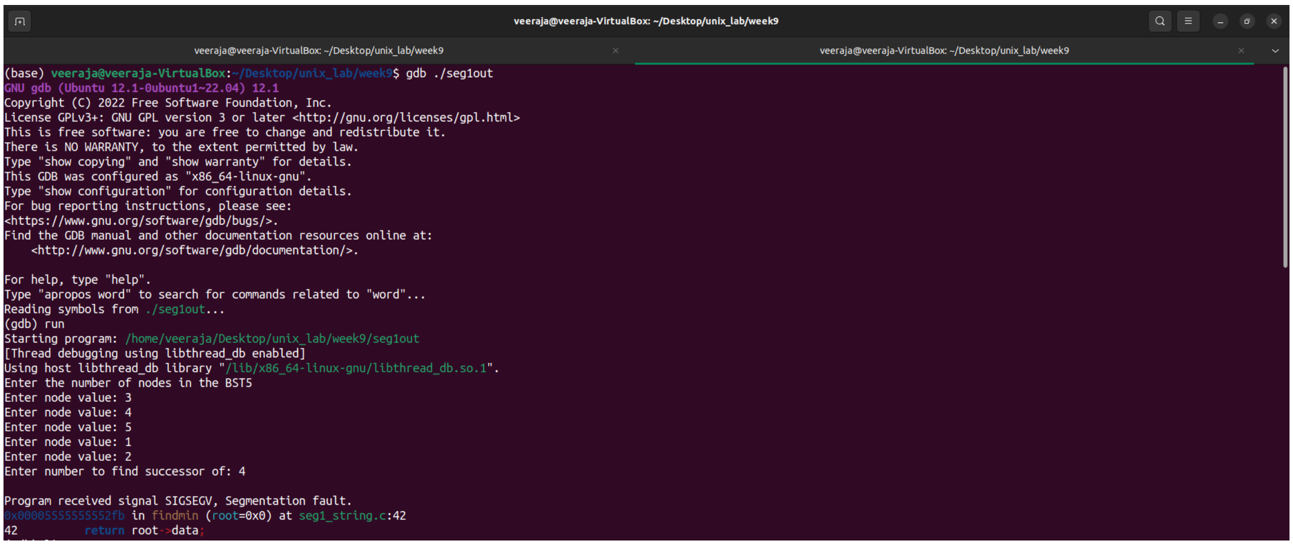
Week 9 GDB

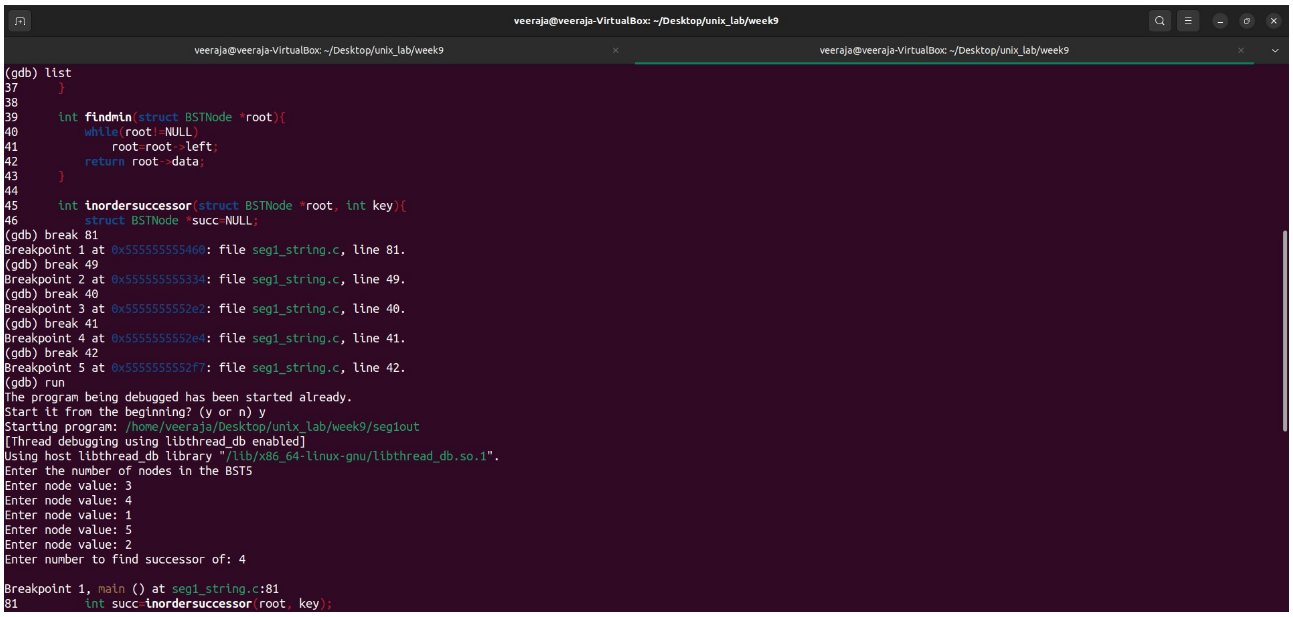
```
//seg1_string.c
#include <stdio.h>
#include <stdlib.h>

struct BSTNode{
    int data;
    struct BSTNode *left;
    struct BSTNode *right;
};

int insertnode(struct BSTNode **root, int data){
    if(!(*root)){
        struct BSTNode *newnode=(struct BSTNode *)malloc(sizeof(struct BSTNode));
        if(!newnode) return 0;
        newnode->data=data;
        newnode->left=newnode->right=NULL;
        *root=newnode;
        return 1;
    }
    else if((*root)->data>data){
        return insertnode(&(*root)->left, data);
    }
    else if((*root)->data<data){
        return insertnode(&(*root)->right, data);
    }
    else
        return 0;
}

int inorder(struct BSTNode *root){
    if(root){
        inorder(root->left);
        printf("%d ", root->data);
        inorder(root->right);
        return 1;
    }
    return 1;
}

int findmin(struct BSTNode *root){
    while(root->left)
        root=root->left;
    return root->data;
}

int inordersuccessor(struct BSTNode *root, int key){
    struct BSTNode *succ=NULL;
    while(root!=NULL){
