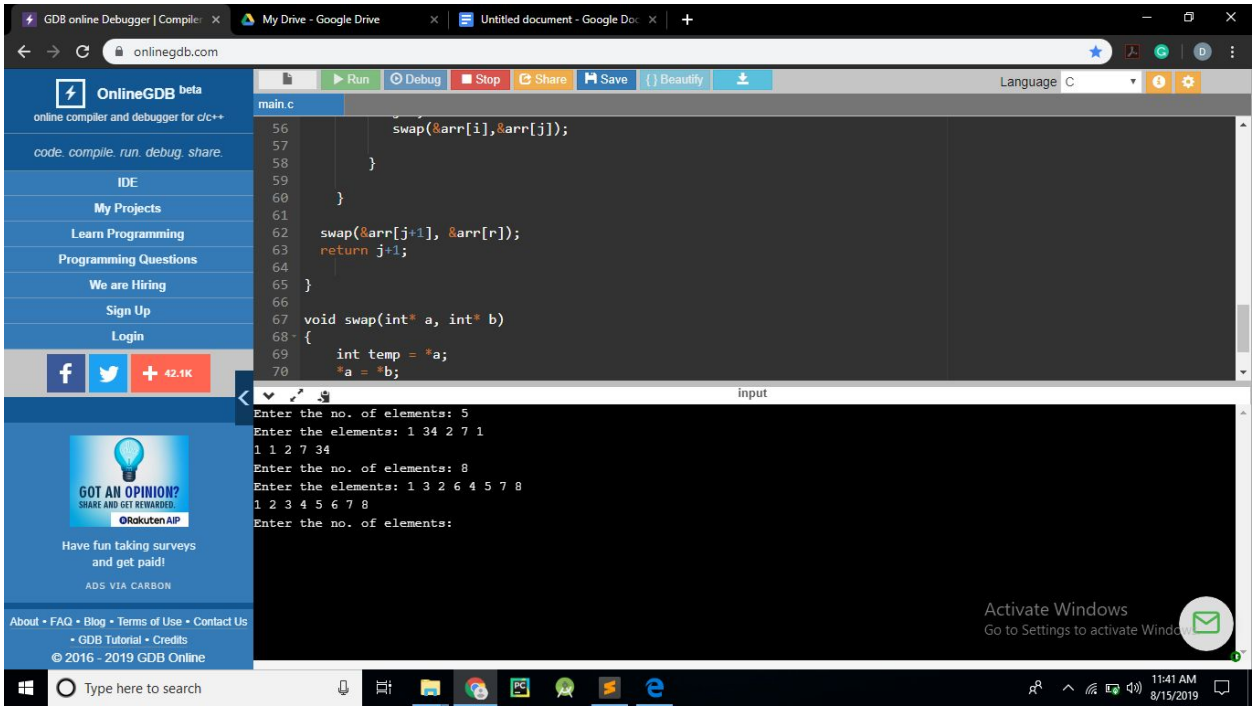
Lab 2 Dhruv Goyal 2018ucp1460

Q1 Quick sort

```
#include<stdio.h>


void quick(int arr[], int p, int r);
int quickpart(int arr[], int p, int r);
void swap(int *a, int* b);

int main()
{
   while(1)
   {
   int a;
   printf("Enter the no. of elements: ");
   scanf("%d", &a);

   int arr[a];

   printf("Enter the elements: ");
   for (int i = 0 ; i < a ; i++)
      scanf("%d", &arr[i]);

   quick(arr, 0, a-1);

   for (int i = 0 ; i < a ; i++)
      printf("%d ", arr[i]);
   printf("\n");
   }

}

void quick(int arr[], int p, int r)
{
   if (p < r)
```

```
    {
     int a = quickpart(arr, p, r);

     quick(arr, p, a-1);
     quick(arr, a+1, r);
    }
}


int quickpart(int arr[], int p, int r)
{


    int pivot = arr[r];

    int j = p-1;
    for (int i = p; i <= r-1 ; i++)
    {

        if (arr[i] <= pivot)
        {
          j++;
          swap(&arr[i],&arr[j]);

        }

    }

  swap(&arr[j+1], &arr[r]);
  return j+1;

}

void swap(int* a, int* b)
{
    int temp = *a;
    *a = *b;
    *b = temp;
```

