Lab Program 1 :

WAP to simulate the working of stack using an array with the following A: PUSH, B: POP & display. The program should print appropriate

```
#include<stdio.h>
#include<process.h>
#include<conio.h>
#define STACK_SIZE 5
int top=-1;
int s[10];
int item;
void push()
{ if(top==STACK_SIZE-1)
  { printf("stack overflow\n");
    return;
  }
  top=top+1;
  s[top]=item;
}
int pop()
{ if(top==-1) return -1;
  return s[top--];
}
void display()
{ int i;
  if(top==-1)
  { printf("stack is empty\n");
  return;
}
  printf("contents of the stack\n");
```

```
for(i=top; i>=0; i--)
{
  printf("%d\n", s[i]);
}
}
int main()
{ int item_deleted;
char choice;
  clrscr();
  for(;;)
{ printf("\n A: push \n B: pop \n C: display \n D: exit");
  printf("enter the choice \n");
  scanf("%c", &choice);
  switch(choice)
{ case 'A':
  case 'A': printf("enter the item to be inserted \n");
  scanf("%d", &item);
  push()
  break;
case 'B': item_deleted=pop(); (-1)
  if(item_deleted == -1)
  printf("stack is empty\n");
  else
  printf("item deleted is %d\n", item_deleted);
  break;
case 'C': display();
  break;
  default: exit(0);
}
}
getch();
}
```

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\LP 1\push and pop.cpp - Dev-C++ 5.11

File Edit Search View Project Execute Tools AStyle Window Help

TDM-GCC 4.9.2 64-bit Release

(globals)

push and pop.cpp

```
1  #include<stdio.h>
2  #include<process.h>
3  #include<conio.h>
4  #define STACK_SIZE 5
5  int top=-1;
6  int s[10];
7  int item;
8  void push ()
9  {
10     if(top==STACK_SIZE-1){printf("stack overflow\n");
11     return;
12     }
13     top=top+1;
14     s[top]=item;
15 }
16 int pop()
17 { if(top==-1)return -1;
18 return s[top--];
19 }
20 void display()
21 { int i;
22 if(top==-1)
23 {printf(" Stack is empty \n");
24 return;
25 }
26 printf("contents of the stack\n");
27 for(i=top;i>=0;i--)
28 { printf("%d\n",s[i]);
29 }
30 }
31 int main()
32 {
33     int item_deleted;
34     int choice;
35     system("cls");
36 for(;;)
37     {
38         printf("\n 1:Push\n 2:Pop\n 3:Display\n 4:Exit\n");
```

Compiler Resources Compile Log Debug Find Results

Line: 11 Col: 12 Sel: 0 Lines: 61 Length: 1123 Insert Done parsing in 0.11 seconds

Type here to search

ENG 14:32 05-10-2020

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\LP 1\push and pop.cpp - Dev-C++ 5.11

File Edit Search View Project Execute Tools AStyle Window Help

TDM-GCC 4.9.2 64-bit Release

(globals)

push and pop.cpp

```
25  }
26  printf("contents of the stack\n");
27  for(i=top;i>=0;i--)
28  { printf("%d\n",s[i]);
29  }
30  }
31  int main()
32  {
33      int item_deleted;
34      int choice;
35      system("cls");
36  for(;;)
37      {
38          printf("\n 1:Push\n 2:Pop\n 3:Display\n 4:Exit\n");
39          printf(" Enter the choice\n");
40          scanf("%d",&choice);
41          switch(choice)
42          {
43              case 1: printf("enter the item to be inserted \n");
44                      scanf("%d\n",&item);
45                      push();
46                      break;
47              case 2: item_deleted=pop(); (-1);
48                      if(item_deleted==-1)
49                      {printf("Stack is empty\n");
50                  }
51                      else
52                      printf("item deleted is %d\n",item_deleted);
53                      break;
54              case 3: display();
55                      break;
56              default: exit(0);
57          }
58      }
59      getch();
60  return 0;
61  }
```

Compiler Resources Compile Log Debug Find Results

Line: 11 Col: 12 Sel: 0 Lines: 61 Length: 1123 Insert Done parsing in 0.11 seconds

Type here to search

ENG 14:32 05-10-2020

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\LP 1\push and pop.exe

```
 1:Push
 2:Pop
 3:Display
 4:Exit
 Enter the choice
1
enter the item to be inserted
12
1

 1:Push
 2:Pop
 3:Display
 4:Exit
 Enter the choice
enter the item to be inserted
12
1

 1:Push
 2:Pop
 3:Display
 4:Exit
 Enter the choice
enter the item to be inserted
15
1

 1:Push
 2:Pop
 3:Display
 4:Exit
 Enter the choice
enter the item to be inserted
43
1

 1:Push
 2:Pop
 3:Display
 4:Exit
 Enter the choice
enter the item to be inserted
32
1

 1:Push
 2:Pop
 3:Display
```

Type here to search

ENG 14:32 05-10-2020

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\LP 1\push and pop.exe

```
 1:Push
 2:Pop
 3:Display
 4:Exit
 Enter the choice
enter the item to be inserted
2

2
stack overflow

 1:Push
 2:Pop
 3:Display
 4:Exit
 Enter the choice
item deleted is 32

 1:Push
 2:Pop
 3:Display
 4:Exit
 Enter the choice
3
contents of the stack
43
15
12
12

 1:Push
 2:Pop
 3:Display
 4:Exit
 Enter the choice
4

--------------------------------
Process exited after 170.3 seconds with return value 0
Press any key to continue . . .
```

Lab Program 2:-

2) Write a program a given valid parenthesized infix arithmetic expression to postfix expression. The expression consists of single character operands and binary operators + (plus), - (minus), * (multiply) & / (divide).

A.

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
#include<process.h>
int F(char symbol)
{ switch (symbol)
  { case '+':
    case '-': return 2;
    case '*':
    case '/': return 4;
    case '^':
    case '$': return 5;
    case '(': return 0;
    case '#': return -1;
    default : return 8;
  }}
  int G(char symbol)
{ switch(symbol)
  { case '+':
    case '-': return 1;
    case '*':
    case '/': return 3;
    case '^':
    case '$': return 6;
    case '(': return 9;
    case ')': return 0;
    default: return 7;
  }}
```

P.T.O.

```
void infix_postfix(char infix[], char postfix[])
{
 int top, i, j;
 char s[30], symbol;
 top = -1;
s[++top] = '#';
j=0;
for(i=0; i<strlen(infix); i++)
{
 symbol = infix[i];
 while(F(s[top])>G(symbol))
{
 postfix[j] = s[top--];
 j++;
}
 if(F(s[top]) != G(symbol))
 s[++top] = symbol;
else
 top--;
}
while(s[top] != '#')
{
 postfix[j++]= s[top--];
}
 postfix[j] = '\0';
}
int main()
{ int t;
 char infix[20];
 char postfix[20];
 system("cls");
 printf("Enter the valid infix expression.\n");
 scanf("%s", infix);
for(t=0; t< strlen(infix); t++){
```

```
if(infix[i]=='+' || infix[i]=='-' || infix[i]=='*' ||
 infix[i]=='/' || infix[i]=='^' || infix[i]=='(')
{
if(infix[i+1]=='+'||infix[i+1]=='-' || infix[i+1]=='*'
 || infix[i+1]=='/'||infix[i+1]=='^' || infix[i+1]==')')
{
 printf("Invalid"); exit(0);
}}}

infix_postfix(infix, postfix);
printf("the postfix exp is \n");
getch();
return 0;
}
```

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\LP 2\dev\LP_2.cpp - Dev-C++ 5.11

File Edit Search View Project Execute Tools AStyle Window Help

TDM-GCC 4.9.2 64-bit Release

(globals)

LP_2.cpp

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
#include<process.h>
int F(char symbol)
{
switch(symbol)
{
case'+':
case'-':return 2;
case'*':
case'/':return 4;
case'^':
case'$':return 5;
case'(':return 0;
case'#':return -1;
default:return 8;
}
}
int G(char symbol)
{
switch(symbol)
{
case'+':
case'-':return 1;
case'*':
case'/':return 3;
case'^':
case'$':return 6;
case'(':return 9;
case')':return 0;
default:return 7;
}
}
void infix_postfix(char infix[],char postfix[])
{
int top,i,j;
char s[30],symbol;
```

Compiler Resources Compile Log Debug Find Results

Line: 58 Col: 2 Sel: 0 Lines: 83 Length: 1356 Insert Done parsing in 0.015 seconds

Type here to search

ENG 15:52 05-10-2020

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\LP 2\dev\LP_2.cpp - Dev-C++ 5.11

File Edit Search View Project Execute Tools AStyle Window Help

TDM-GCC 4.9.2 64-bit Release

(globals)

LP_2.cpp

```
35  void infix_postfix(char infix[],char postfix[])
36  {
37  int top,i,j;
38  char s[30],symbol;
39  top=-1;
40  s[++top]='#';
41  j=0;
42  for(i=0;i<strlen(infix);i++)
43  {
44  symbol=infix[i];
45  while(F(s[top])>G(symbol))
46  {
47  postfix[j]=s[top--];
48  j++;
49  }
50  if(F(s[top])!=G(symbol))
51  s[++top]=symbol;
52  else
53  top--;
54  }
55  while(s[top]!='#')
56  {
57  postfix[j++]=s[top--];
58  }
59  postfix[j]='\0';
60  }
61  int main()
62  {int t;
63  char infix[20];
64  char postfix[20];
65  system("cls");
66  printf("Enter the valid infix expression\n");
67  scanf("%s",infix);
68  for(t=0;t<strlen(infix);t++)
69  {
70  if(infix[t]=='+'||infix[t]=='-'||infix[t]=='*'||infix[t]=='/'||infix[t]=='^'||infix[t]=='(')
71  {
72  if(infix[t+1]=='+'||infix[t+1]=='-'||infix[t+1]=='*'||infix[t+1]=='/'||infix[t+1]=='^'||infix[t+1]==')')
```

Compiler Resources Compile Log Debug Find Results

Line: 58 Col: 2 Sel: 0 Lines: 83 Length: 1356 Insert Done parsing in 0.015 seconds

Type here to search

ENG 15:53 05-10-2020

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\LP 2\dev\LP_2.cpp - Dev-C++ 5.11

File Edit Search View Project Execute Tools AStyle Window Help

TDM-GCC 4.9.2 64-bit Release

(globals)

LP_2.cpp

```
47  postfix[j]=s[top--];
48  j++;
49  }
50  if(F(s[top])!=G(symbol))
51  s[++top]=symbol;
52  else
53  top--;
54  }
55  while(s[top]!='#')
56  {
57  postfix[j++]=s[top--];
58  }
59  postfix[j]='\0';
60  }
61  int main()
62  {int t;
63  char infix[20];
64  char postfix[20];
65  system("cls");
66  printf("Enter the valid infix expression\n");
67  scanf("%s",infix);
68  for(t=0;t<strlen(infix);t++)
69  {
70  if(infix[t]=='+'||infix[t]=='-'||infix[t]=='*'||infix[t]=='/'||infix[t]=='^'||infix[t]=='(')
71  {
72  if(infix[t+1]=='+'||infix[t+1]=='-'||infix[t+1]=='*'||infix[t+1]=='/'||infix[t+1]=='^'||infix[t+1]==')')
73  { printf("Invalid");exit(0);
74  }
75  }
76  }
77  infix_postfix(infix,postfix);
78  printf("the postfix exp is\n");
79  printf("%s \n",postfix);
80  getch();
81  return 0;
82  }
83
```

Compiler Resources Compile Log Debug Find Results

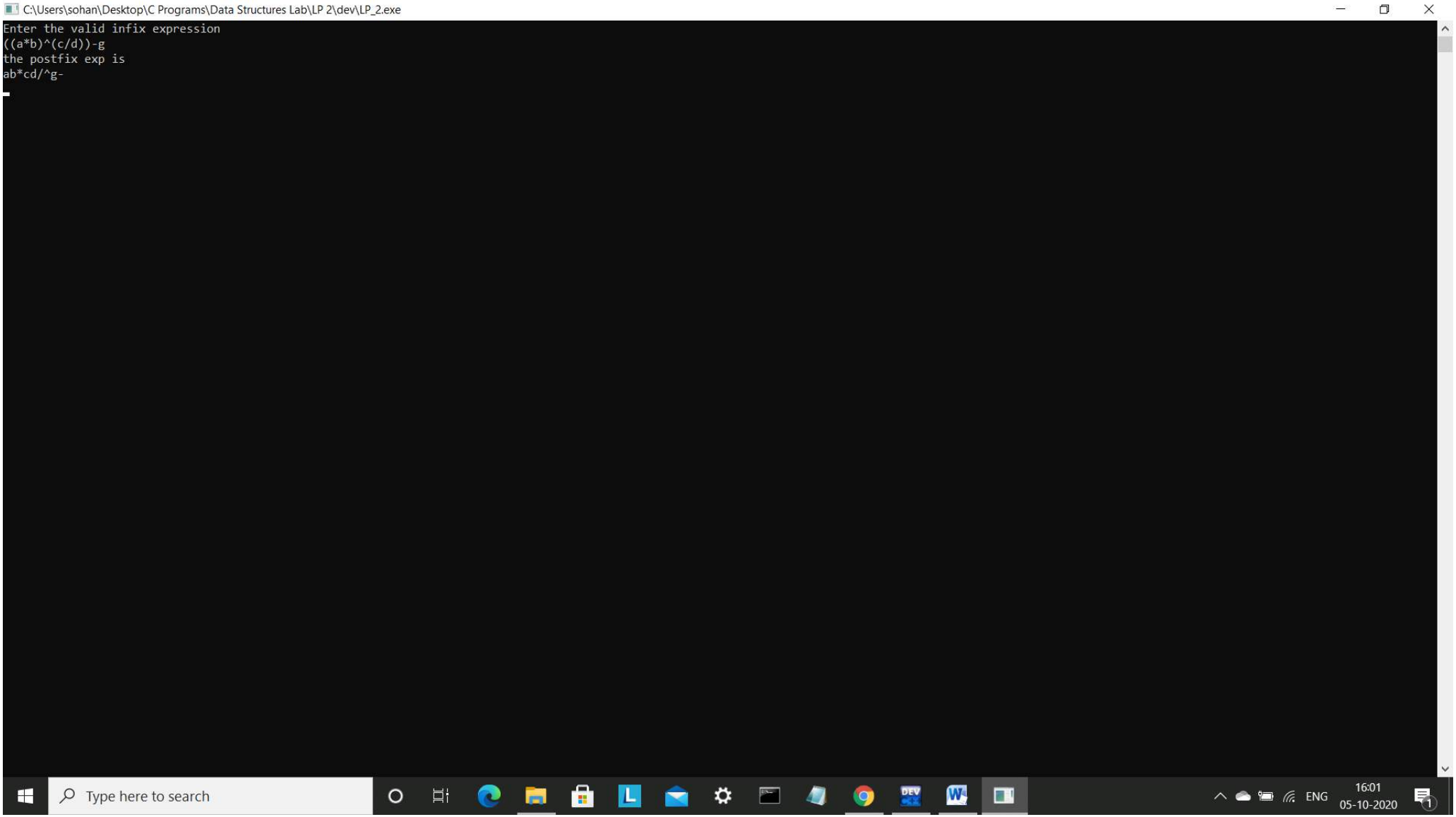
Line: 58 Col: 2 Sel: 0 Lines: 83 Length: 1356 Insert Done parsing in 0.015 seconds

Type here to search

ENG 15:53 05-10-2020

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\LP 2\dev\LP_2.exe

```
Enter the valid infix expression
a++b*c(/d)
Invalid
--------------------------------
Process exited after 36.19 seconds with return value 0
Press any key to continue . . .
```


C:\Users\sohan\Desktop\C Programs\Data Structures Lab\LP 2\dev\LP_2.exe
Enter the valid infix expression
((a*b)^(c/d))-g
the postfix exp is
ab*cd/^g-
Type here to search
ENG
16:01
05-10-2020

