
```

```
15. make a newnode
16. set outputtail→next = newnode
17. set outputtail = outputtail→next
18. set outputtail→data = $l_2$→data
19. set $l_2 = l_2$→next
    Return output
```