}



Q2 Linked List

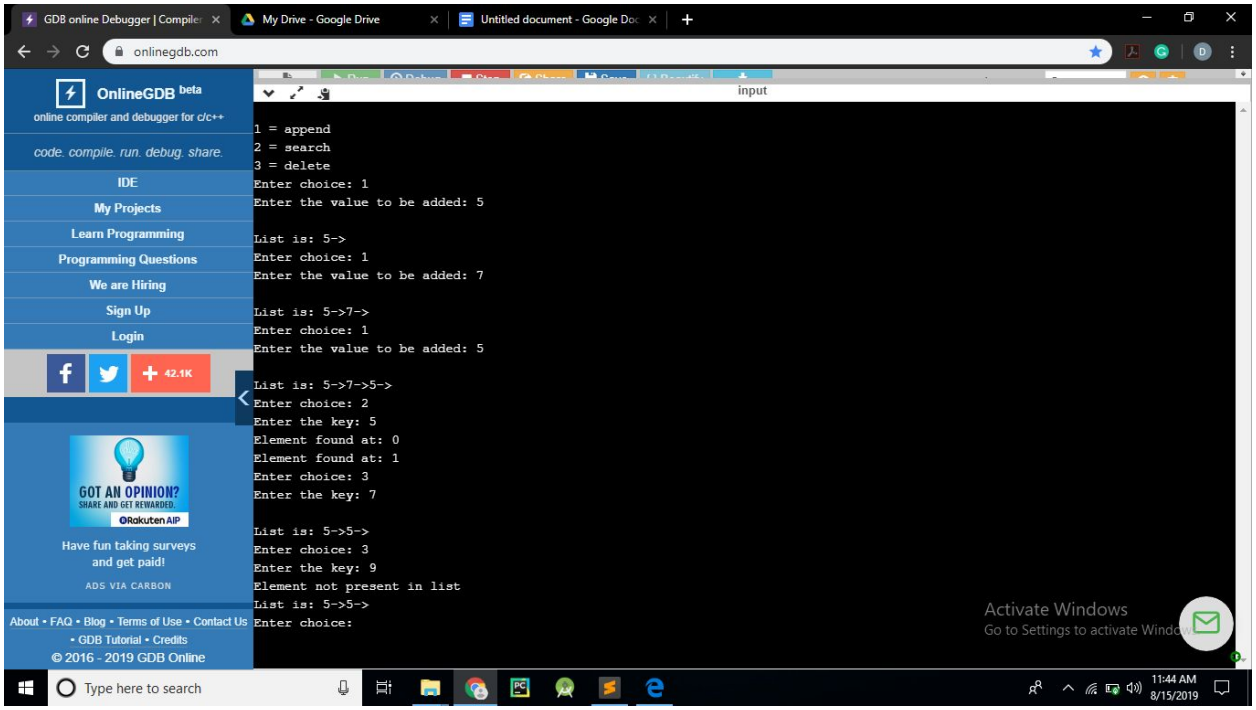
```
#include <stdio.h>
#include<stdlib.h>
struct node{
    int data;
    struct node* next;
};

void append();
void delete();
void search();
void print();
struct node* head = NULL;
struct node* tail = NULL;
int main()
{
    printf("\n1 = append\n2 = search\n3 = delete\n");
```

```
    while(1)
    {
        int n;


        printf("Enter choice: ");
        scanf("%d",&n);

        switch(n)
        {
            case 1 : append();break;
            case 2 : search();break;
            case 3 : delete();break;
            default : printf("Invalid Number\n");
        }



}
}

void print()
{
    struct node*p = head;
    printf("\nList is: ");
    while(p != NULL)
    {
        printf("%d->",p->data);
        p = p->next;
    }
    printf("\n");
}

void append()
{
    struct node* temp = (struct node*)malloc(sizeof(struct node));
    temp->next = NULL;
    printf("Enter the value to be added: ");
    scanf("%d", &(temp->data));
```

```
    if (head == NULL)
    {
        head = temp;
        tail = temp;
    }
    else
    {
        tail->next = temp;
        tail = temp;
    }

   print();
}

void search()
{
    int counter = 0;
    int key;

    printf("Enter the key: ");
    scanf("%d",&key);

    if (head == NULL)
    {
         printf("No elements in the list\n");
    }
    else
    {
        struct node* p = head;
        while(p != NULL)
        {
            if (p->data == key)
            {

                printf("Element found at: %d\n",counter);
                counter++;
            }
            p = p->next;
```

```
        }

          if (counter == 0)
             printf("Element not found");
    }



}


void delete()
{
    int counter = 0 , a = 0;
    int key;

    printf("Enter the key: ");
    scanf("%d",&key);

    if (head == NULL)
         printf("No elements in the list\n");
    else
    {
        struct node* q = head;
        struct node* p = head;
        while(p != NULL)
        {
            if (p->data == key)
            {
                if (p == head)
                {
                    a = 1;
                    head = head->next;
                    struct node* temp = p;
                    p = p->next;
                    temp->next = NULL;
                    free(temp);
```

```
            }
            else
            {
            for (int i = 0 ; i < counter ; i++)
                q = q->next;

            q->next = p->next;
            struct node* temp = p;
            p = p->next;
            temp->next = NULL;
            free(temp);

            counter++;
            }
          }
          else
          {
           p = p->next;
          }


       }
     if (counter == 0 && a == 0)
        printf("Element not present in list");
   }
   print();
}
```