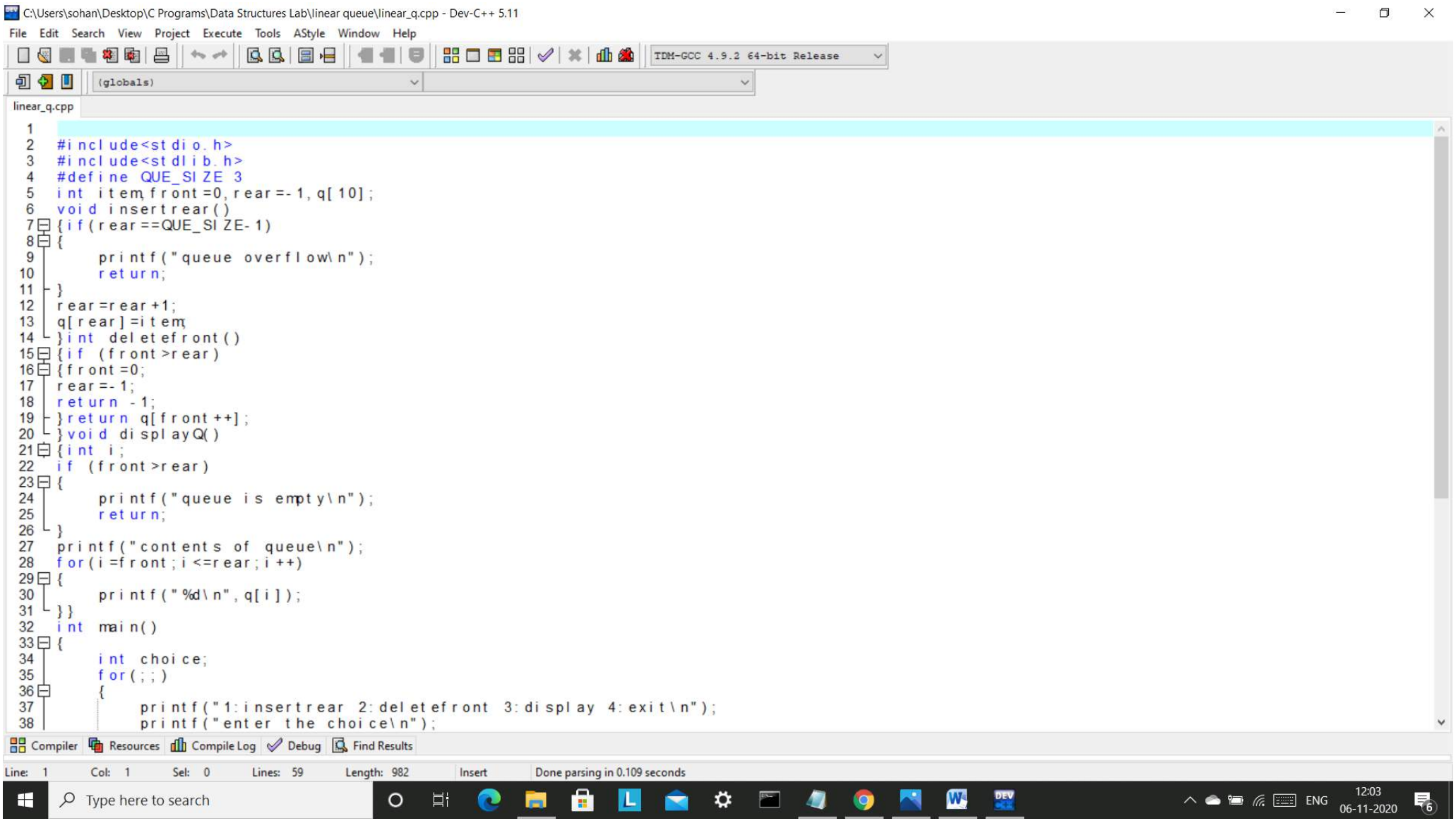
Lab Program 3:-

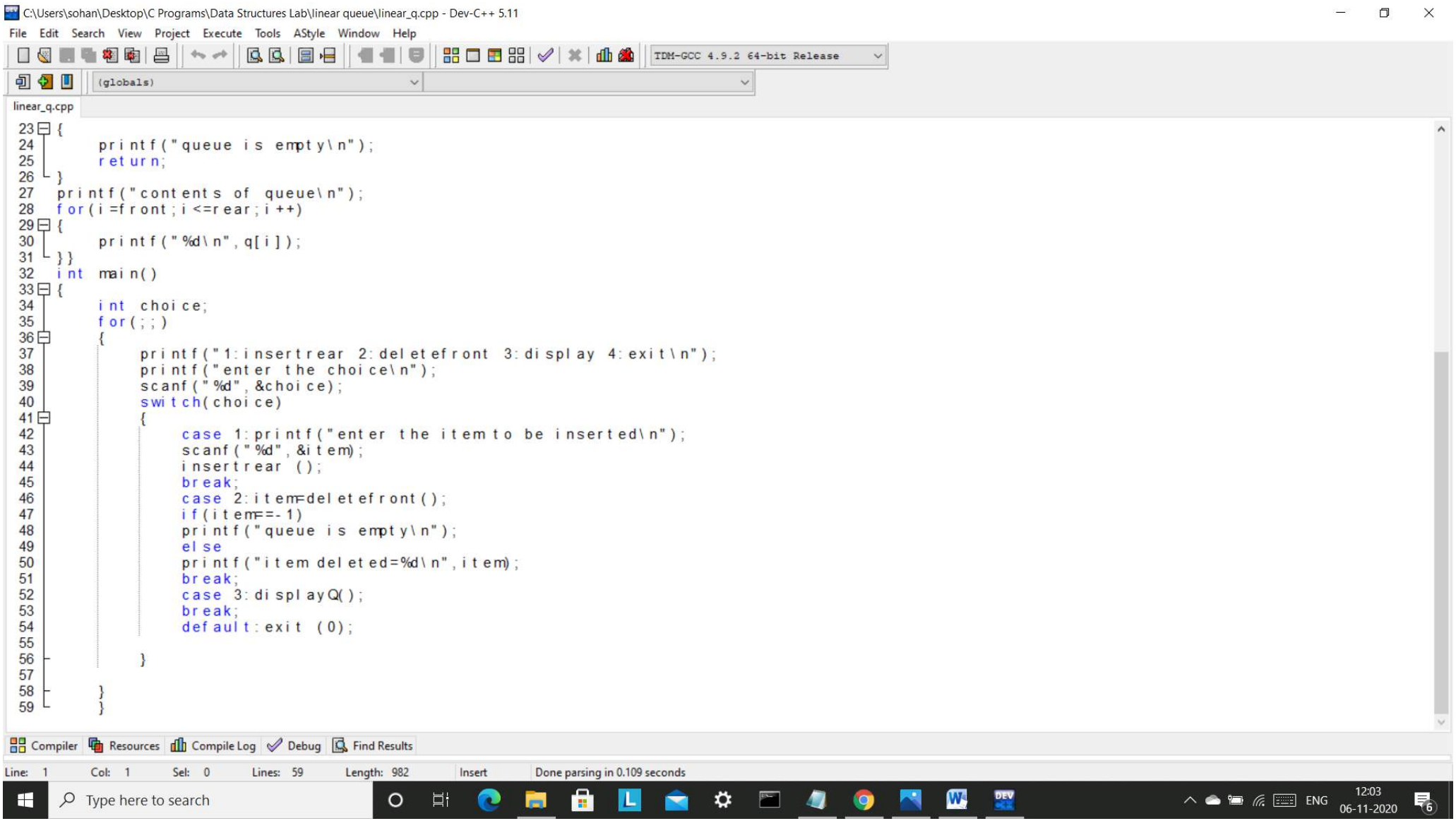
3) Write a program to simulate the working of a queue of integers using an array. Provide the following operations.
a) ~~Push~~ ~~b) Pop~~ a) Insert b) Delete c) Display.
The program should print appropriate messages for queue empty & queue overflow conditions.

A)

```
#include<stdio.h>
#include<stdlib.h>
#define QUE_SIZE 3
int item, front = 0, rear = -1, q[10];
void insertrear(){
if(rear == QUE_SIZE - 1){
printf("queue overflow \n");
return;
}
rear = rear + 1;
q[rear] = item;
}
int deletefront(){
If (front > rear){
front = 0;
rear = -1;
return -1;
}
return q[front++];
}
void displayQ(){
int i;
if(front > rear){
printf("queue is emptly \n");
return;
}
printf("contents of queue \n");
```

```
for(i=front; i<=rear; i++){
printf("%d \n", q[i]);
}}
int main(){
int choice;
for(;;){
printf("1: insert rear \n 2: delete front \n 3: display \4: exit \n");
printf(" enter your choice \n");
scanf("%d", &choice);
switch(choice){
case 1: printf("enter item to be ~~ing~~ inserted \n");
scanf("%d", &item);
insertrear();
break;
case 2: item=deletedfront();
if(item==-1)
printf("queue is empty \n");
else
printf("item deleted = %d \n", item);
break;
case 3: displayQ();
break;
default: exit(0);
}}}
```


C:\Users\sohan\Desktop\C Programs\Data Structures Lab\linear queue\linear_q.cpp - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
linear_q.cpp
#include<stdio.h>
#include<stdlib.h>
#define QUE_SIZE 3
int item,front=0,rear=-1,q[10];
void insertrear()
{if(rear==QUE_SIZE-1)
{
printf("queue overflow\n");
return;
}
rear=rear+1;
q[rear]=item;
}int deletefront()
{if (front>rear)
{front=0;
rear=-1;
return -1;
}return q[front++];
}void displayQ()
{int i;
if (front>rear)
{
printf("queue is empty\n");
return;
}
printf("contents of queue\n");
for(i=front;i<=rear;i++)
{
printf("%d\n",q[i]);
}}
int main()
{
int choice;
for(;;)
{
printf("1:insertrear 2:deletefront 3:display 4:exit\n");
printf("enter the choice\n");
Compiler Resources Compile Log Debug Find Results
Line: 1 Col: 1 Sel: 0 Lines: 59 Length: 982 Insert Done parsing in 0.109 seconds
Type here to search
ENG
12:03
06-11-2020


C:\Users\sohan\Desktop\C Programs\Data Structures Lab\linear queue\linear_q.cpp - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
linear_q.cpp
{
printf("queue is empty\n");
return;
}
printf("contents of queue\n");
for(i=front;i<=rear;i++)
{
printf("%d\n",q[i]);
}}
int main()
{
int choice;
for(;;)
{
printf("1:insertrear 2:deletefront 3:display 4:exit\n");
printf("enter the choice\n");
scanf("%d",&choice);
switch(choice)
{
case 1:printf("enter the item to be inserted\n");
scanf("%d",&item);
insertrear ();
break;
case 2:item=deletefront();
if(item==-1)
printf("queue is empty\n");
else
printf("item deleted=%d\n",item);
break;
case 3:displayQ();
break;
default:exit (0);
}
}
}
Compiler Resources Compile Log Debug Find Results
Line: 1 Col: 1 Sel: 0 Lines: 59 Length: 982 Insert Done parsing in 0.109 seconds
Type here to search
ENG
12:03
06-11-2020

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\linear queue\linear_q.exe

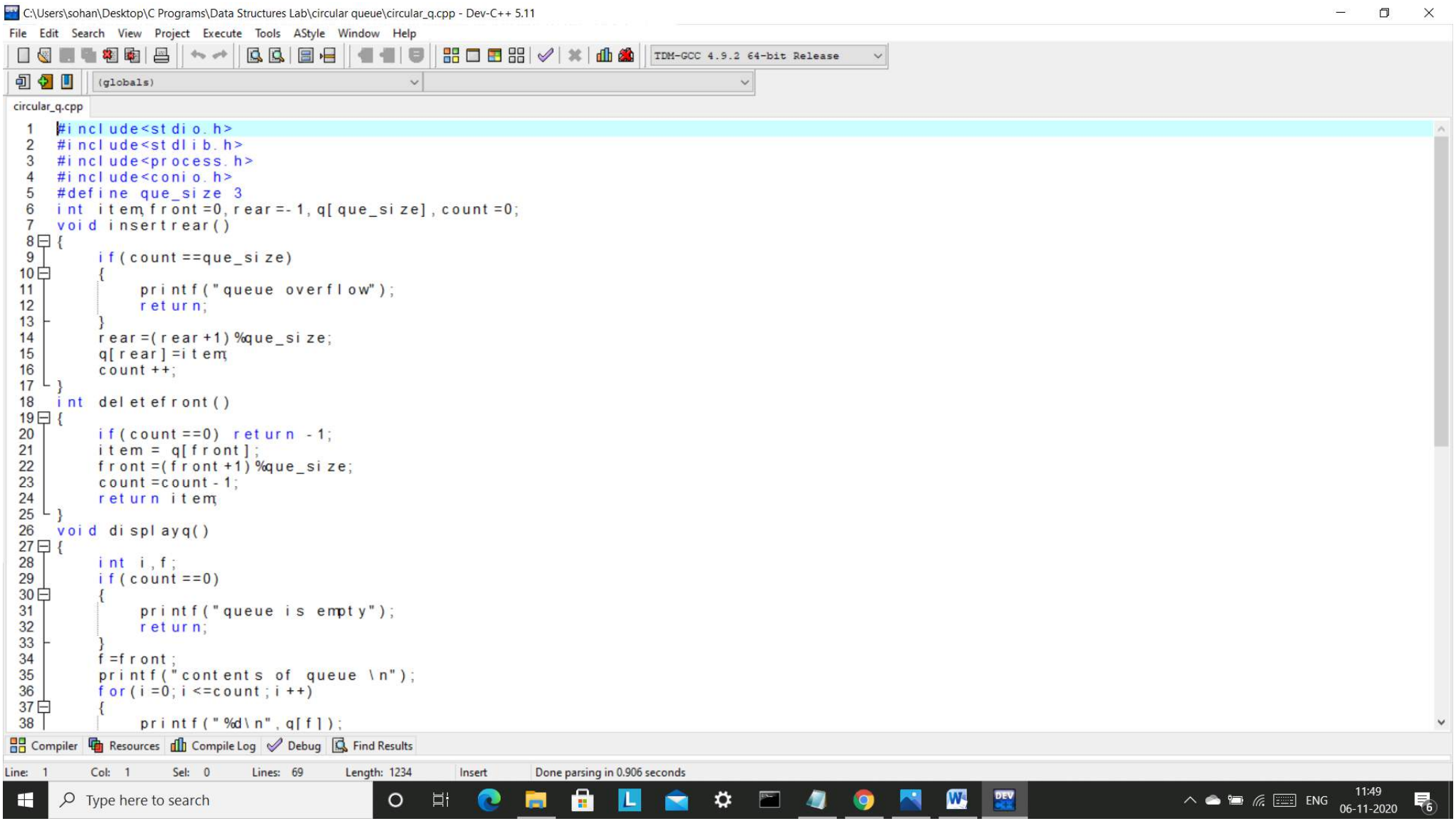
```
1:insertrear 2:deletefront 3:display 4:exit
enter the choice
1
enter the item to be inserted
12
1:insertrear 2:deletefront 3:display 4:exit
enter the choice
1
enter the item to be inserted
32
1:insertrear 2:deletefront 3:display 4:exit
enter the choice
1
enter the item to be inserted
55
1:insertrear 2:deletefront 3:display 4:exit
enter the choice
1
enter the item to be inserted
3
queue overflow
1:insertrear 2:deletefront 3:display 4:exit
enter the choice
2
item deleted=12
1:insertrear 2:deletefront 3:display 4:exit
enter the choice
3
contents of queue
32
55
1:insertrear 2:deletefront 3:display 4:exit
enter the choice
4

--------------------------------
Process exited after 29.8 seconds with return value 0
Press any key to continue . . .
```

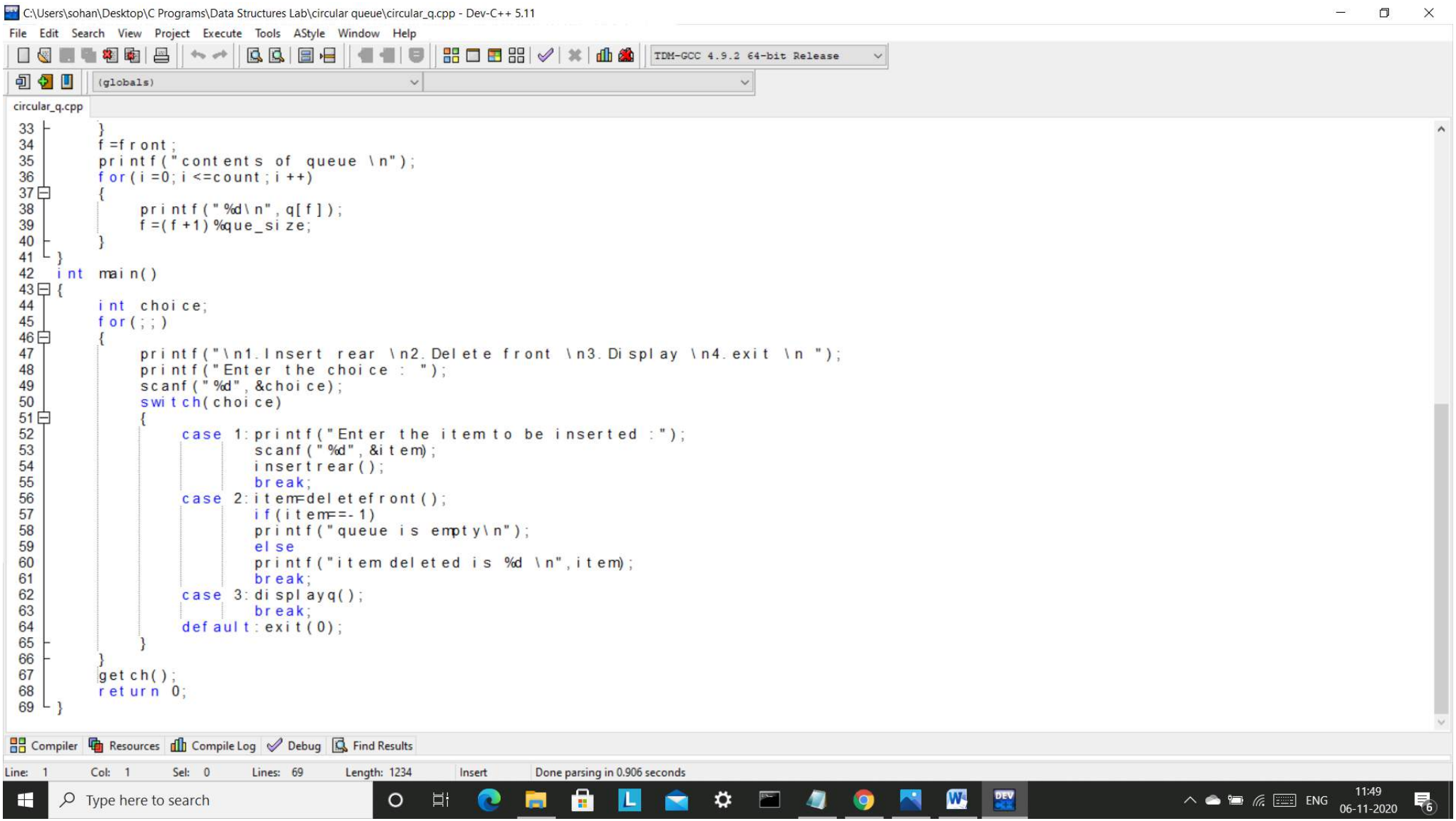
Lab Program 4:

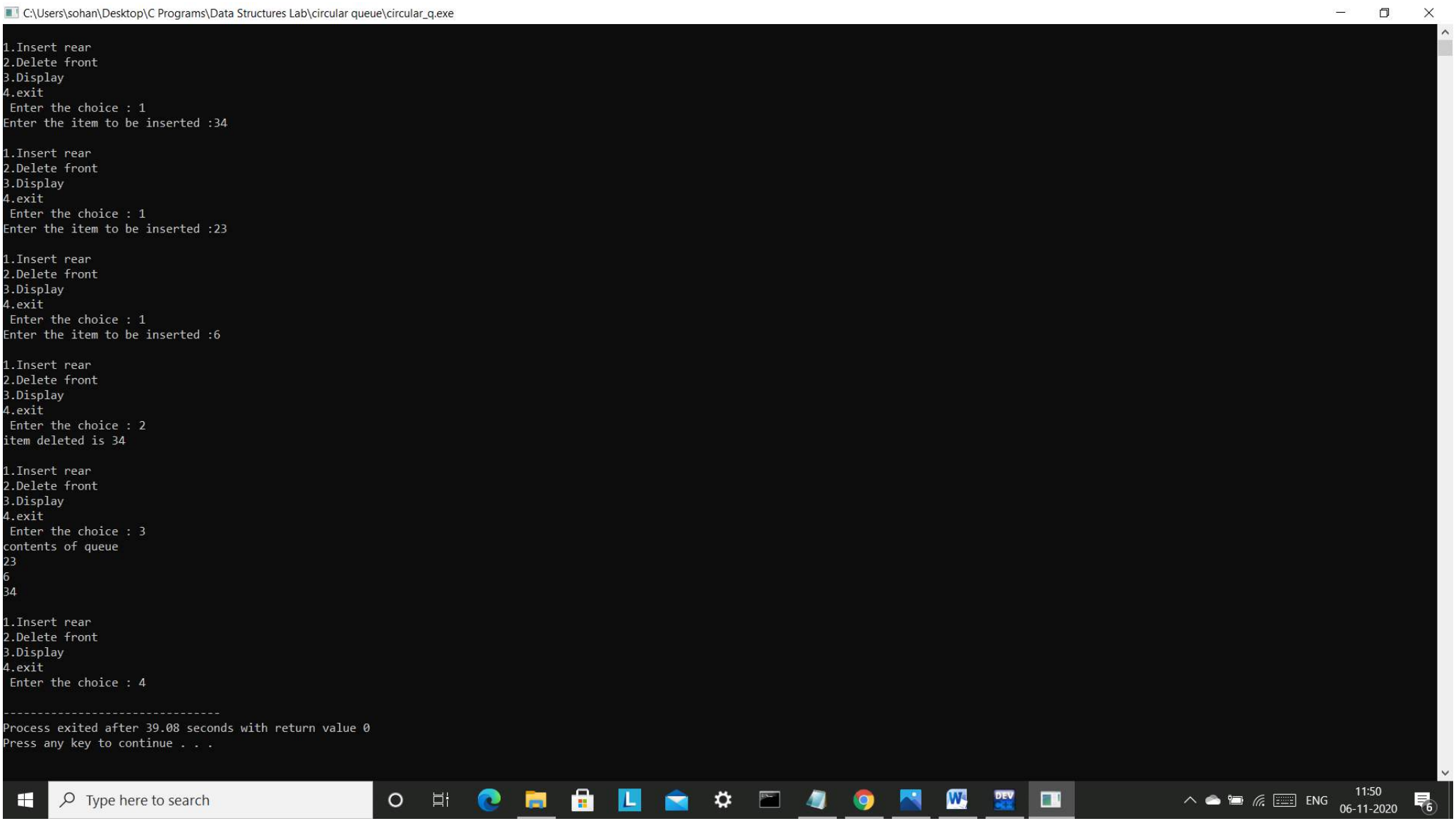
Q) Write a program to simulate the working of a circular queue of integers using an array. Provide the following operations.
a) Insert & Delete & Display.
The program should print appropriate messages for queue empty and queue overflow conditions.

A)
```
#include<stdio.h>
#include<stdlib.h>
#include<process.h>
#include<conio.h>
#define que_size 3
int item, front=0, rear=-1, q[que_size], count=0;
void insertrear(){
if(count==que_size){
printf("queue overflow");
return;
}
rear=(rear+1)%queue_size;
q[rear]=item;
count++;
}
int deletefront(){
if(count==0) return -1;
item=q[front];
front=(front+1)%que_size;
count=count-1;
return item;
}
void displayq(){
int i,f;
if(count==0){
printf("queue is empty");
return;
}
```

```
f = front;
printf(" contents of queue \n");
for(i=0; i<=count; i++){
printf("%d \n", q[f]);
f = (f+1) % que_size;
}}

int main(){
int choice;
for(;;){
printf("\n 1. insert rear \n 2. delete front \n 3. display \n
        4. exit \n");
printf(" Enter the choice : ");
scanf("%d", &choice);
switch(choice){
case 1: printf(" Enter the item to be inserted : ");
        scanf("%d ", &item);
        insertrear();
        break;
case 2: item = deletefront();
        if(item == -1)
        printf("queue is empty \n");
        else
        printf(" item deleted is %d \n", item);
        break;
case 3: displayq();
        break;
default: exit(0);
}}

getch();
return 0;
}
```


C:\Users\sohan\Desktop\C Programs\Data Structures Lab\circular queue\circular_q.cpp - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
circular_q.cpp
#include<stdio.h>
#include<stdlib.h>
#include<process.h>
#include<conio.h>
#define que_size 3
int item,front=0,rear=-1,q[que_size],count=0;
void insertrear()
{
if(count==que_size)
{
printf("queue overflow");
return;
}
rear=(rear+1)%que_size;
q[rear]=item;
count++;
}
int deletefront()
{
if(count==0) return -1;
item = q[front];
front=(front+1)%que_size;
count=count-1;
return item;
}
void displayq()
{
int i,f;
if(count==0)
{
printf("queue is empty");
return;
}
f=front;
printf("contents of queue \n");
for(i=0;i<=count;i++)
{
printf("%d\n",q[f]);
Compiler Resources Compile Log Debug Find Results
Line: 1 Col: 1 Sel: 0 Lines: 69 Length: 1234 Insert Done parsing in 0.906 seconds
Type here to search
ENG
11:49
06-11-2020


C:\Users\sohan\Desktop\C Programs\Data Structures Lab\circular queue\circular_q.cpp - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
circular_q.cpp
}
f=front;
printf("contents of queue \n");
for(i=0;i<=count;i++)
{
printf("%d\n",q[f]);
f=(f+1)%que_size;
}
}
int main()
{
int choice;
for(;;)
{
printf("\n1.Insert rear \n2.Delete front \n3.Display \n4.exit \n ");
printf("Enter the choice : ");
scanf("%d",&choice);
switch(choice)
{
case 1:printf("Enter the item to be inserted :");
scanf("%d",&item);
insertrear();
break;
case 2:item=deletefront();
if(item==-1)
printf("queue is empty\n");
else
printf("item deleted is %d \n",item);
break;
case 3:displayq();
break;
default:exit(0);
}
}
getch();
return 0;
}
Compiler Resources Compile Log Debug Find Results
Line: 1 Col: 1 Sel: 0 Lines: 69 Length: 1234 Insert Done parsing in 0.906 seconds
Type here to search
ENG
11:49
06-11-2020


C:\Users\sohan\Desktop\C Programs\Data Structures Lab\circular queue\circular_q.exe
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 1
Enter the item to be inserted :34
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 1
Enter the item to be inserted :23
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 1
Enter the item to be inserted :6
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 2
item deleted is 34
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 3
contents of queue
23
6
34
1.Insert rear
2.Delete front
3.Display
4.exit
Enter the choice : 4
Process exited after 39.08 seconds with return value 0
Press any key to continue . . .
Type here to search
ENG
11:50
06-11-2020

Lab program 5:

5) Write a program to implement Singly Linked List with following operations : a) Create an linked list, b) Insertion of a node at first position, at any position and at end of list. c) Display the contents of linked list.

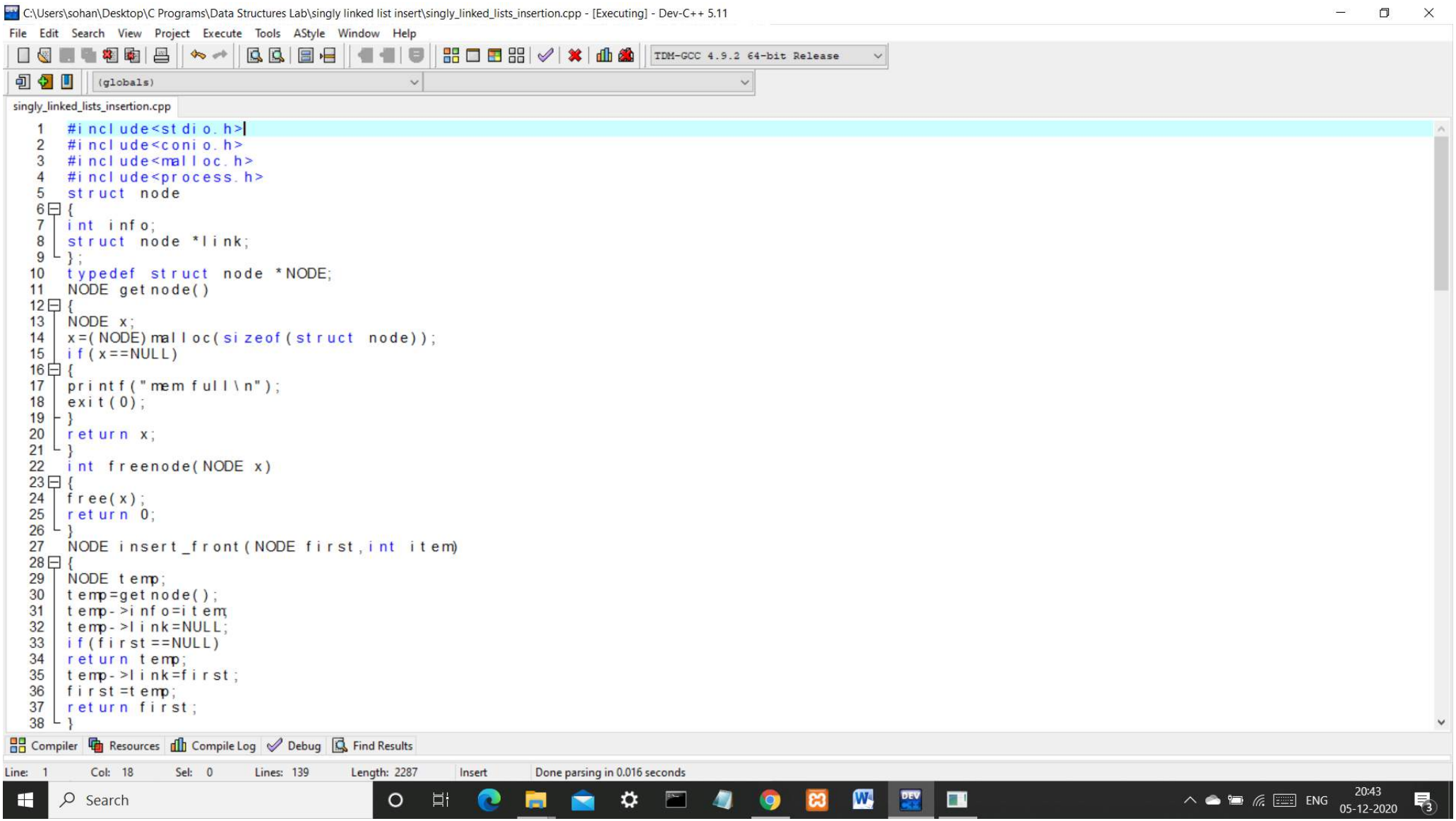
A)

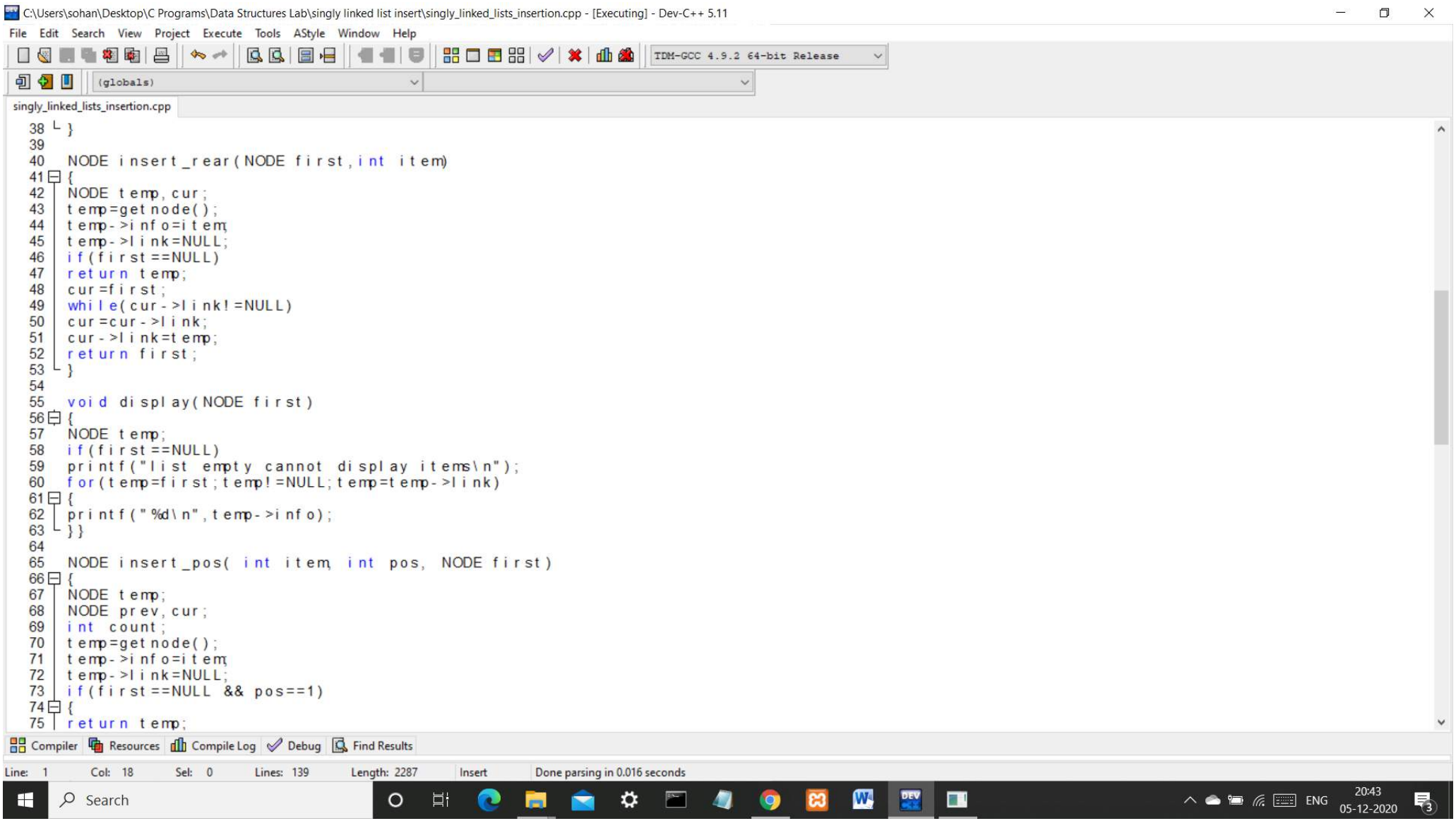
```
#include <stdio.h>
#include <conio.h>
#include <malloc.h>
#include <process.h>
struct node
{ int info;
  struct node *link;
};
typedef struct node *NODE;
NODE getnode()
{
NODE x;
x=(NODE)malloc(sizeof(struct node));
if(x==NULL){
printf("mem full \n");
exit(0);
}
return x;
}
int freenode(NODE x){
free(x);
return 0;
}
NODE insert_front(NODE first, int item){
NODE temp;
temp = getnode();
temp->info = item;
temp->link = NULL;
```