        if (root->data == key && root->right != NULL)
            return findmin(root->right);
        else if (key < root->data)
        {
            succ = root;
            root = root->left;
        }
        else if (key > root->data)
            root = root->right;
        else
            break;
    }
    return succ->data;
}
```

```
int main(){
    int n;
    printf("Enter the number of nodes in the BST");
    scanf("%d",&n);

    struct BSTNode *root=NULL;
    while(n--){
        int elem;
        printf("Enter node value: ");
        scanf("%d", &elem);

        insertnode(&root, elem);
    }

    int key;
    printf("Enter number to find successor of: ");
    scanf("%d",&key);

    int succ=inordersuccessor(root, key);
    printf("Successor: %d", succ);

    return 0;
}
```

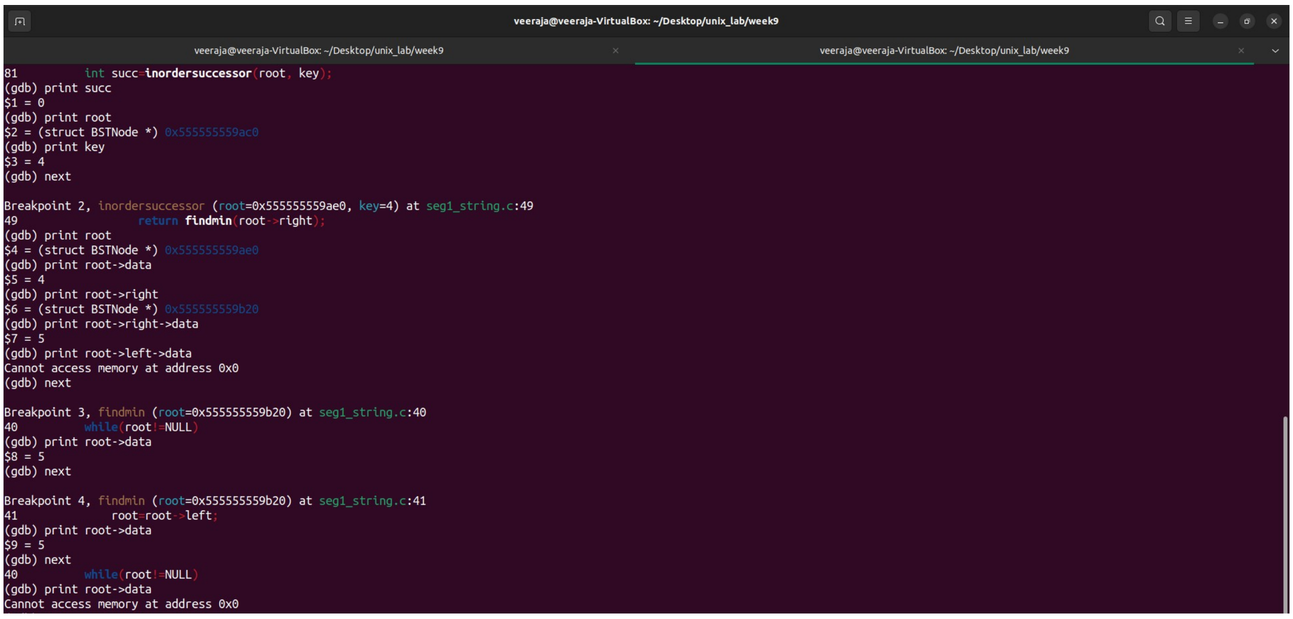
Screenshots


veeraja@veeraja-VirtualBox: ~/Desktop/unix_lab/week9
veeraja@veeraja-VirtualBox: ~/Desktop/unix_lab/week9
veeraja@veeraja-VirtualBox: ~/Desktop/unix_lab/week9

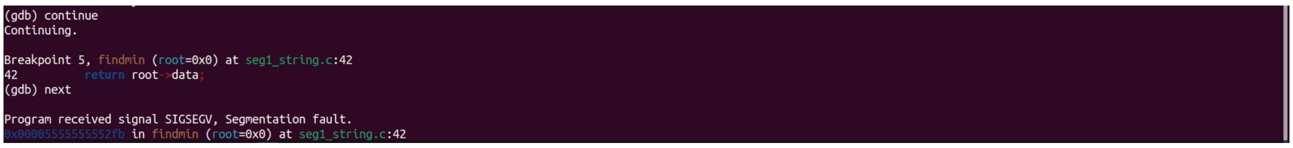
```
(base) veeraja@veeraja-VirtualBox:~/Desktop/unix_lab/week9$ gdb ./seg1out
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./seg1out...