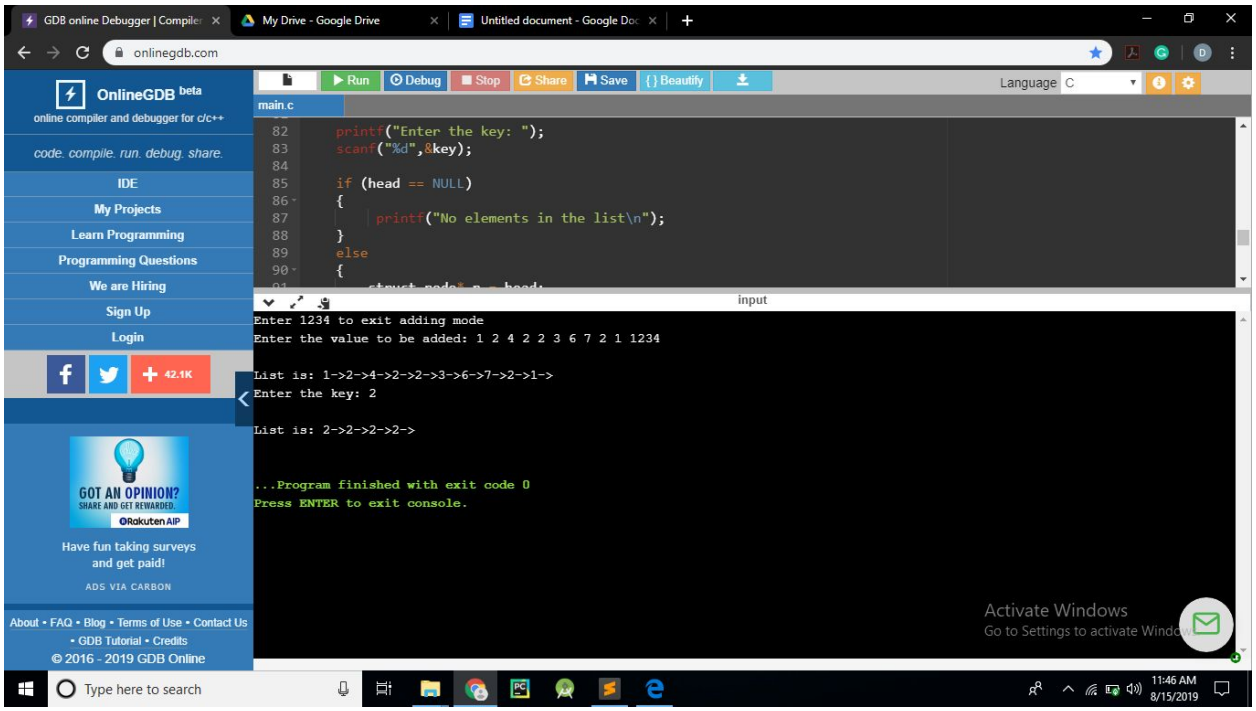
Q3 Key search in Linked List

```
#include <stdio.h>
#include<stdlib.h>
struct node{
    int data;
    struct node* next;
};

void append(struct node** head, struct node** tail, int a);
void search(struct node* head,struct node** h, struct node** tail2);
void print(struct node* head);

int main()
{
    int a;
    struct node* head1 = NULL;
    struct node* tail1 = NULL;

    struct node* head2 = NULL;
```

```
    struct node* tail2 = NULL;

     printf("Enter 1234 to exit adding mode");
     printf("\nEnter the value to be added: ");
    while(1)
    {
     scanf("%d", &a);
     if (a == 1234)
        break;
     append(&head1,&tail1,a);
    }
    print(head1);
    search(head1,&head2,&tail2);
    print(head2);

}

void print(struct node* head)
{
    struct node*p = head;
    printf("\nList is: ");
    while(p != NULL)
    {
        printf("%d->",p->data);
        p = p->next;
    }
    printf("\n");
}

void append(struct node** head, struct node** tail, int a)
{
    struct node* temp = (struct node*)malloc(sizeof(struct node));
    temp->next = NULL;
    temp->data = a;


    if ((*head) == NULL)
    {
        (*head) = temp;
```

```
        (*tail) = temp;
    }
    else
    {
        (*tail)->next = temp;
        (*tail) = temp;
    }


}

void search(struct node* head,struct node** h, struct node** tail2)
{
    int counter = 0;
    int key;

    printf("Enter the key: ");
    scanf("%d",&key);

    if (head == NULL)
    {
         printf("No elements in the list\n");
    }
    else
    {
        struct node* p = head;
        while(p != NULL)
        {
            if (p->data == key)
            {
                append(h,tail2,key);
                counter++;
            }
            p = p->next;

        }

          if (counter == 0)
             printf("Element not found");
```

```
    }


}
```