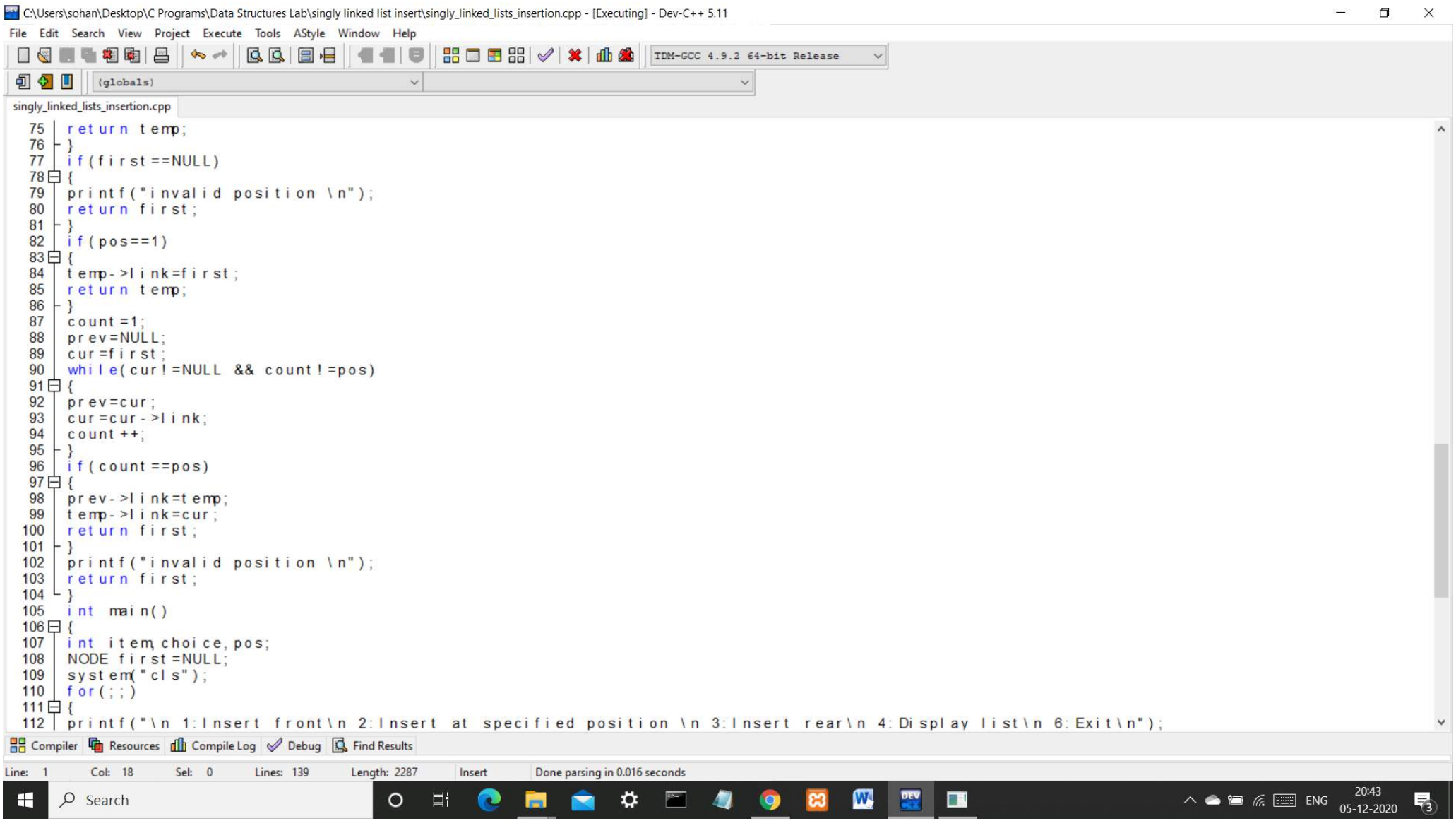
```
if (first == NULL)
return temp;
temp -> link = first;
first = temp;
return temp;
return first;
}
NODE insert_rear(NODE first, int item){
NODE temp, cur;
temp = getnode();
temp -> info = item;
temp -> link = NULL;
if (first == NULL)
return temp;
cur = first;
while(cur -> link != NULL)
cur = cur -> link;
cur -> link = temp;
return first;
}
void display(NODE first){
NODE temp;
if (first == NULL)
printf("list empty cannot display items \n");
for(temp = first; temp != NULL; temp = temp -> link){
printf("%d \n", temp -> info);
}}
NODE insert_pos(int item, int pos, NODE first){
NODE temp;
NODE prev, cur;
int count;
temp = getnode();
temp -> info = item;
temp -> link = NULL;
```

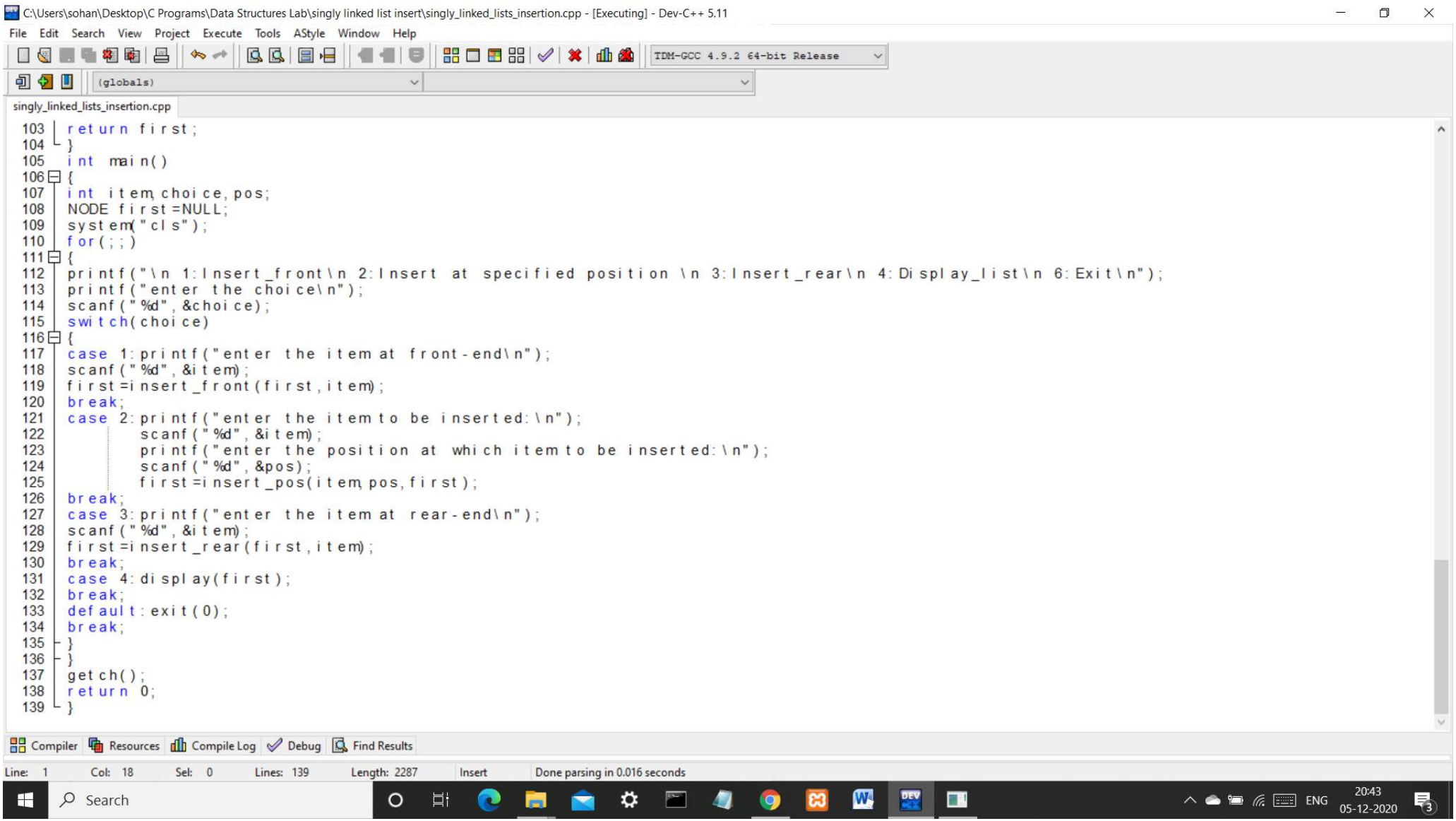
```
if(first==NULL && pos==1){
return temp;
}
if(first==NULL){
printf("invalid position\n");
return first;
}
if(pos==1){
temp->link=first;
return temp;
}
count=1;
prev=NULL;
cur=first;
while(cur!=NULL && count!=pos){
prev=cur;
cur=cur->link;
count++;
}
if(count==pos){
prev->link=temp;
temp->link=cur;
return first;
}
printf("invalid position\n");
return first;
}
int main(){
int item, choice, pos;
NODE first=NULL;
system("cls");
for(;;){
printf("\n1: Insert front \n2: Insert at specified
position \n3: Insert rear \n4: Display list \n5: Exit
```

```
\n");
return first;
}
printf("enter the choice \n");
scanf("%d", &choice);
switch(choice){
case 1: printf("enter the item at front-end: \n");
        scanf("%d", &item);
        first = insert_front(first, item);
        break;
case 2: printf("enter the item to be inserted: \n");
        scanf("%d", &item);
        printf("enter the position at which item to be inserted: \n");
        scanf("%d", &pos);
        first = insert_pos(item, pos, first);
        break;
case 3: printf("enter the item at rear-end \n");
        scanf("%d", &item);
        first = insert_rear(first, item);
        break;
case 4: display(first);
break;
default: exit(0);
break;
}}
getch();
return 0;
}
```


C:\Users\sohan\Desktop\C Programs\Data Structures Lab\singly linked list insert\singly_linked_lists_insertion.cpp - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
singly_linked_lists_insertion.cpp
#include<stdio.h>
#include<conio.h>
#include<malloc.h>
#include<process.h>
struct node
{
int info;
struct node *link;
};
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;
}
int freenode(NODE x)
{
free(x);
return 0;
}
NODE insert_front(NODE first,int item)
{
NODE temp;
temp=getnode();
temp->info=item;
temp->link=NULL;
if(first==NULL)
return temp;
temp->link=first;
first=temp;
return first;
}
Compiler Resources Compile Log Debug Find Results
Line: 1 Col: 18 Sel: 0 Lines: 139 Length: 2287 Insert Done parsing in 0.016 seconds
Search
ENG
20:43
05-12-2020


C:\Users\sohan\Desktop\C Programs\Data Structures Lab\singly linked list insert\singly_linked_lists_insertion.cpp - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
singly_linked_lists_insertion.cpp
38 }
39
40 NODE insert_rear(NODE first,int item)
41 {
42 NODE temp,cur;
43 temp=getnode();
44 temp->info=item;
45 temp->link=NULL;
46 if(first==NULL)
47 return temp;
48 cur=first;
49 while(cur->link!=NULL)
50 cur=cur->link;
51 cur->link=temp;
52 return first;
53 }
54
55 void display(NODE first)
56 {
57 NODE temp;
58 if(first==NULL)
59 printf("list empty cannot display items\n");
60 for(temp=first;temp!=NULL;temp=temp->link)
61 {
62 printf("%d\n",temp->info);
63 }}
64
65 NODE insert_pos(int item, int pos, NODE first)
66 {
67 NODE temp;
68 NODE prev,cur;
69 int count;
70 temp=getnode();
71 temp->info=item;
72 temp->link=NULL;
73 if(first==NULL && pos==1)
74 {
75 return temp;
Compiler Resources Compile Log Debug Find Results
Line: 1 Col: 18 Sel: 0 Lines: 139 Length: 2287 Insert Done parsing in 0.016 seconds
Search
ENG
20:43
05-12-2020


C:\Users\sohan\Desktop\C Programs\Data Structures Lab\singly linked list insert\singly_linked_lists_insertion.cpp - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
singly_linked_lists_insertion.cpp
return temp;
}
if(first==NULL)
{
printf("invalid position \n");
return first;
}
if(pos==1)
{
temp->link=first;
return temp;
}
count=1;
prev=NULL;
cur=first;
while(cur!=NULL && count!=pos)
{
prev=cur;
cur=cur->link;
count++;
}
if(count==pos)
{
prev->link=temp;
temp->link=cur;
return first;
}
printf("invalid position \n");
return first;
}
int main()
{
int item,choice,pos;
NODE first=NULL;
system("cls");
for(;;)
{
printf("\n 1:Insert front\n 2:Insert at specified position \n 3:Insert rear\n 4:Display list\n 6:Exit\n");
Compiler Resources Compile Log Debug Find Results
Line: 1 Col: 18 Sel: 0 Lines: 139 Length: 2287 Insert Done parsing in 0.016 seconds
Search
ENG
20:43
05-12-2020


C:\Users\sohan\Desktop\C Programs\Data Structures Lab\singly linked list insert\singly_linked_lists_insertion.cpp - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
singly_linked_lists_insertion.cpp
return first;
}
int main()
{
int item,choice,pos;
NODE first=NULL;
system("cls");
for(;;)
{
printf("\n 1:Insert_front\n 2:Insert at specified position \n 3:Insert_rear\n 4:Display_list\n 6: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);
first=insert_front(first,item);
break;
case 2:printf("enter the item to be inserted:\n");
scanf("%d",&item);
printf("enter the position at which item to be inserted:\n");
scanf("%d",&pos);
first=insert_pos(item,pos,first);
break;
case 3:printf("enter the item at rear-end\n");
scanf("%d",&item);
first=insert_rear(first,item);
break;
case 4:display(first);
break;
default:exit(0);
break;
}
}
getch();
return 0;
}
Compiler Resources Compile Log Debug Find Results
Line: 1 Col: 18 Sel: 0 Lines: 139 Length: 2287 Insert Done parsing in 0.016 seconds
Search
ENG
20:43
05-12-2020

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\singly linked list insert\singly_linked_lists_insertion.exe

```
 1:Insert_front
 2:Insert at specified position
 3:Insert_rear
 4:Display_list
 6:Exit
enter the choice
1
enter the item at front-end
54

 1:Insert_front
 2:Insert at specified position
 3:Insert_rear
 4:Display_list
 6:Exit
enter the choice
3
enter the item at rear-end
78

 1:Insert_front
 2:Insert at specified position
 3:Insert_rear
 4:Display_list
 6:Exit
enter the choice
2
enter the item to be inserted:
65
enter the position at which item to be inserted:
3

 1:Insert_front
 2:Insert at specified position
 3:Insert_rear
 4:Display_list
 6:Exit
enter the choice
2
enter the item to be inserted:
1
enter the position at which item to be inserted:
2

 1:Insert_front
 2:Insert at specified position
 3:Insert_rear
 4:Display_list
 6:Exit
```

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\singly linked list insert\singly_linked_lists_insertion.exe

```
78

 1:Insert_front
 2:Insert at specified position
 3:Insert_rear
 4:Display_list
 6:Exit
enter the choice
2
enter the item to be inserted:
65
enter the position at which item to be inserted:
3

 1:Insert_front
 2:Insert at specified position
 3:Insert_rear
 4:Display_list
 6:Exit
enter the choice
2
enter the item to be inserted:
1
enter the position at which item to be inserted:
2

 1:Insert_front
 2:Insert at specified position
 3:Insert_rear
 4:Display_list
 6:Exit
enter the choice
4
54
1
78
65

 1:Insert_front
 2:Insert at specified position
 3:Insert_rear
 4:Display_list
 6:Exit
enter the choice
```

Lab Program 6:-

6) Write a program implement Singly linked list with following operation:

a) Create a linked list. b) Deletion of first element, specified element and last element in the list.

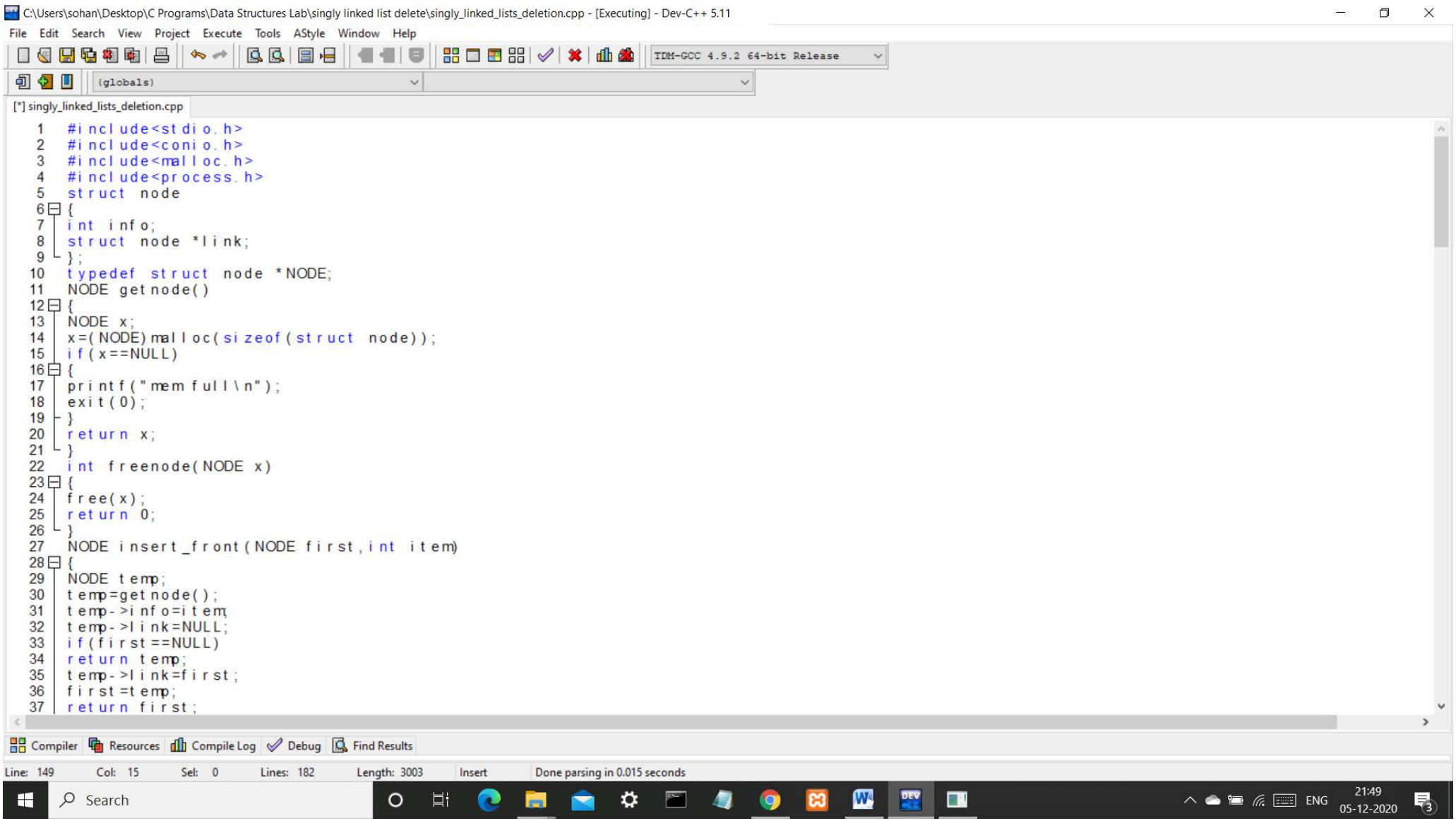
c) Display the contents of the linked list.