(gdb) run
Starting program: /home/veeraja/Desktop/unix_lab/week9/seg1out
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Enter the number of nodes in the BST5
Enter node value: 3
Enter node value: 4
Enter node value: 5
Enter node value: 1
Enter node value: 2
Enter number to find successor of: 4

Program received signal SIGSEGV, Segmentation fault.
0x00005555555552fb in findmin (root=0x0) at seg1_string.c:42
42          return root->data;
```


veeraja@veeraja-VirtualBox: ~/Desktop/unix_lab/week9
veeraja@veeraja-VirtualBox: ~/Desktop/unix_lab/week9
veeraja@veeraja-VirtualBox: ~/Desktop/unix_lab/week9

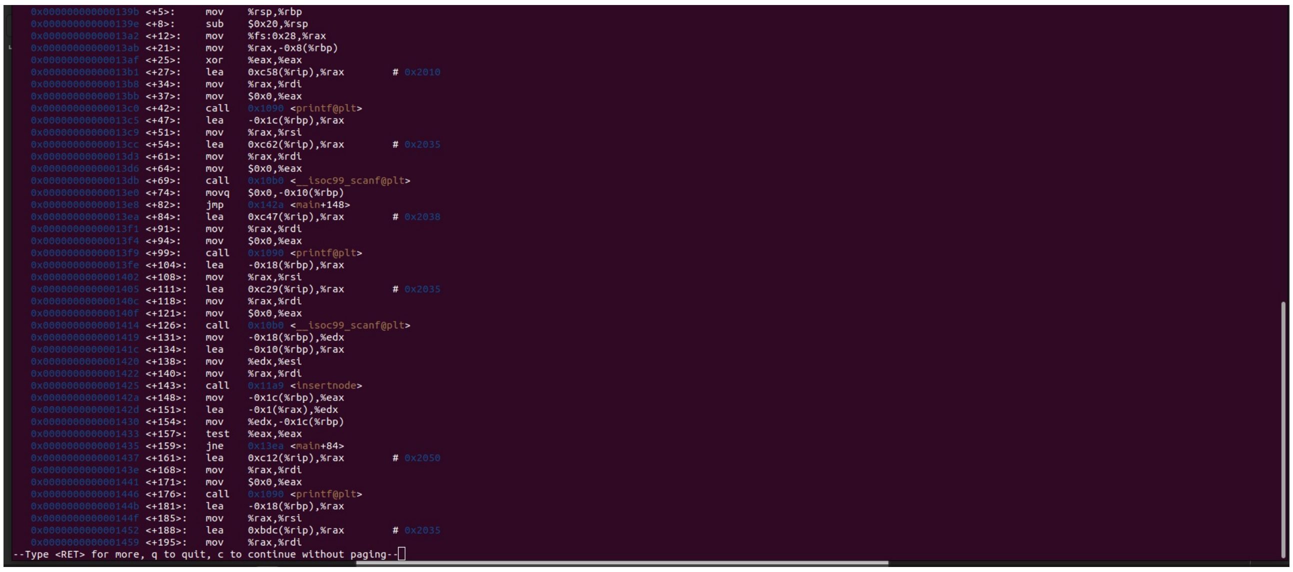
```
(gdb) list
37      }
38
39      int findmin(struct BSTNode *root){
40          while(root!=NULL)
41              root=root->left;
42          return root->data;
43      }
44
45      int inordersuccessor(struct BSTNode *root, int key){
46          struct BSTNode *succ=NULL;
(gdb) break 81
Breakpoint 1 at 0x555555555460: file seg1_string.c, line 81.
(gdb) break 49
Breakpoint 2 at 0x555555555334: file seg1_string.c, line 49.
(gdb) break 40
Breakpoint 3 at 0x5555555552e2: file seg1_string.c, line 40.
(gdb) break 41
Breakpoint 4 at 0x5555555552e4: file seg1_string.c, line 41.
(gdb) break 42
Breakpoint 5 at 0x5555555552f7: file seg1_string.c, line 42.
(gdb) run
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/veeraja/Desktop/unix_lab/week9/seg1out
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Enter the number of nodes in the BST5
Enter node value: 3
Enter node value: 4
Enter node value: 1
Enter node value: 5
Enter node value: 2
Enter number to find successor of: 4

Breakpoint 1, main () at seg1_string.c:81
81          int succ=inordersuccessor(root, key);
```


veeraja@veeraja-VirtualBox: ~/Desktop/unix_lab/week9
veeraja@veeraja-VirtualBox: ~/Desktop/unix_lab/week9
veeraja@veeraja-VirtualBox: ~/Desktop/unix_lab/week9

```
81          int succ=inordersuccessor(root, key);
(gdb) print succ
$1 = 0
(gdb) print root
$2 = (struct BSTNode *) 0x555555559ac0
(gdb) print key
$3 = 4
(gdb) next

Breakpoint 2, inordersuccessor (root=0x555555559ae0, key=4) at seg1_string.c:49
49                  return findmin(root->right);
(gdb) print root
$4 = (struct BSTNode *) 0x555555559ae0
(gdb) print root->data
$5 = 4
(gdb) print root->right
$6 = (struct BSTNode *) 0x555555559b20
(gdb) print root->right->data
$7 = 5
(gdb) print root->left->data
Cannot access memory at address 0x0
(gdb) next

Breakpoint 3, findmin (root=0x555555559b20) at seg1_string.c:40
40          while(root!=NULL)
(gdb) print root->data
$8 = 5
(gdb) next

Breakpoint 4, findmin (root=0x555555559b20) at seg1_string.c:41
41              root=root->left;
(gdb) print root->data
$9 = 5
(gdb) next
40          while(root!=NULL)
(gdb) print root->data
Cannot access memory at address 0x0
```



```
(gdb) continue
Continuing.