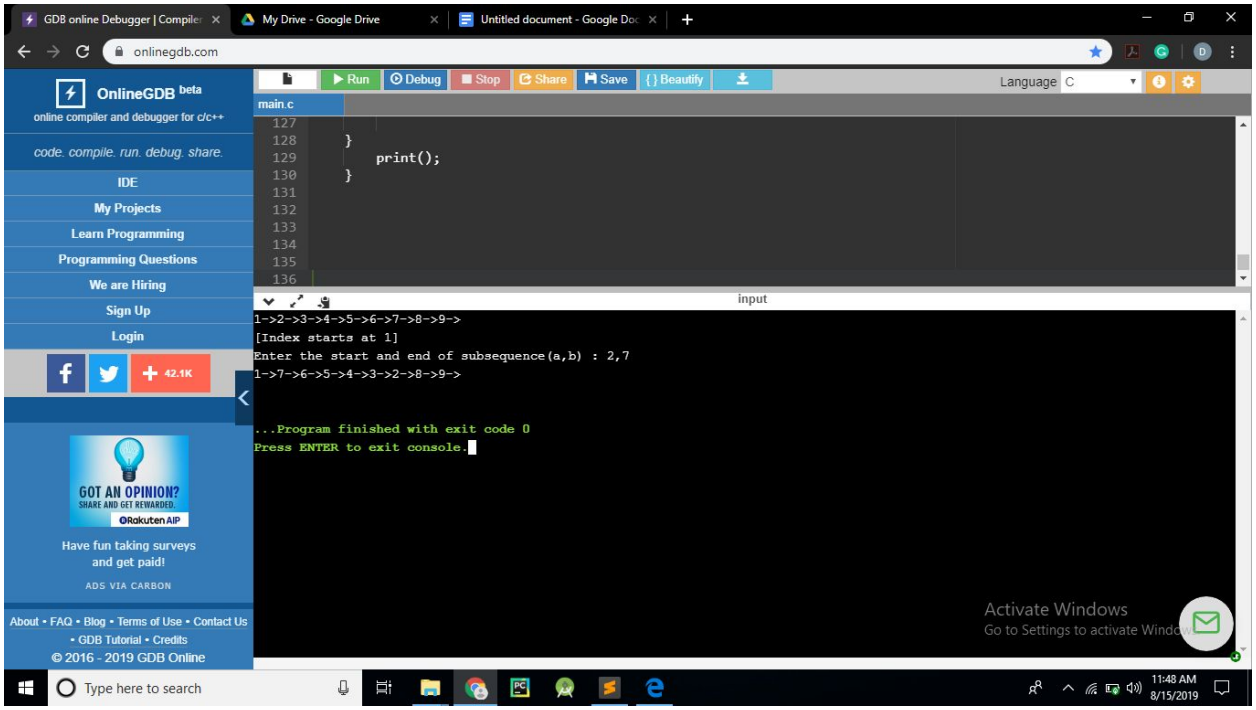
Q4 Subsequence of Linked List reversing

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

void append(int);
void print();
int length();
void reverse();

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

```
struct node* head = NULL;
struct node* tail = NULL;

int main()
{
    int arr[] = {1,2,3,4,5,6,7,8,9};

    for(int i = 0; i < 9 ; i++)
    {
        append(arr[i]);
    }
    print();

    reverse();


    return 0;
}

void append(int a)
{
     struct node* temp = (struct node*)malloc(sizeof(struct node*));
        temp->data = a;
        temp->next = NULL;

        if (head == NULL)
        {
            head = temp;
            tail = temp;
        }
        else
        {
            tail->next = temp;
            tail = temp;
        }
}

void print()
{
```

```
    struct node* p = head;
    while (p != NULL)
    {
        printf("%d->",p->data);
        p = p->next;
    }
    printf("\n");
}

int length()
{
    int count = 0;
    struct node* p = head;
    while(p != NULL)
    {
        count++;
        p = p->next;
    }
    return count;
}



void reverse()
{
    int a, b;
    printf("[Index starts at 1]\nEnter the start and end of subsequence(a,b) : ");
    scanf("%d,%d",&a,&b);

    if (a >= b)
        printf("Start index must be a lower value\n");
    else if (a > length() || b > length())
        printf("Subsequence must not exceed length of subsequence\n");