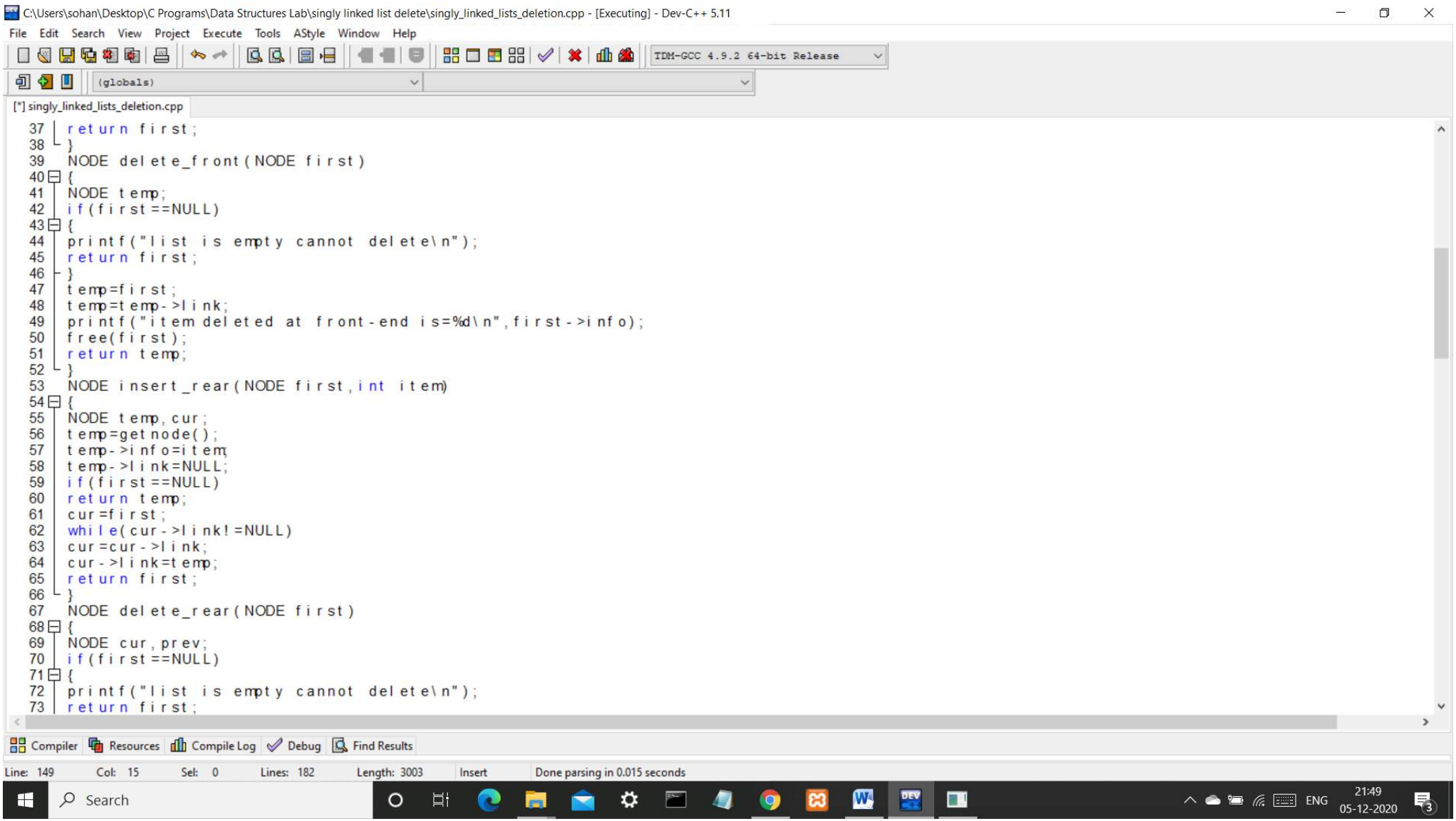
```
#include<stdio.h>
#include<conio.h>
#include<malloc.h>
#include<process.h>
struct node
{
int info;
struct node *link;
};
typedef struct node *NODE;
NODE getnode(){
x=(NODE) malloc(sizeof(struct node));
if(x==NULL){
printf("mem full\n");
exit(0);
}return x;
}int freenode(NODE x){
free(x);
return 0;
}
NODE insert_front(NODE first, int item){
NODE temp;
temp=getnode();
temp->info=item;
temp->link=NULL;
if(first==NULL)
return temp;
```

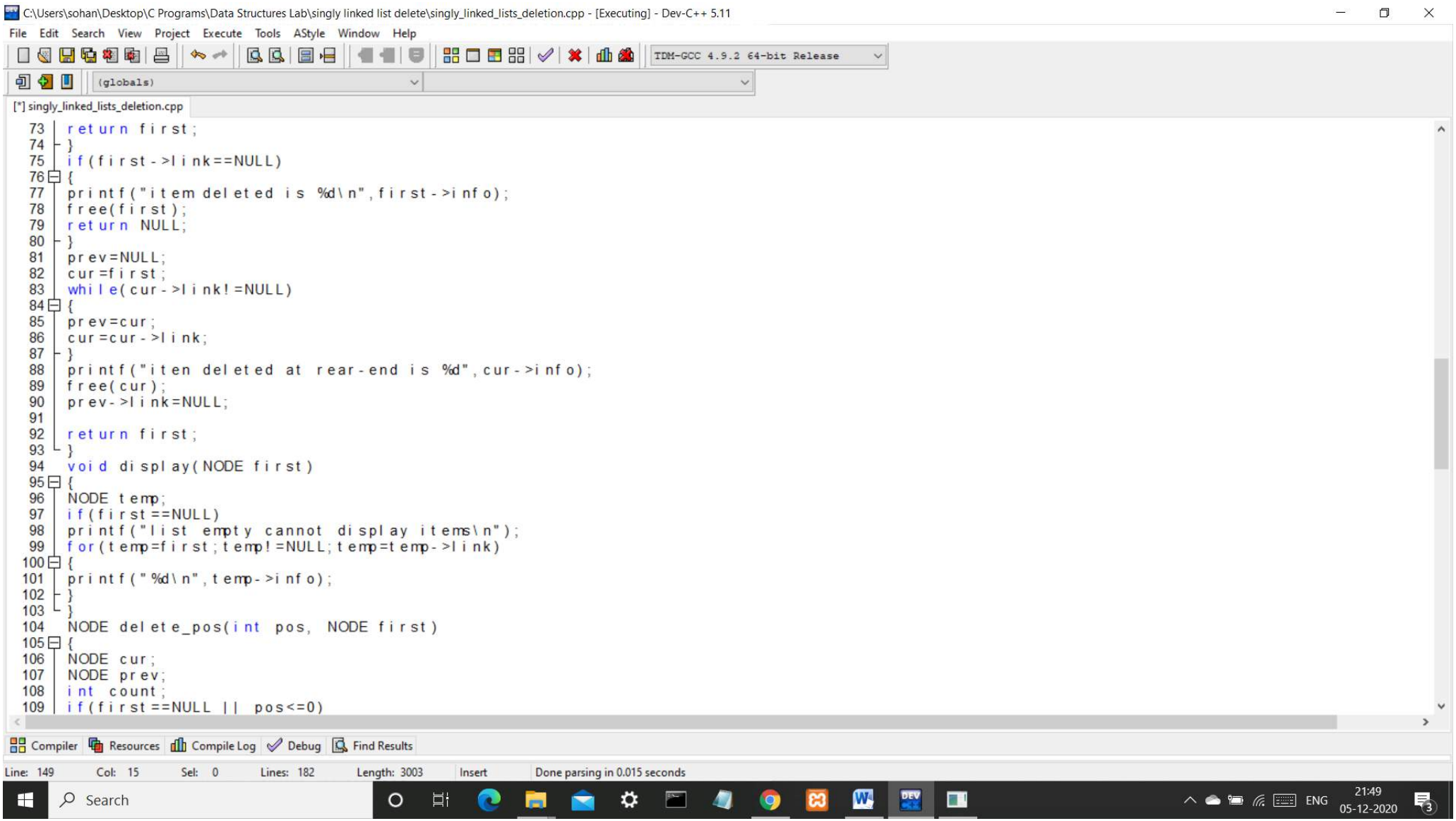
```
temp->link = first;
first = temp;
return first;
}

NODE delete_front(NODE first){
NODE temp;
if(first == NULL){
printf("list is empty cannot delete \n");
return first;
}

NODE delete_rear(NODE first){
NODE cur, prev;
if(first == NULL){
printf("list is empty cannot delete \n");
return first;
if(first->link == NULL){
printf("item deleted is %d \n", first->info);
free(first);
return NULL;
}

prev = NULL;
cur = first;
while(cur->link != NULL){
prev = cur;
cur = cur->link;
}

printf("item deleted at rear-end is %d", cur->info);
free(cur);
prev->link = NULL;
return first;
}

void display(NODE first){
NODE temp;
if(first == NULL)
```

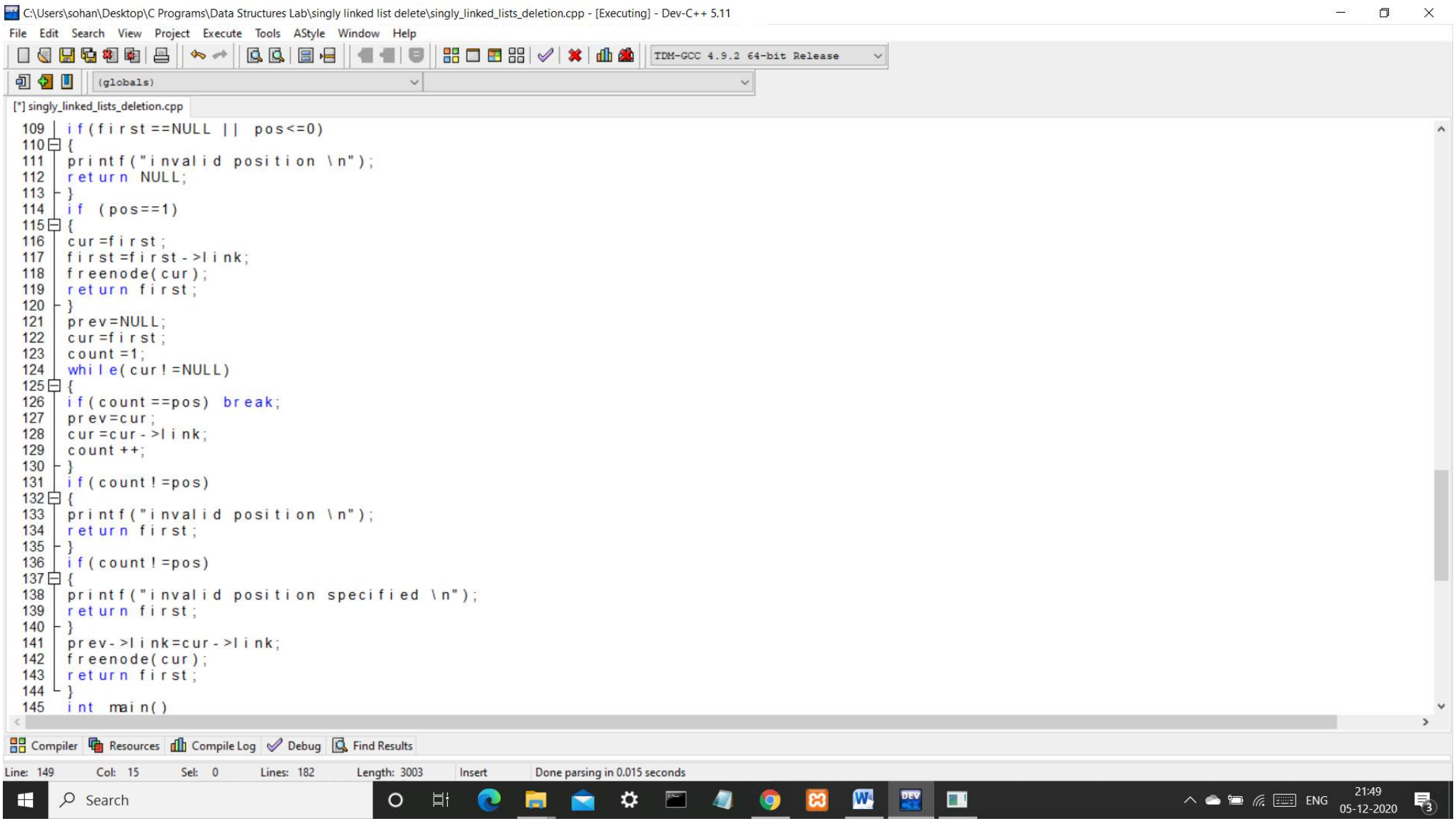
```
printf ("list empty cannot display items \n");
for (temp = first; temp != NULL; temp = temp -> link) {
printf ("%d \n", temp -> info);
}}

NODE delete_pos(int pos, NODE first) {
NODE cur;
NODE prev;
int count;
if (first == NULL || pos <= 0) {
printf ("invalid position \n");
return NULL;
}
if (pos == 1) {
cur = first;
first = first -> link;
freenode (cur);
return first;
}
prev = NULL;
cur = first;
count = 1;
while (cur != NULL) {
if (count == pos) break;
prev = cur;
cur = cur -> link;
count++;
}
if (count != pos) {
printf ("invalid position \n");
return first;
}
if (count != pos) {
printf ("invalid position specified \n");
return first;
}
```

```
prev->link = cur->link;
freenode(cur);
return first;
}
int main(){
int item, choice, pos;
NODE first = NULL;
system("cls");
for(;;){
printf("\n 1: Insert_front \n 2: Delete_front \n 3: Insert_rear \n
 4: Delete_rear \n 5. Delete at specified position. \n6: Display_list
\n 7: 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);
        first=insert_front(first, item);
        break;
case 2: first=delete_front(first);
        break;
case 3: printf("enter the item at rear-end \n");
      scanf("%d", &item);
      first = insert_rear(first, item);
      break;
case 4: first = delete_rear(first);
break;
case 5: printf("enter the position of the item to be deleted: \n")
        scanf("%d", &pos);
        first = delete_pos(pos, first);
        break;
case 6: display(first);
break;
default: exit(0); break; }} getch(); return 0; }
```


C:\Users\sohan\Desktop\C Programs\Data Structures Lab\singly linked list delete\singly_linked_lists_deletion.cpp - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
[*] singly_linked_lists_deletion.cpp
#include<stdio.h>
#include<conio.h>
#include<malloc.h>
#include<process.h>
struct node
{
int info;
struct node *link;
};
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;
}
int freenode(NODE x)
{
free(x);
return 0;
}
NODE insert_front(NODE first,int item)
{
NODE temp;
temp=getnode();
temp->info=item;
temp->link=NULL;
if(first==NULL)
return temp;
temp->link=first;
first=temp;
return first;
Compiler Resources Compile Log Debug Find Results
Line: 149 Col: 15 Sel: 0 Lines: 182 Length: 3003 Insert Done parsing in 0.015 seconds
Search
ENG
21:49
05-12-2020


C:\Users\sohan\Desktop\C Programs\Data Structures Lab\singly linked list delete\singly_linked_lists_deletion.cpp - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
[*] singly_linked_lists_deletion.cpp
return first;
}
NODE delete_front(NODE first)
{
NODE temp;
if(first==NULL)
{
printf("list is empty cannot delete\n");
return first;
}
temp=first;
temp=temp->link;
printf("item deleted at front-end is=%d\n",first->info);
free(first);
return temp;
}
NODE insert_rear(NODE first,int item)
{
NODE temp,cur;
temp=getnode();
temp->info=item;
temp->link=NULL;
if(first==NULL)
return temp;
cur=first;
while(cur->link!=NULL)
cur=cur->link;
cur->link=temp;
return first;
}
NODE delete_rear(NODE first)
{
NODE cur,prev;
if(first==NULL)
{
printf("list is empty cannot delete\n");
return first;
Compiler Resources Compile Log Debug Find Results
Line: 149 Col: 15 Sel: 0 Lines: 182 Length: 3003 Insert Done parsing in 0.015 seconds
Search
ENG
21:49
05-12-2020


C:\Users\sohan\Desktop\C Programs\Data Structures Lab\singly linked list delete\singly_linked_lists_deletion.cpp - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
[*] singly_linked_lists_deletion.cpp
73 return first;
74 }
75 if(first->link==NULL)
76 {
77 printf("item deleted is %d\n",first->info);
78 free(first);
79 return NULL;
80 }
81 prev=NULL;
82 cur=first;
83 while(cur->link!=NULL)
84 {
85 prev=cur;
86 cur=cur->link;
87 }
88 printf("iten deleted at rear-end is %d",cur->info);
89 free(cur);
90 prev->link=NULL;
91
92 return first;
93 }
94 void display(NODE first)
95 {
96 NODE temp;
97 if(first==NULL)
98 printf("list empty cannot display items\n");
99 for(temp=first;temp!=NULL;temp=temp->link)
100 {
101 printf("%d\n",temp->info);
102 }
103 }
104 NODE delete_pos(int pos, NODE first)
105 {
106 NODE cur;
107 NODE prev;
108 int count;
109 if(first==NULL || pos<=0)
Compiler Resources Compile Log Debug Find Results
Line: 149 Col: 15 Sel: 0 Lines: 182 Length: 3003 Insert Done parsing in 0.015 seconds
Search
ENG
21:49
05-12-2020


C:\Users\sohan\Desktop\C Programs\Data Structures Lab\singly linked list delete\singly_linked_lists_deletion.cpp - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
[*] singly_linked_lists_deletion.cpp
109 if(first==NULL || pos<=0)
110 {
111 printf("invalid position \n");
112 return NULL;
113 }
114 if (pos==1)
115 {
116 cur=first;
117 first=first->link;
118 freenode(cur);
119 return first;
120 }
121 prev=NULL;
122 cur=first;
123 count=1;
124 while(cur!=NULL)
125 {
126 if(count==pos) break;
127 prev=cur;
128 cur=cur->link;
129 count++;
130 }
131 if(count!=pos)
132 {
133 printf("invalid position \n");
134 return first;
135 }
136 if(count!=pos)
137 {
138 printf("invalid position specified \n");
139 return first;
140 }
141 prev->link=cur->link;
142 freenode(cur);
143 return first;
144 }
145 int main()
Compiler Resources Compile Log Debug Find Results
Line: 149 Col: 15 Sel: 0 Lines: 182 Length: 3003 Insert Done parsing in 0.015 seconds
Search
ENG
21:49
05-12-2020

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\singly linked list delete\singly_linked_lists_deletion.cpp - [Executing] - Dev-C++ 5.11