Breakpoint 5, findmin (root=0x0) at seg1_string.c:42
42          return root->data;
(gdb) next

Program received signal SIGSEGV, Segmentation fault.
0x00005555555552fb in findmin (root=0x0) at seg1_string.c:42
```



```
   0x000000000000139b <+5>:     mov    %rsp,%rbp
   0x000000000000139e <+8>:     sub    $0x20,%rsp
   0x00000000000013a2 <+12>:    mov    %fs:0x28,%rax
   0x00000000000013ab <+21>:    mov    %rax,-0x8(%rbp)
   0x00000000000013af <+25>:    xor    %eax,%eax
   0x00000000000013b1 <+27>:    lea    0xc58(%rip),%rax        # 0x2010
   0x00000000000013b8 <+34>:    mov    %rax,%rdi
   0x00000000000013bb <+37>:    mov    $0x0,%eax
   0x00000000000013c0 <+42>:    call   0x1090 <printf@plt>
   0x00000000000013c5 <+47>:    lea    -0x1c(%rbp),%rax
   0x00000000000013c9 <+51>:    mov    %rax,%rsi
   0x00000000000013cc <+54>:    lea    0xc62(%rip),%rax        # 0x2035
   0x00000000000013d3 <+61>:    mov    %rax,%rdi
   0x00000000000013d6 <+64>:    mov    $0x0,%eax
   0x00000000000013db <+69>:    call   0x10b0 <__isoc99_scanf@plt>
   0x00000000000013e0 <+74>:    movq   $0x0,-0x10(%rbp)
   0x00000000000013e8 <+82>:    jmp    0x142a <main+148>
   0x00000000000013ea <+84>:    lea    0xc47(%rip),%rax        # 0x2038
   0x00000000000013f1 <+91>:    mov    %rax,%rdi
   0x00000000000013f4 <+94>:    mov    $0x0,%eax
   0x00000000000013f9 <+99>:    call   0x1090 <printf@plt>
   0x00000000000013fe <+104>:   lea    -0x18(%rbp),%rax
   0x0000000000001402 <+108>:   mov    %rax,%rsi
   0x0000000000001405 <+111>:   lea    0xc29(%rip),%rax        # 0x2035
   0x000000000000140c <+118>:   mov    %rax,%rdi
   0x000000000000140f <+121>:   mov    $0x0,%eax
   0x0000000000001414 <+126>:   call   0x10b0 <__isoc99_scanf@plt>
   0x0000000000001419 <+131>:   mov    -0x18(%rbp),%edx
   0x000000000000141c <+134>:   lea    -0x10(%rbp),%rax
   0x0000000000001420 <+138>:   mov    %edx,%esi
   0x0000000000001422 <+140>:   mov    %rax,%rdi
   0x0000000000001425 <+143>:   call   0x11a9 <insertnode>
   0x000000000000142a <+148>:   mov    -0x1c(%rbp),%eax