    else if (a <= 0 || b <= 0)
        printf("Indexes must be positive\n");
    else
    {
        struct node* p = head;
        struct node* q = NULL;
```

```
    int index = 1;
    while (index != a)
        {
            p = p->next;
            index++;
        }


    while(p != q)
    {
        q = head;
        index = 1;
        while (index != b)
         {
             q = q->next;
             index++;
         }
        int temp = q->data;
        q->data = p->data;
        p->data = temp;

        p = p->next;
        b--;



    }

}
    print();
}
```


GDB online Debugger | Compiler
My Drive - Google Drive
Untitled document - Google Doc
onlinegdb.com
OnlineGDB beta
online compiler and debugger for c/c++
code. compile. run. debug. share.
IDE
My Projects
Learn Programming
Programming Questions
We are Hiring
Sign Up
Login
42.1K
GOT AN OPINION?
SHARE AND GET REWARDED
Rakuten AIP
Have fun taking surveys and get paid!
ADS VIA CARBON
About • FAQ • Blog • Terms of Use • Contact Us • GDB Tutorial • Credits
© 2016 - 2019 GDB Online
Run
Debug
Stop
Share
Save
{ } Beautify
Language C
main.c
127
128 }
129 print();
130 }
131
132
133
134
135
136
input
1->2->3->4->5->6->7->8->9->
[Index starts at 1]
Enter the start and end of subsequence(a,b) : 2,7
1->7->6->5->4->3->2->8->9->
...Program finished with exit code 0
Press ENTER to exit console.
Activate Windows
Go to Settings to activate Windows
Type here to search
11:48 AM
8/15/2019