File Edit Search View Project Execute Tools AStyle Window Help

TDM-GCC 4.9.2 64-bit Release

(globals)

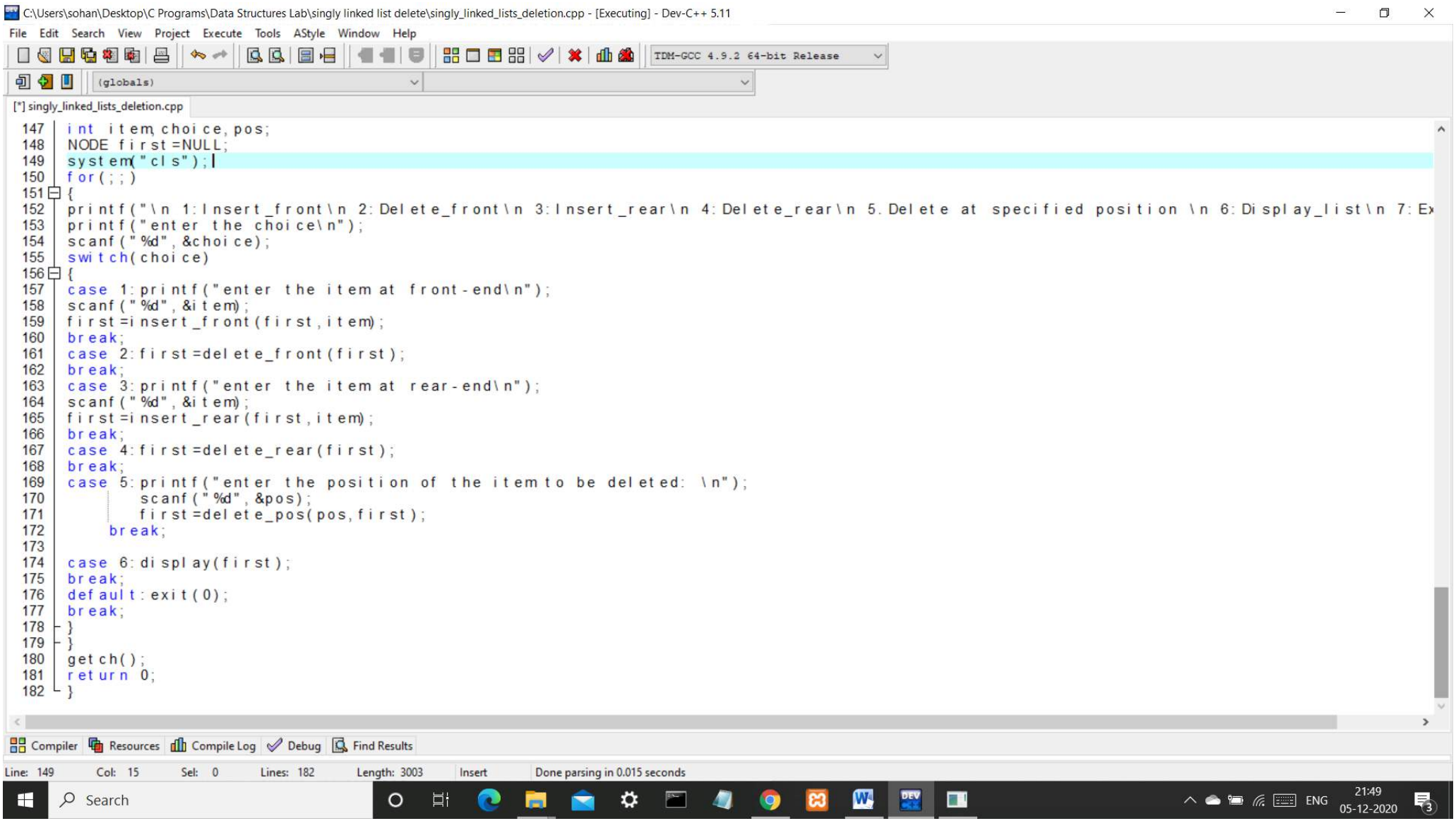
[*] singly_linked_lists_deletion.cpp

```
143 return first;
144 }
145 int main()
146 {
147 int item,choice,pos;
148 NODE first=NULL;
149 system("cls");
150 for(;;)
151 {
152 printf("\n 1:Insert_front\n 2:Delete_front\n 3:Insert_rear\n 4:Delete_rear\n 5.Delete at specified position \n 6:Display_list\n 7:Ex
153 printf("enter the choice\n");
154 scanf("%d",&choice);
155 switch(choice)
156 {
157 case 1:printf("enter the item at front-end\n");
158 scanf("%d",&item);
159 first=insert_front(first,item);
160 break;
161 case 2:first=delete_front(first);
162 break;
163 case 3:printf("enter the item at rear-end\n");
164 scanf("%d",&item);
165 first=insert_rear(first,item);
166 break;
167 case 4:first=delete_rear(first);
168 break;
169 case 5:printf("enter the position of the item to be deleted: \n");
170        scanf("%d",&pos);
171        first=delete_pos(pos,first);
172     break;
173
174 case 6:display(first);
175 break;
176 default:exit(0);
177 break;
178 }
179 }
```

Compiler Resources Compile Log Debug Find Results

Line: 149 Col: 15 Sel: 0 Lines: 182 Length: 3003 Insert Done parsing in 0.015 seconds

Search ENG 21:49 05-12-2020

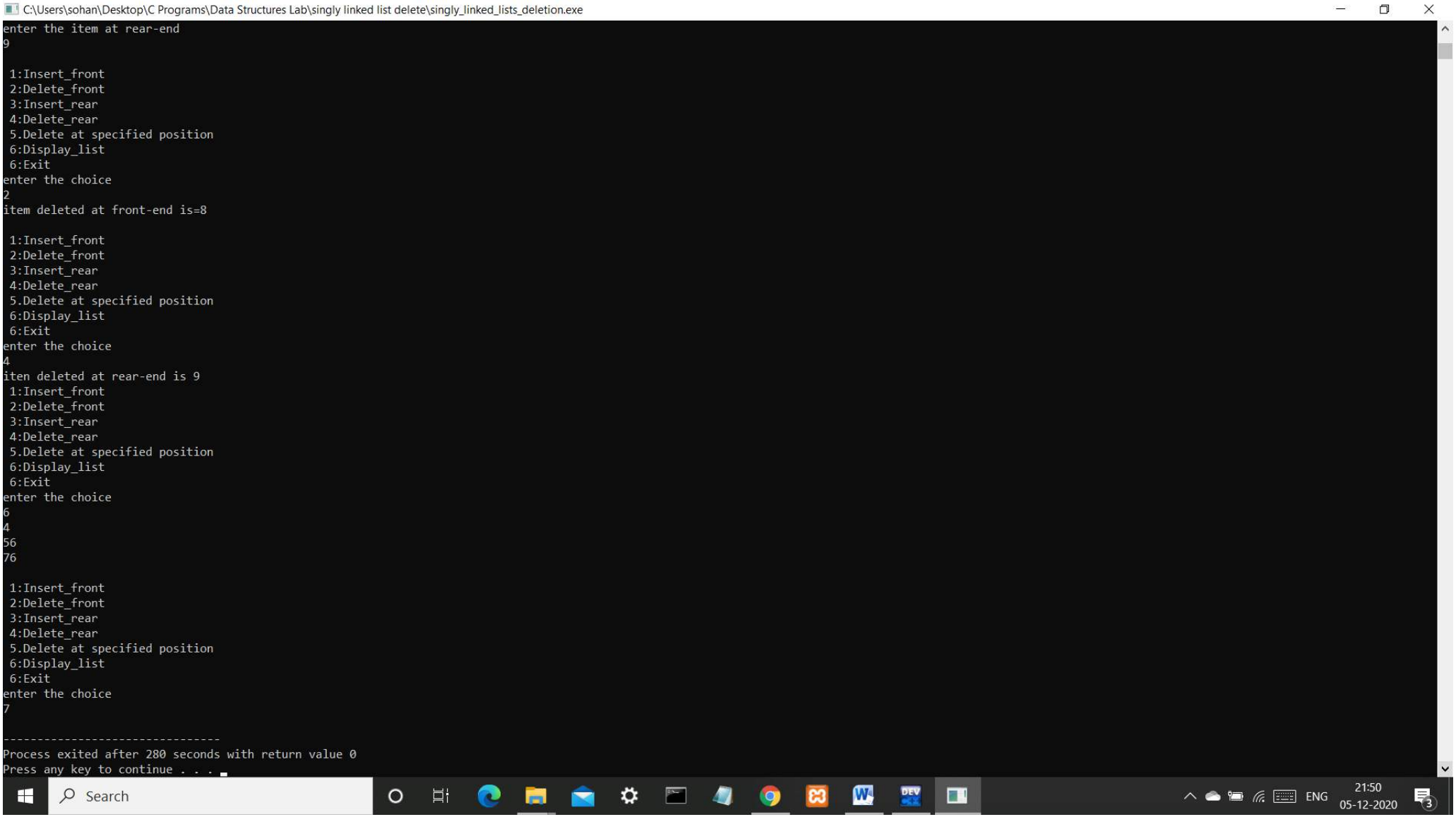

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\singly linked list delete\singly_linked_lists_deletion.cpp - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
[*] singly_linked_lists_deletion.cpp
147 int item,choice,pos;
148 NODE first=NULL;
149 system("cls");
150 for(;;)
151 {
152 printf("\n 1:Insert_front\n 2:Delete_front\n 3:Insert_rear\n 4:Delete_rear\n 5.Delete at specified position \n 6:Display_list\n 7:Ex
153 printf("enter the choice\n");
154 scanf("%d",&choice);
155 switch(choice)
156 {
157 case 1:printf("enter the item at front-end\n");
158 scanf("%d",&item);
159 first=insert_front(first,item);
160 break;
161 case 2:first=delete_front(first);
162 break;
163 case 3:printf("enter the item at rear-end\n");
164 scanf("%d",&item);
165 first=insert_rear(first,item);
166 break;
167 case 4:first=delete_rear(first);
168 break;
169 case 5:printf("enter the position of the item to be deleted: \n");
170 scanf("%d",&pos);
171 first=delete_pos(pos,first);
172 break;
173
174 case 6:display(first);
175 break;
176 default:exit(0);
177 break;
178 }
179 }
180 getch();
181 return 0;
182 }
Compiler Resources Compile Log Debug Find Results
Line: 149 Col: 15 Sel: 0 Lines: 182 Length: 3003 Insert Done parsing in 0.015 seconds
Search
ENG
21:49
05-12-2020

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\singly linked list delete\singly_linked_lists_deletion.exe

```
 1:Insert_front
 2:Delete_front
 3:Insert_rear
 4:Delete_rear
 5.Delete at specified position
 6:Display_list
 6:Exit
enter the choice
1
enter the item at front-end
56

 1:Insert_front
 2:Delete_front
 3:Insert_rear
 4:Delete_rear
 5.Delete at specified position
 6:Display_list
 6:Exit
enter the choice
3
enter the item at rear-end
69

 1:Insert_front
 2:Delete_front
 3:Insert_rear
 4:Delete_rear
 5.Delete at specified position
 6:Display_list
 6:Exit
enter the choice
1
enter the item at front-end
4

 1:Insert_front
 2:Delete_front
 3:Insert_rear
 4:Delete_rear
 5.Delete at specified position
 6:Display_list
 6:Exit
enter the choice
3
enter the item at rear-end
76

 1:Insert_front
```

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\singly linked list delete\singly_linked_lists_deletion.exe

```
enter the item at rear-end
76

 1:Insert_front
 2:Delete_front
 3:Insert_rear
 4:Delete_rear
 5.Delete at specified position
 6:Display_list
 6:Exit
enter the choice
5
enter the position of the item to be deleted:
3

 1:Insert_front
 2:Delete_front
 3:Insert_rear
 4:Delete_rear
 5.Delete at specified position
 6:Display_list
 6:Exit
enter the choice
1
enter the item at front-end
8

 1:Insert_front
 2:Delete_front
 3:Insert_rear
 4:Delete_rear
 5.Delete at specified position
 6:Display_list
 6:Exit
enter the choice
3
enter the item at rear-end
9

 1:Insert_front
 2:Delete_front
 3:Insert_rear
 4:Delete_rear
 5.Delete at specified position
 6:Display_list
 6:Exit
enter the choice
2
item deleted at front-end is=8
```


C:\Users\sohan\Desktop\C Programs\Data Structures Lab\singly linked list delete\singly_linked_lists_deletion.exe
enter the item at rear-end
9
1:Insert_front
2:Delete_front
3:Insert_rear
4:Delete_rear
5.Delete at specified position
6:Display_list
6:Exit
enter the choice
2
item deleted at front-end is=8
1:Insert_front
2:Delete_front
3:Insert_rear
4:Delete_rear
5.Delete at specified position
6:Display_list
6:Exit
enter the choice
4
iten deleted at rear-end is 9
1:Insert_front
2:Delete_front
3:Insert_rear
4:Delete_rear
5.Delete at specified position
6:Display_list
6:Exit
enter the choice
6
4
56
76
1:Insert_front
2:Delete_front
3:Insert_rear
4:Delete_rear
5.Delete at specified position
6:Display_list
6:Exit
enter the choice
7
Process exited after 280 seconds with return value 0
Press any key to continue . . .
Search
ENG
21:50
05-12-2020

Lab Program 7 :-

7) Write a program implement Single link list with following operations
a) Sort the linked list . b) Reverse linked list
c) Concentenation of two linked lists.

A)
```
#include<stdio.h>
#include<conio.h>
#include<malloc.h>
#include<stdlib.h>
Struct node {
int info;
Struct node *link;
};
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 insert_front(NODE first, int item) {
NODE temp;
temp = getnode();
temp->info = item;
temp->link = NULL;
if (first == NULL)
return temp;
temp->link = first;
```

```
    first = temp;
    return first;
}
NODE IF(NODE second, int item ){
NODE temp;
temp = getnode();
temp -> info = item;
temp -> link = NULL;
if (second == NULL)
return temp;
temp -> link = second;
second = temp;
return second;
}
NODE delete_front(NODE first){
NODE temp;
if( first == NULL){
printf("list is empty cannot delete \n");
return first;
}
temp = first;
temp = temp -> link;
printf(" item deleted at front-end is = %d \n", first -> info);
free(first);
return temp;
}
NODE insert_rear(NODE first, int item){
NODE temp, cur;
temp = getnode();
temp -> info = item;
temp -> link = NULL;
if (first == NULL)
return temp;
cur = first;
```

```
while (cur -> link != NULL)
cur = cur -> link;
cur -> link = temp;
return first;
}

NODE IR (NODE second, int item){
NODE temp, cur;
temp = getnode();
temp->info = item;
temp -> link = NULL;
if (second == NULL)
return temp;
cur = second;
while (cur -> link != NULL)
cur = cur -> link;
cur -> link = temp;
return second;
}

NODE delete_rear(NODE first){
NODE cur, prev;
if(first == NULL){
printf ("list is empty cannot delete \n");
return first;
}
if( first -> link == NULL) {
printf ("item deleted is %d \n", first -> info);
free (first);
return NULL;
}
prev = NULL;
cur = first;
while(cur -> link != NULL) {
prev = cur;
cur = cur -> link;
}
```

```
printf("item deleted at rear end is %d", cur->info);
free(cur);
prev->link = NULL;
return first;
}

NODE insert_pos(int item, int pos, NODE first){
NODE temp;
NODE prev, cur;
int count;
temp = getnode();
temp->info = item;
temp->link = NULL;
if(first == NULL && pos == 1)
return temp;
if(first == NULL){
printf("invalid pos\n");
return first;
}
if(pos == 1){
temp->link = first;
return temp;
}
count = 1;
prev = NULL;
cur = first;
while(cur != NULL && count != pos){
prev = cur;
cur = cur->link;
count++;
}
if(count == pos){
prev->link = temp;
temp->link = cur;
return first;
}
```

```
printf("Invalid position \n");
return first;
}
NODE delete_pos(int pos, NODE first){
NODE cur;
NODE prev;
int count;
if(first==NULL || pos<=0){
printf("invalid position \n");
return NULL;
}
if(pos==1){
cur=first;
first=first->link;
freenode(cur);
return first;
}
prev=NULL;
cur=first;
count=1;
while(cur!=NULL){
if(count==pos)
break;
prev=cur;
cur=cur->link;
count++;
}
if(count!=pos){
printf("invalid position \n");
return first;
}
prev->link=cur->link;
freenode(cur);
return first;
}
```

```
NODE reverse(NODE first){
NODE cur, temp;
cur =NULL;
while(first != NULL){
temp = first;
first = first -> link;
temp->link = cur;
cur = temp;
}
return cur;
}
NODE asc(NOD first){
NODE prev = first;
NODE cur = NULL;
int temp;
if( first == NULL){
return 0;
}
else{
while(prev = NULL){
cur = prev -> link;
while (cur != NULL){
if(prev -> info > cur -> info){
temp = prev -> info;
prev -> info = cur -> info;
cur -> info = temp;
}
cur = cur -> link;
} prev = prev -> link;
}}
return first;
}
NODE desc(NODE first){
NODE prev = first;
```

```
NODE cur = NULL;
int temp;
if (first == NULL){
return 0;
}
else{
while ( prev1 = NULL){
cur = prev -> link;
while (cur1 = NULL){
if (prev -> info < cur -> info ){
temp = prev -> info ;
prev -> info = cur -> info ;
cur -> info = temp;
} cur = cur -> link;
} prev = prev -> link;
}}
return first;
}
NODE concate (NODE first, NODE second){
NODE cur;
if (first == NULL)
return second;
if (second == NULL)
return first;
cur = first;
while ( cur -> link != NULL){
cur = cur -> link;
}
cur -> link = second;
return first;
}
void display (NODE first){
NODE temp;
if ( first == NULL)
```

```
printf("list empty cannot display items \n");
for(temp=first; temp!=NULL; temp=temp->link){
printf("%d \n", temp->info);
}

int main(){
int item, choice, pos, element, option, choice2, item1, num;
system("cls");
NODE first=NULL;
NODE second = NULL;
for(;;){
printf("\n 1: insert_front \n 2: delete_front \n 3: insert_rear \n
4: delete_rear \n 5: random_position \n 6: reverse \n 7: sort \n 8:
concat \n 9: display \n 10: Exit \n");
scanf("%d", &choice);
switch(choice){
case 1: printf("enter the item at front_end \n");
        scanf("%d", &item);
        first = insert_front(first, item);
        break;
case 2: first = delete_front(first);
break;
case 3: printf("enter the item at rear-end \n");
        scanf("%d", &item);
        first = insert_rear(first, item);
        break;
case 4: first = delete_rear(first);
break;
case 5: printf("press 1 to insert or 2 to delete at any
        desired position \n");
        scanf("%d", &element);
if(element==1){
printf("enter the position to insert \n");
scanf("%d", &pos);
```