   0x000000000000142d <+151>:   lea    -0x1(%rax),%edx
   0x0000000000001430 <+154>:   mov    %edx,-0x1c(%rbp)
   0x0000000000001433 <+157>:   test   %eax,%eax
   0x0000000000001435 <+159>:   jne    0x13ea <main+84>
   0x0000000000001437 <+161>:   lea    0xc12(%rip),%rax        # 0x2050
   0x000000000000143e <+168>:   mov    %rax,%rdi
   0x0000000000001441 <+171>:   mov    $0x0,%eax
   0x0000000000001446 <+176>:   call   0x1090 <printf@plt>
   0x000000000000144b <+181>:   lea    -0x18(%rbp),%rax
   0x000000000000144f <+185>:   mov    %rax,%rsi
   0x0000000000001452 <+188>:   lea    0xbdc(%rip),%rax        # 0x2035
   0x0000000000001459 <+195>:   mov    %rax,%rdi
--Type <RET> for more, q to quit, c to continue without paging--
```

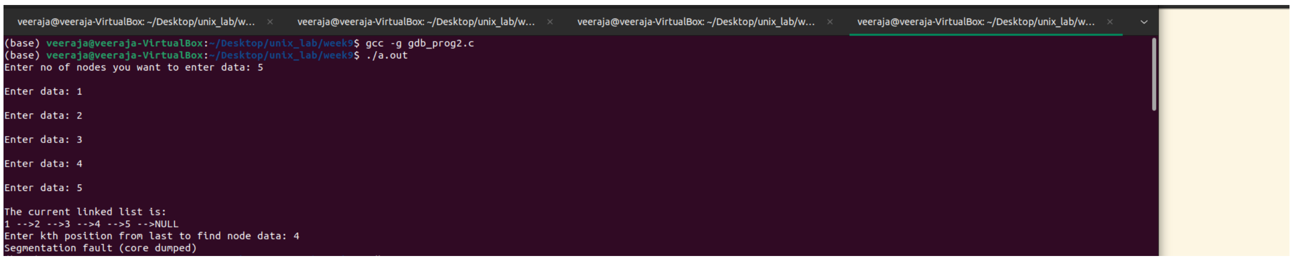
```
//gdb_prog2.c
#include <stdio.h>
#include <stdlib.h>

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



struct node *head;

int initList(struct node **head){
	*head=NULL;
	return 1;
}
```

```
int search(struct node **head, int data, struct node **ptrToKey, int *pos){
        if (*head==NULL) return 0;
        *pos=1;
        struct node *ptr=*head;
        for (;ptr!=NULL && ptr->data!=data;ptr=ptr->next){
                *pos=(*pos)+1;
        }
                *ptrToKey=ptr;

        if (!ptr) return 0;
        return 1;
}


int insert(struct node **head, int position, int data){
        struct node *newnode=(struct node *)malloc(sizeof(struct node));
        if (newnode==NULL) return 0;

        newnode->data=data;
        if (position==1){
                newnode->next=*head;
                *head = newnode;
                return 1;
        }
        //to make sure there are no duplicate insertions we search if given data is already present in
linked list
        struct node *ptrToKey=NULL;
        int pos=0;
        if (!search(head, data,&ptrToKey, &pos)){
                struct node *ptr=*head;
                for (int i=1; i<position-1 && ptr!=NULL;i++)
                ptr=ptr->next;

            if (ptr==NULL) return 0;
            else{
                newnode->next=ptr->next;
                ptr->next=newnode;
                return 1;
            }
        }
        else{
            printf("Element already present in address: %p \n",ptrToKey);
            return 0;
        }
}

int traverse(struct node *head){
        if (!head){
            printf("NULL \n");
            return 1;
        }

        for (struct node *ptr=head;ptr!=NULL;ptr=ptr->next)
            printf("%d -->",ptr->data);
        printf("NULL \n");
        return 1;
}

int kFromLast(struct node *head, int k ,int *data){
        if(!head) return 0;
        struct node *fast=head;
        struct node *slow=NULL;
        int i=1;
```

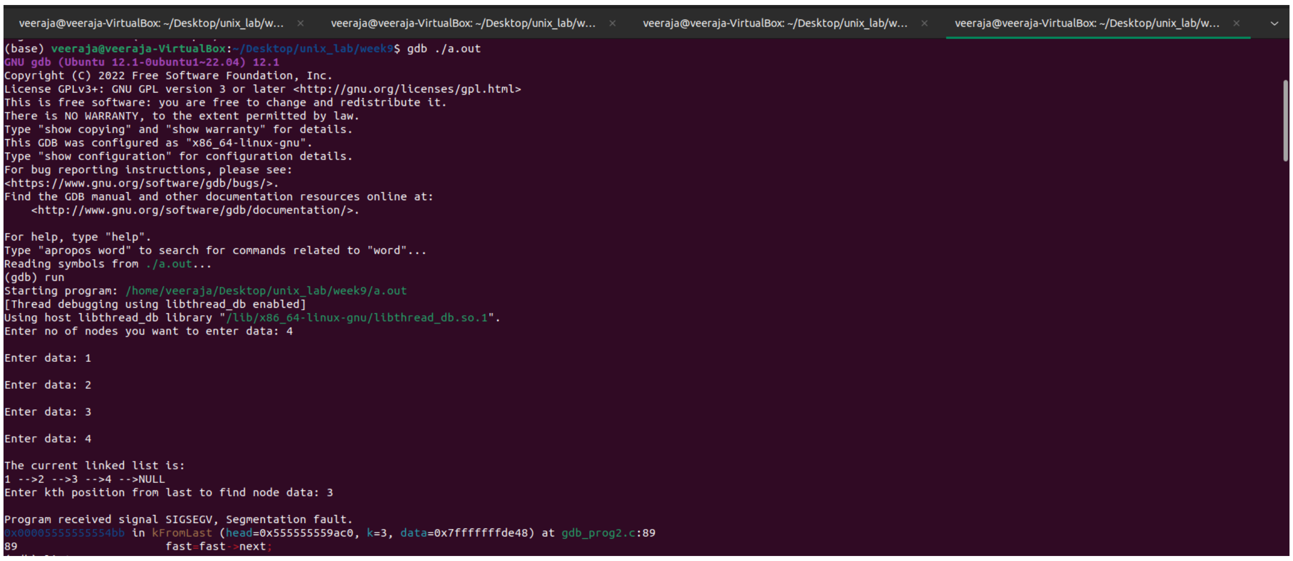
```
	while(fast!=NULL && i<=k){
		fast=fast->next;
		i++;
	}
	if(fast==NULL && i<k) return 0;
	slow=head;
	while(fast!=NULL){
		slow=slow->next;
		fast=fast->next;
	}
	*data=slow->data;
	return 1;
}

int main(){
	struct node *head;
	initList(&head);
	int n;

	printf("Enter no of nodes you want to enter data: ");
	scanf("%d",&n);
	int pos=1;
	while (n--){
		int data;
		printf("\nEnter data: ");
		scanf("%d",&data);
		if (!insert(&head,pos++,data)) return 0;
	}

	printf("\nThe current linked list is:\n");
	traverse(head);
```

```c
    int k, data;
    printf("Enter kth position from last to find node data: ");
    scanf("%d", &k);

    kFromLast(head, k, &data);
    printf("Data: %d\n",data);
    return 0;
}
```

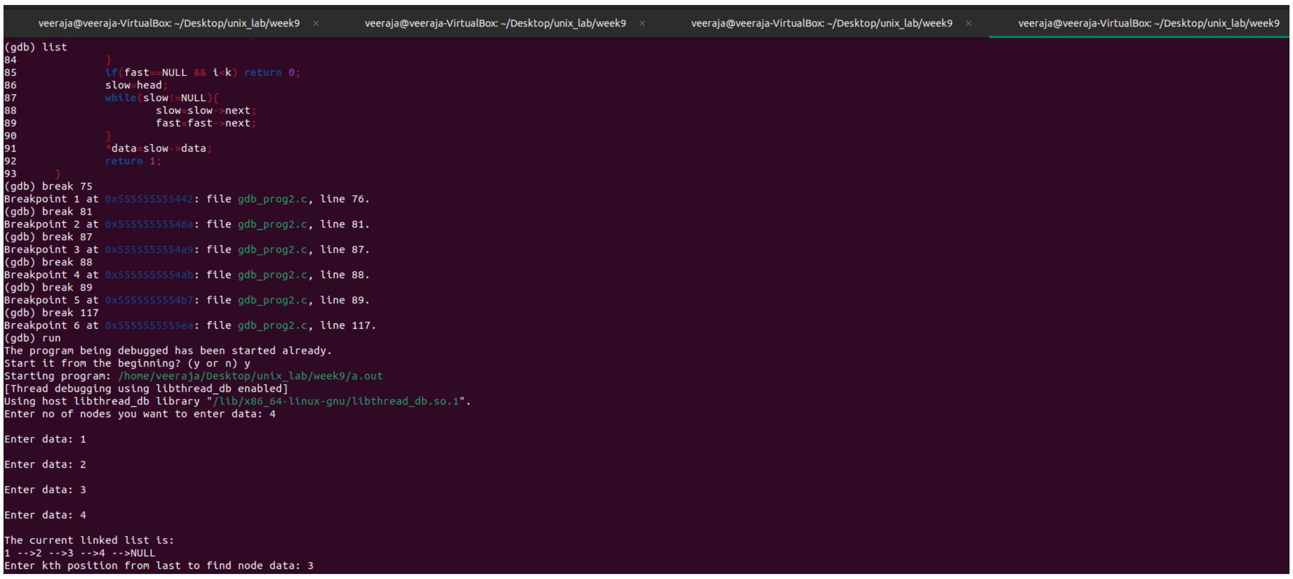
Outputs:



```
(base) veeraja@veeraja-VirtualBox:~/Desktop/unix_lab/week9$ gcc -g gdb_prog2.c
(base) veeraja@veeraja-VirtualBox:~/Desktop/unix_lab/week9$ ./a.out
Enter no of nodes you want to enter data: 5

Enter data: 1

Enter data: 2

Enter data: 3

Enter data: 4

Enter data: 5

The current linked list is:
1 -->2 -->3 -->4 -->5 -->NULL
Enter kth position from last to find node data: 4
Segmentation fault (core dumped)
```

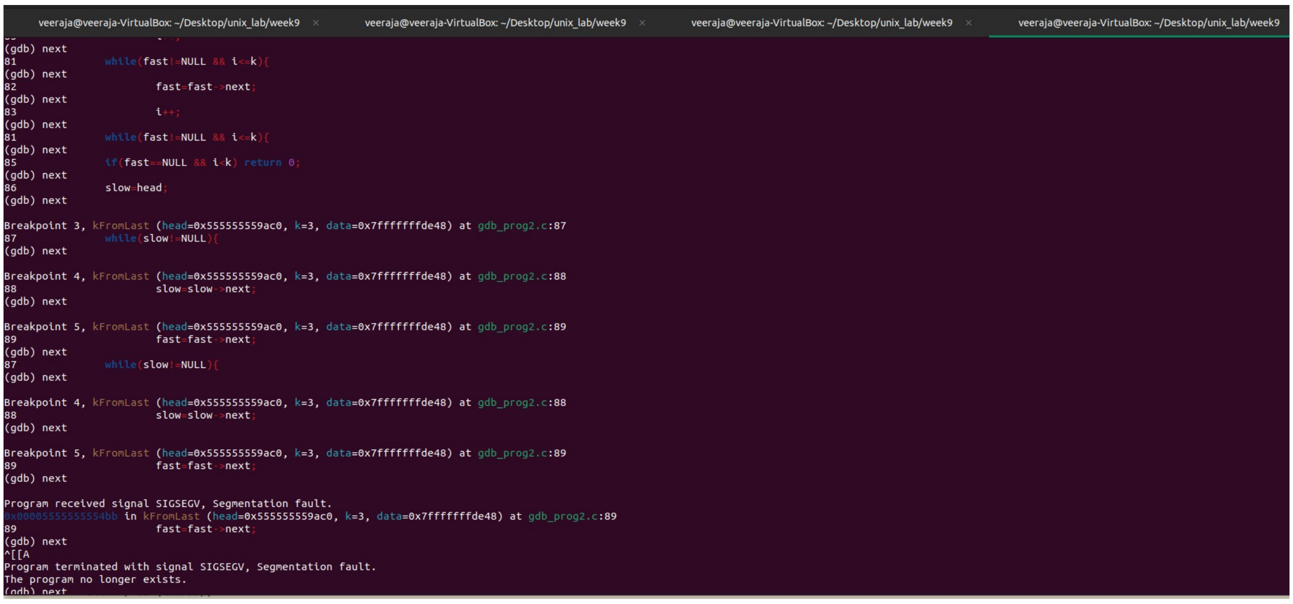


```
(base) veeraja@veeraja-VirtualBox:~/Desktop/unix_lab/week9$ gdb ./a.out
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./a.out...