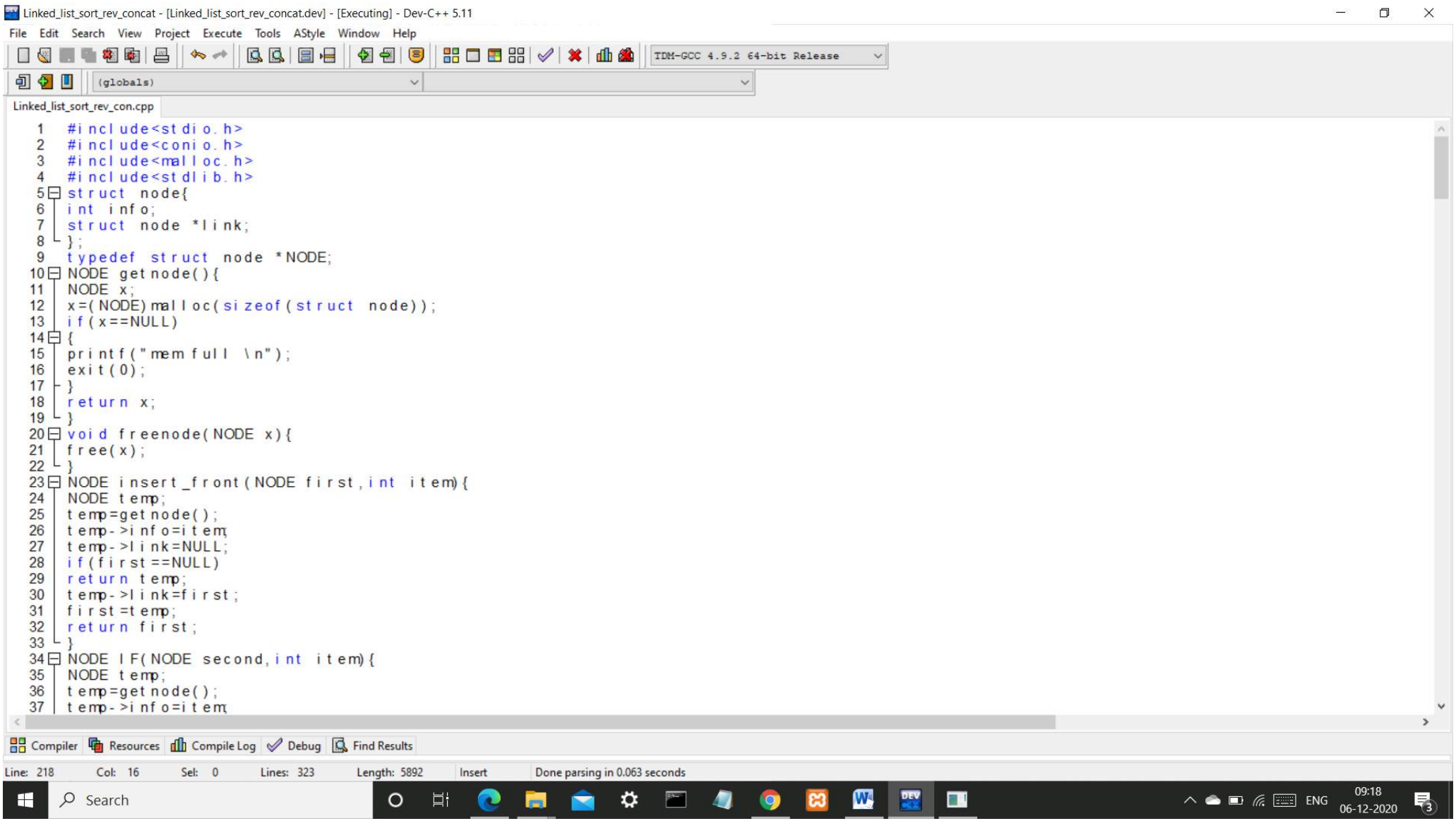
```
printf ("enter the item to insert : \n");
scanf ("%d", &item);
first = insert_pos (item, pos, first); }
if (element == 2) {
printf ("enter position to delete \n");
scanf ("%d", &pos);
first = delete_pos (pos, first);
}
break;
case 6: first = reverse (first);
        break;
case 7: printf ("press 1 for ascending : sort &
                 2 for descending sort : \n");
scanf ("%d", &option);
if (option == 1)
first = asc (first);
if (option == 2)
first = des (first);
break;
case 8: printf ("create a second list \n");
        printf ("enter the number of elements in second list : \n");
scanf ("%d", &num);
for (int i = 1; i <= num; i++) {
printf ("\n press 1 to insert front & 2 to insert rear \n");
scanf ("%d", &choice2);
if (choice2 == 1) {
printf ("enter the item at the front end \n");
scanf ("%d", &item1);
}
if (choice2 == 2) {
printf ("enter the item at rear end \n");
scanf ("%d", &item);
second = IR (second, item1);
}}
```

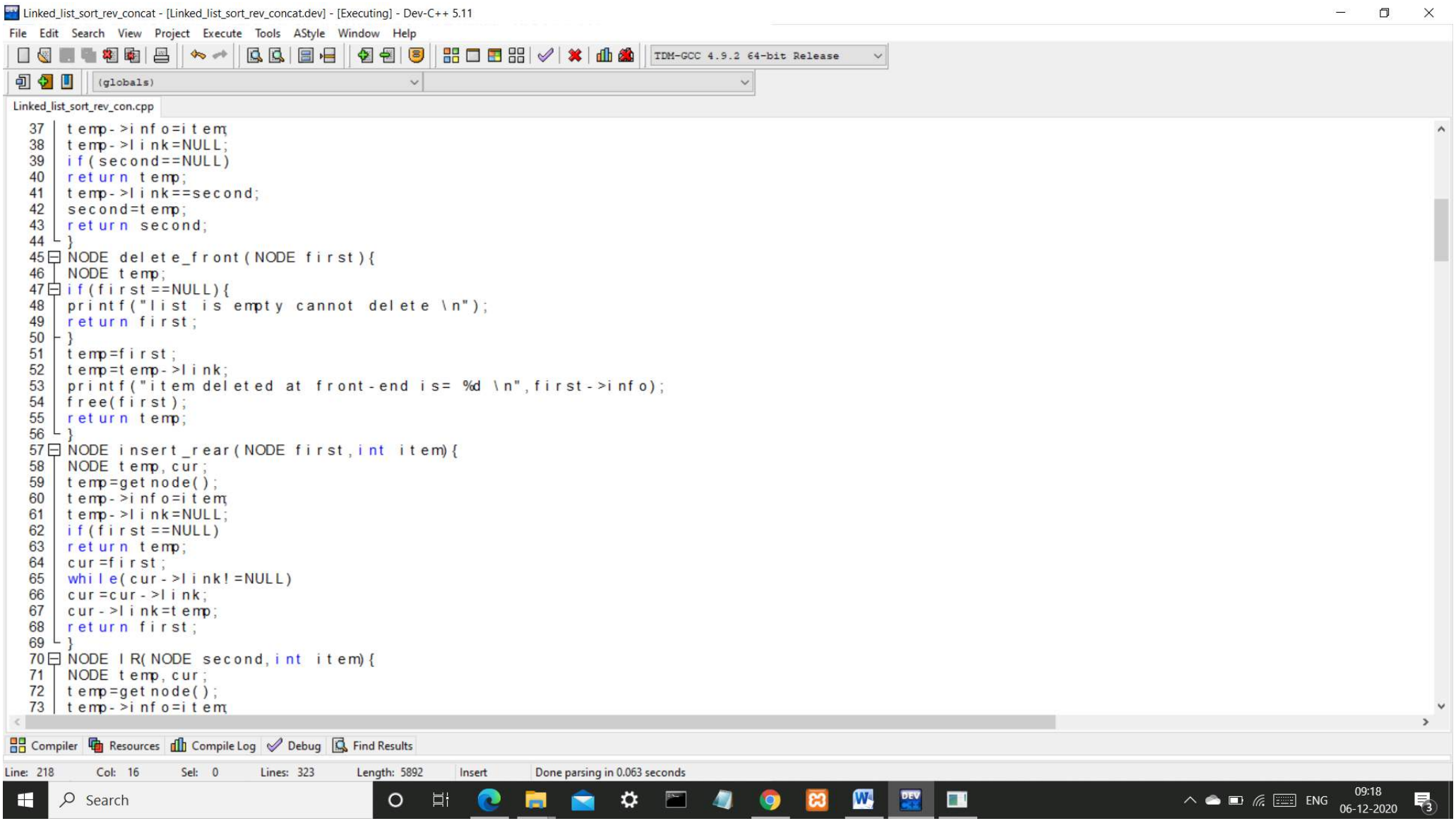
10		c) To dis...

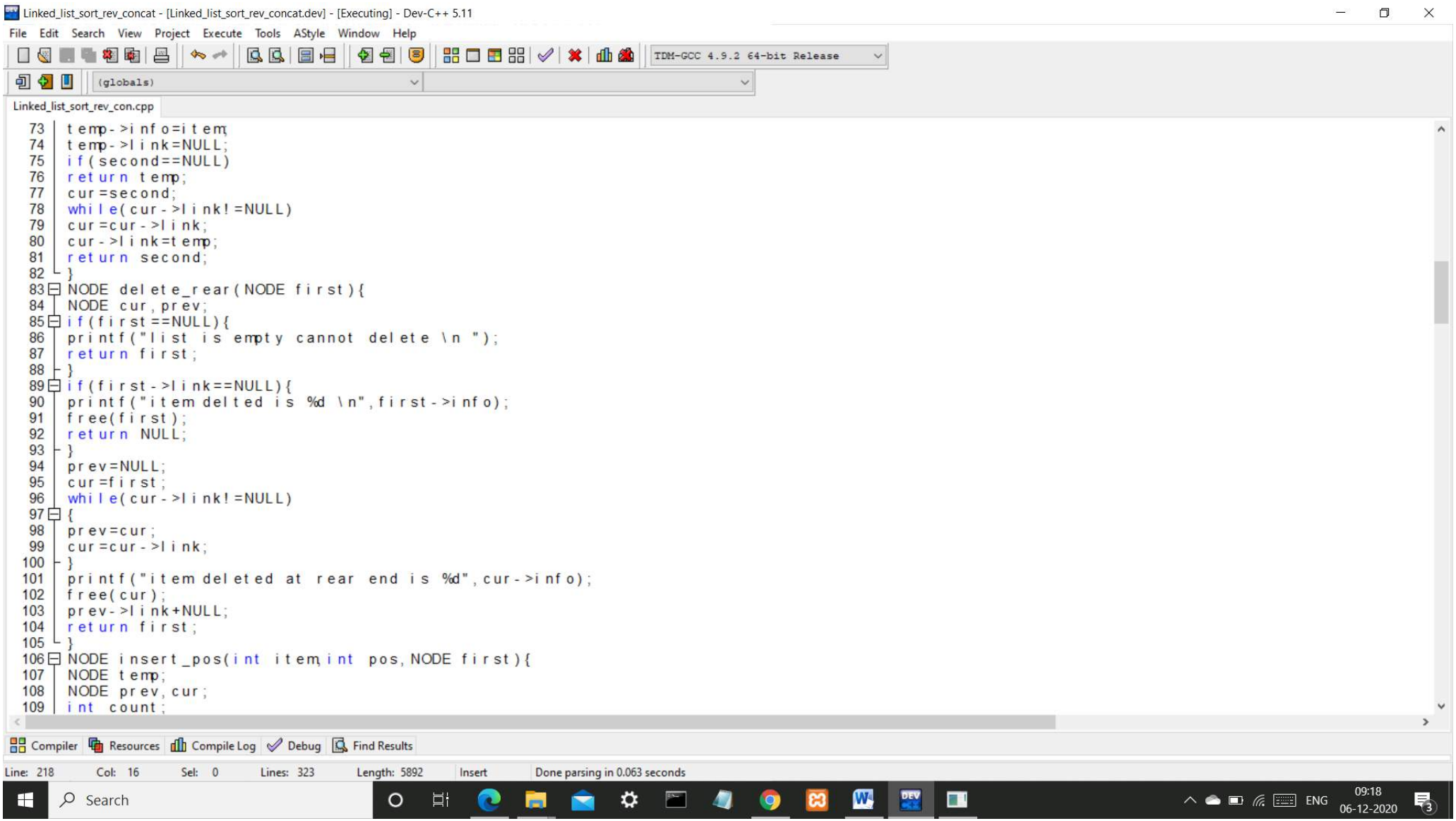
G Proposed Alternate Assessment Tool Plan *(if applic...*

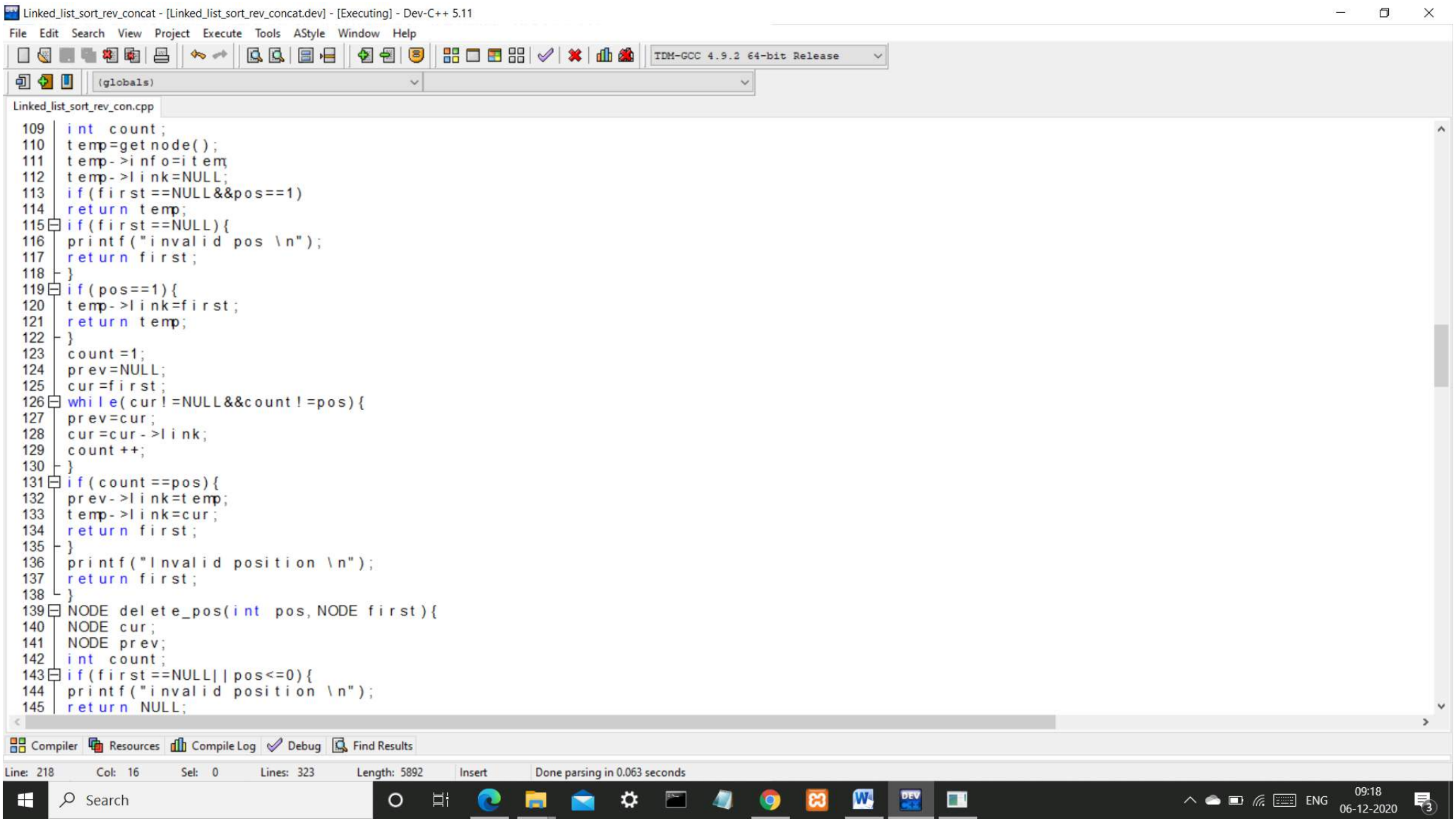
... Question paper format	One Question to be asked for 20 Marks
	...stion to be asked for 20 Marks
	...d for 20 Marks each