(gdb) run
Starting program: /home/veeraja/Desktop/unix_lab/week9/a.out
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Enter no of nodes you want to enter data: 4

Enter data: 1

Enter data: 2

Enter data: 3

Enter data: 4

The current linked list is:
1 -->2 -->3 -->4 -->NULL
Enter kth position from last to find node data: 3

Program received signal SIGSEGV, Segmentation fault.
0x00005555555554bb in kFromLast (head=0x555555559ac0, k=3, data=0x7fffffffde48) at gdb_prog2.c:89
89                      fast=fast->next;
```



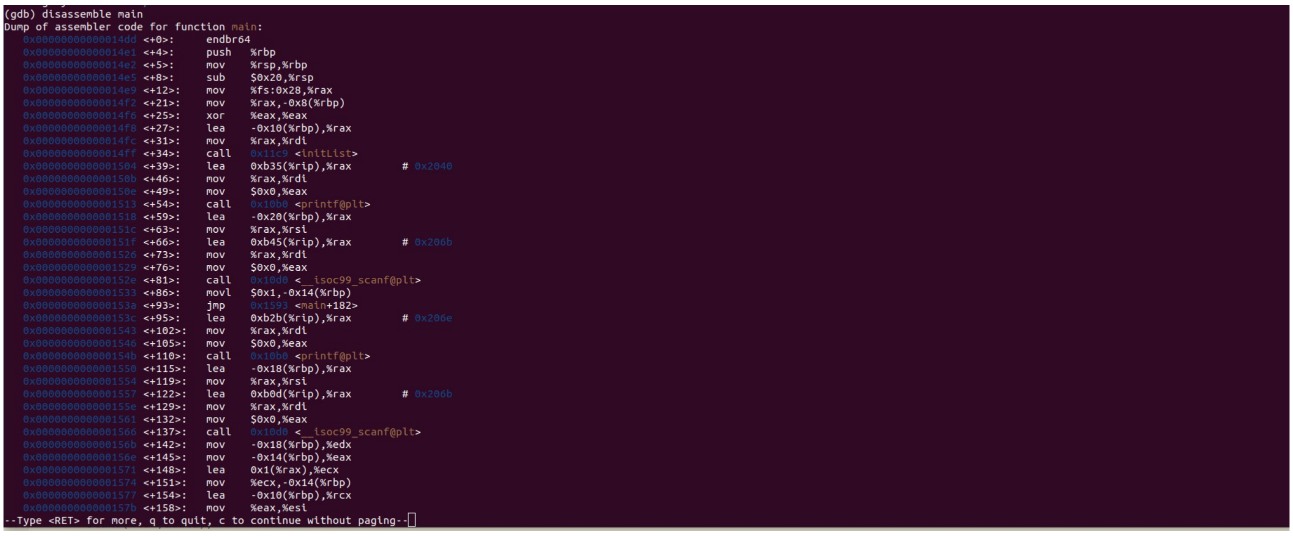
veeraja@veeraja-VirtualBox: ~/Desktop/unix_lab/week9 × veeraja@veeraja-VirtualBox: ~/Desktop/unix_lab/week9 × veeraja@veeraja-VirtualBox: ~/Desktop/unix_lab/week9 × veeraja@veeraja-VirtualBox: ~/Desktop/unix_lab/week9

```
(gdb) list
84              }
85              if(fast==NULL && i<k) return 0;
86              slow=head;
87              while(slow!=NULL){
88                      slow=slow->next;
89                      fast=fast->next;
90              }
91              *data=slow->data;
92              return 1;
93      }
(gdb) break 75
Breakpoint 1 at 0x555555555442: file gdb_prog2.c, line 76.
(gdb) break 81
Breakpoint 2 at 0x55555555546a: file gdb_prog2.c, line 81.
(gdb) break 87
Breakpoint 3 at 0x5555555554a9: file gdb_prog2.c, line 87.
(gdb) break 88
Breakpoint 4 at 0x5555555554ab: file gdb_prog2.c, line 88.
(gdb) break 89
Breakpoint 5 at 0x5555555554b7: file gdb_prog2.c, line 89.
(gdb) break 117
Breakpoint 6 at 0x5555555555ea: file gdb_prog2.c, line 117.
(gdb) run
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/veeraja/Desktop/unix_lab/week9/a.out
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Enter no of nodes you want to enter data: 4

Enter data: 1

Enter data: 2

Enter data: 3

Enter data: 4

The current linked list is:
1 -->2 -->3 -->4 -->NULL
Enter kth position from last to find node data: 3
```



veeraja@veeraja-VirtualBox: ~/Desktop/unix_lab/week9 × veeraja@veeraja-VirtualBox: ~/Desktop/unix_lab/week9 × veeraja@veeraja-VirtualBox: ~/Desktop/unix_lab/week9 × veeraja@veeraja-VirtualBox: ~/Desktop/unix_lab/week9

```
(gdb) next
81              while(fast!=NULL && i<=k){
(gdb) next
82                      fast=fast->next;
(gdb) next
83                      i++;
(gdb) next
81              while(fast!=NULL && i<=k){
(gdb) next
85              if(fast==NULL && i<k) return 0;
(gdb) next
86              slow=head;
(gdb) next

Breakpoint 3, kFromLast (head=0x555555559ac0, k=3, data=0x7fffffffde48) at gdb_prog2.c:87
87              while(slow!=NULL){
(gdb) next

Breakpoint 4, kFromLast (head=0x555555559ac0, k=3, data=0x7fffffffde48) at gdb_prog2.c:88
88                      slow=slow->next;
(gdb) next

Breakpoint 5, kFromLast (head=0x555555559ac0, k=3, data=0x7fffffffde48) at gdb_prog2.c:89
89                      fast=fast->next;
(gdb) next
87              while(slow!=NULL){
(gdb) next

Breakpoint 4, kFromLast (head=0x555555559ac0, k=3, data=0x7fffffffde48) at gdb_prog2.c:88
88                      slow=slow->next;
(gdb) next

Breakpoint 5, kFromLast (head=0x555555559ac0, k=3, data=0x7fffffffde48) at gdb_prog2.c:89
89                      fast=fast->next;
(gdb) next

Program received signal SIGSEGV, Segmentation fault.
0x00005555555554bb in kFromLast (head=0x555555559ac0, k=3, data=0x7fffffffde48) at gdb_prog2.c:89
89                      fast=fast->next;
(gdb) next
^[[A
Program terminated with signal SIGSEGV, Segmentation fault.
The program no longer exists.
(gdb) next
```



```
(gdb) disassemble main
Dump of assembler code for function main:
   0x00000000000014dd <+0>:     endbr64
   0x00000000000014e1 <+4>:     push   %rbp
   0x00000000000014e2 <+5>:     mov    %rsp,%rbp
   0x00000000000014e5 <+8>:     sub    $0x20,%rsp
   0x00000000000014e9 <+12>:    mov    %fs:0x28,%rax
   0x00000000000014f2 <+21>:    mov    %rax,-0x8(%rbp)
   0x00000000000014f6 <+25>:    xor    %eax,%eax
   0x00000000000014f8 <+27>:    lea    -0x10(%rbp),%rax
   0x00000000000014fc <+31>:    mov    %rax,%rdi
   0x00000000000014ff <+34>:    call   0x11c9 <initList>
   0x0000000000001504 <+39>:    lea    0xb35(%rip),%rax        # 0x2040
   0x000000000000150b <+46>:    mov    %rax,%rdi
   0x000000000000150e <+49>:    mov    $0x0,%eax
   0x0000000000001513 <+54>:    call   0x10b0 <printf@plt>
   0x0000000000001518 <+59>:    lea    -0x20(%rbp),%rax
   0x000000000000151c <+63>:    mov    %rax,%rsi
   0x000000000000151f <+66>:    lea    0xb45(%rip),%rax        # 0x206b
   0x0000000000001526 <+73>:    mov    %rax,%rdi
   0x0000000000001529 <+76>:    mov    $0x0,%eax
   0x000000000000152e <+81>:    call   0x10d0 <__isoc99_scanf@plt>
   0x0000000000001533 <+86>:    movl   $0x1,-0x14(%rbp)
   0x000000000000153a <+93>:    jmp    0x1593 <main+182>
   0x000000000000153c <+95>:    lea    0xb2b(%rip),%rax        # 0x206e
   0x0000000000001543 <+102>:   mov    %rax,%rdi
   0x0000000000001546 <+105>:   mov    $0x0,%eax
   0x000000000000154b <+110>:   call   0x10b0 <printf@plt>
   0x0000000000001550 <+115>:   lea    -0x18(%rbp),%rax
   0x0000000000001554 <+119>:   mov    %rax,%rsi
   0x0000000000001557 <+122>:   lea    0xb0d(%rip),%rax        # 0x206b
   0x000000000000155e <+129>:   mov    %rax,%rdi
   0x0000000000001561 <+132>:   mov    $0x0,%eax
   0x0000000000001566 <+137>:   call   0x10d0 <__isoc99_scanf@plt>
   0x000000000000156b <+142>:   mov    -0x18(%rbp),%edx
   0x000000000000156e <+145>:   mov    -0x14(%rbp),%eax
   0x0000000000001571 <+148>:   lea    0x1(%rax),%ecx
   0x0000000000001574 <+151>:   mov    %ecx,-0x14(%rbp)
   0x0000000000001577 <+154>:   lea    -0x10(%rbp),%rcx
   0x000000000000157b <+158>:   mov    %eax,%esi
--Type <RET> for more, q to quit, c to continue without paging--
```