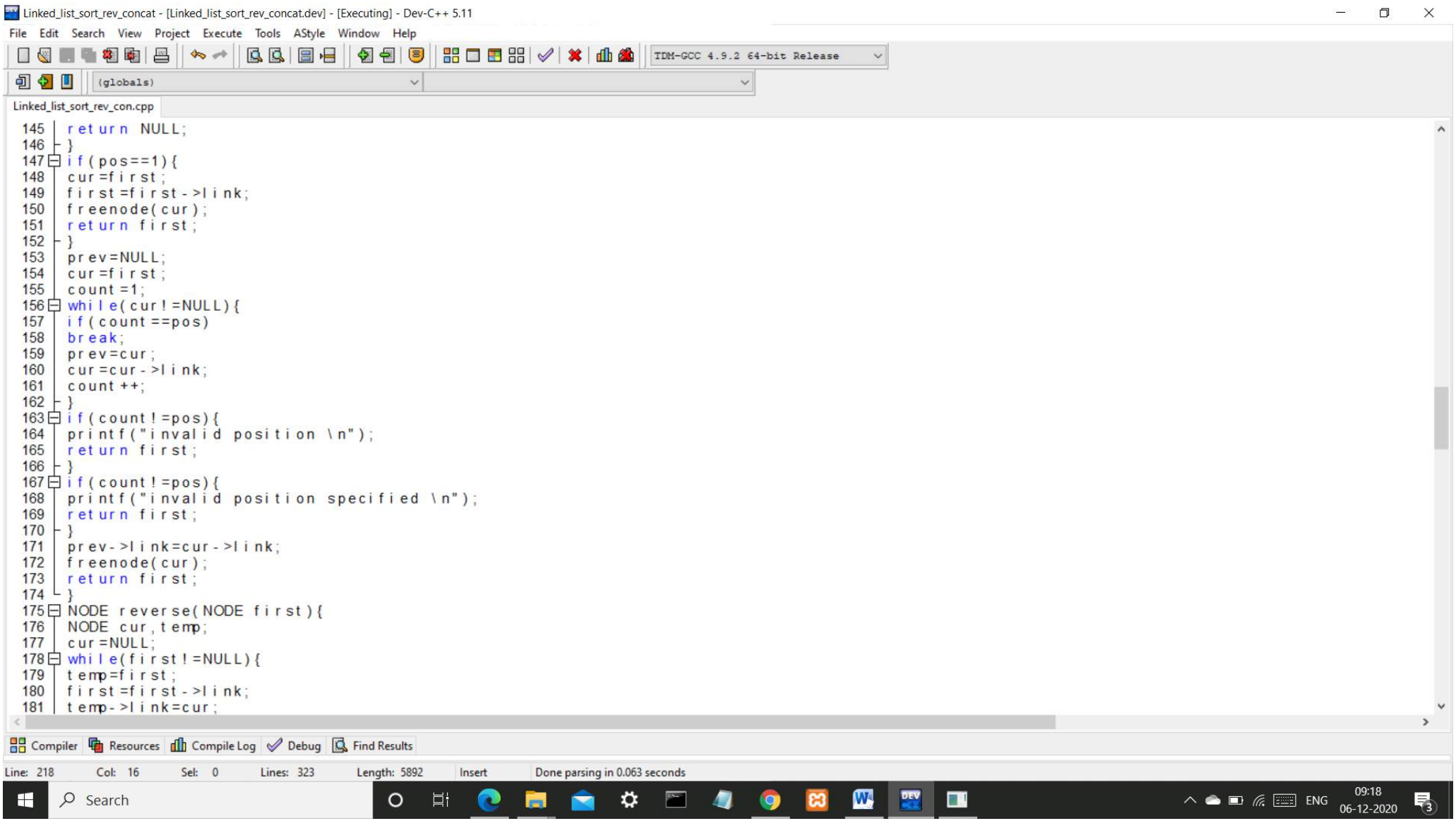
```
first = concate (first, second);
break;
case 9 : display (first);
break;
default : exit(0);
break;
}
getch();
return 0;
}
```

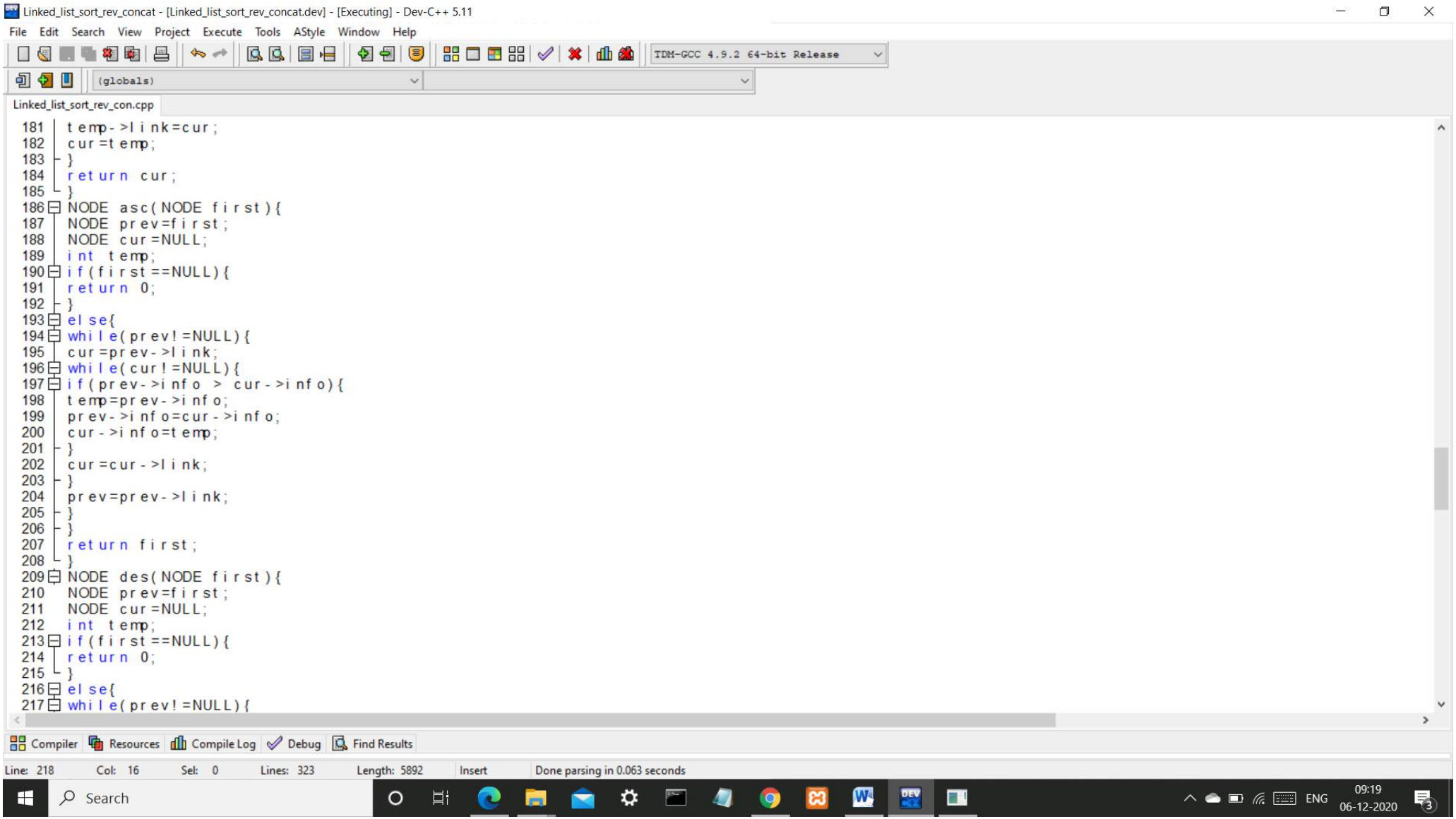

Linked_list_sort_rev_concat - [Linked_list_sort_rev_concat.dev] - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Linked_list_sort_rev_con.cpp
1 #include<stdio.h>
2 #include<conio.h>
3 #include<malloc.h>
4 #include<stdlib.h>
5 struct node{
6 int info;
7 struct node *link;
8 };
9 typedef struct node *NODE;
10 NODE getnode(){
11 NODE x;
12 x=(NODE)malloc(sizeof(struct node));
13 if(x==NULL)
14 {
15 printf("mem full \n");
16 exit(0);
17 }
18 return x;
19 }
20 void freenode(NODE x){
21 free(x);
22 }
23 NODE insert_front(NODE first,int item){
24 NODE temp;
25 temp=getnode();
26 temp->info=item;
27 temp->link=NULL;
28 if(first==NULL)
29 return temp;
30 temp->link=first;
31 first=temp;
32 return first;
33 }
34 NODE IF(NODE second,int item){
35 NODE temp;
36 temp=getnode();
37 temp->info=item;
Compiler Resources Compile Log Debug Find Results
Line: 218 Col: 16 Sel: 0 Lines: 323 Length: 5892 Insert Done parsing in 0.063 seconds
Search
ENG
09:18
06-12-2020

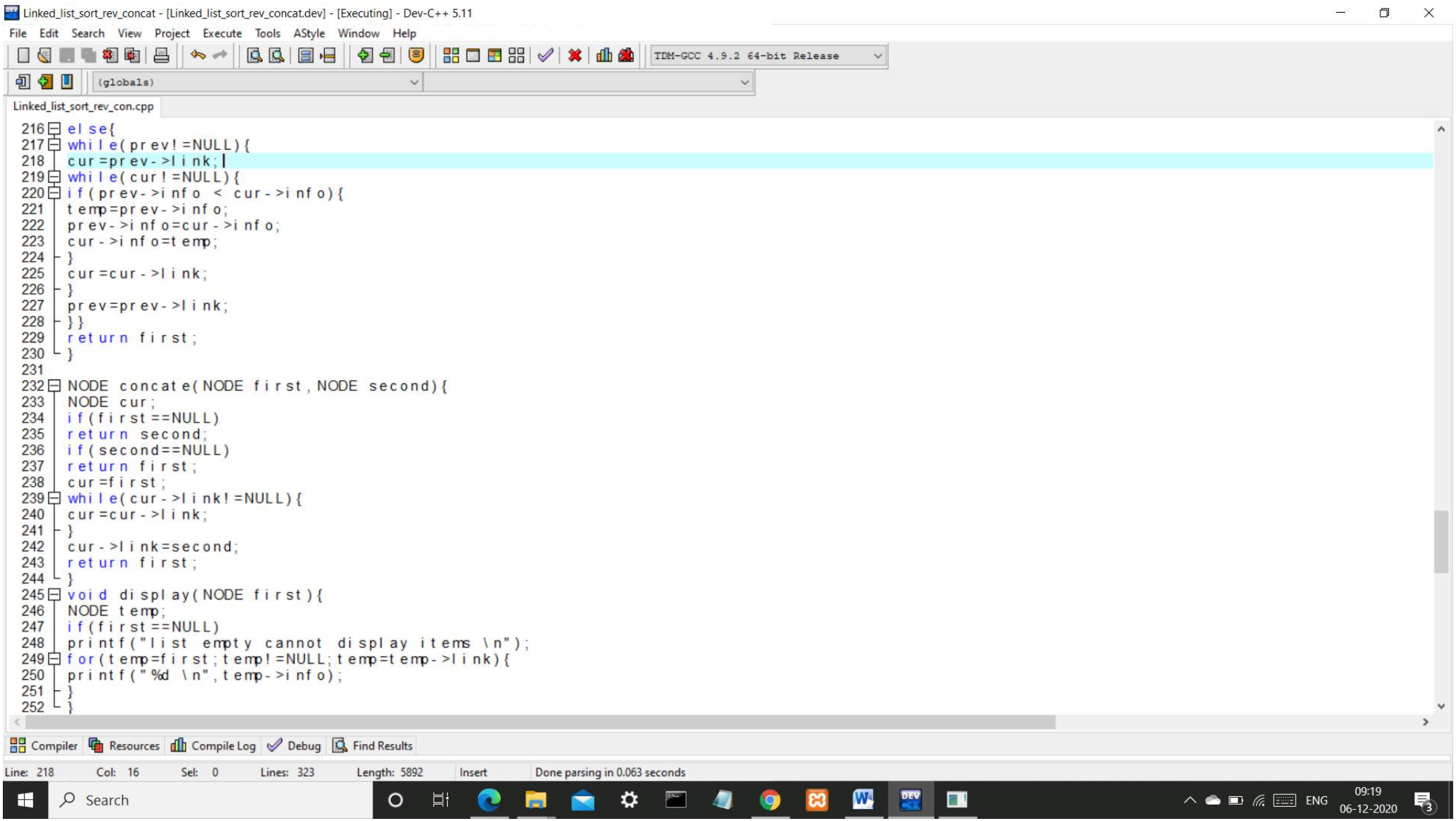

Linked_list_sort_rev_concat - [Linked_list_sort_rev_concat.dev] - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Linked_list_sort_rev_con.cpp
temp->info=item;
temp->link=NULL;
if(second==NULL)
return temp;
temp->link==second;
second=temp;
return second;
}
NODE delete_front(NODE first){
NODE temp;
if(first==NULL){
printf("list is empty cannot delete \n");
return first;
}
temp=first;
temp=temp->link;
printf("item deleted at front-end is= %d \n",first->info);
free(first);
return temp;
}
NODE insert_rear(NODE first,int item){
NODE temp,cur;
temp=getnode();
temp->info=item;
temp->link=NULL;
if(first==NULL)
return temp;
cur=first;
while(cur->link!=NULL)
cur=cur->link;
cur->link=temp;
return first;
}
NODE IR(NODE second,int item){
NODE temp,cur;
temp=getnode();
temp->info=item;
Compiler Resources Compile Log Debug Find Results
Line: 218 Col: 16 Sel: 0 Lines: 323 Length: 5892 Insert Done parsing in 0.063 seconds
Search
ENG
09:18
06-12-2020

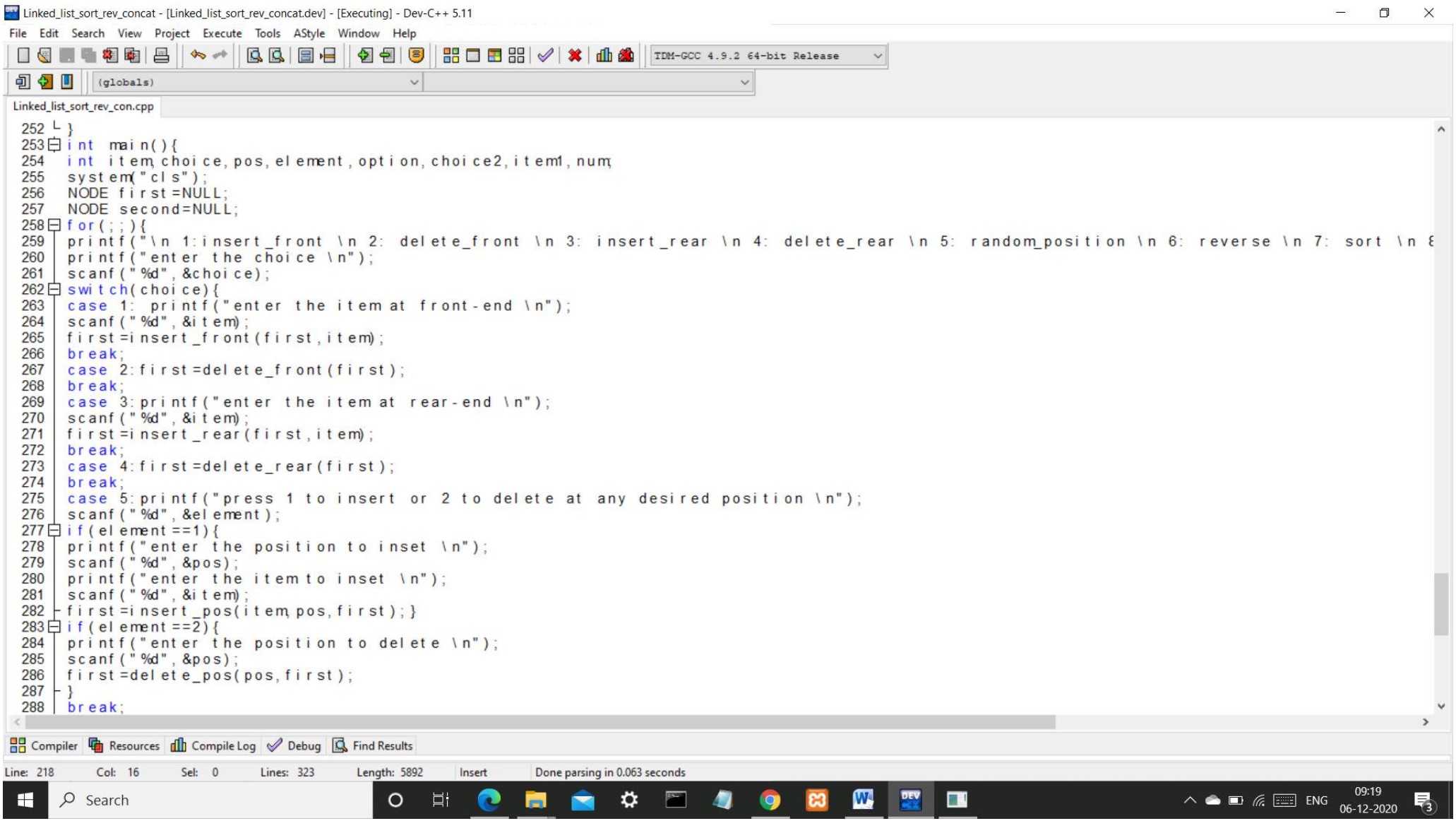

Linked_list_sort_rev_concat - [Linked_list_sort_rev_concat.dev] - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Linked_list_sort_rev_con.cpp
temp->info=item;
temp->link=NULL;
if(second==NULL)
return temp;
cur=second;
while(cur->link!=NULL)
cur=cur->link;
cur->link=temp;
return second;
}
NODE delete_rear(NODE first){
NODE cur,prev;
if(first==NULL){
printf("list is empty cannot delete \n ");
return first;
}
if(first->link==NULL){
printf("item delted is %d \n",first->info);
free(first);
return NULL;
}
prev=NULL;
cur=first;
while(cur->link!=NULL)
{
prev=cur;
cur=cur->link;
}
printf("item deleted at rear end is %d",cur->info);
free(cur);
prev->link+NULL;
return first;
}
NODE insert_pos(int item,int pos,NODE first){
NODE temp;
NODE prev,cur;
int count;
Compiler Resources Compile Log Debug Find Results
Line: 218 Col: 16 Sel: 0 Lines: 323 Length: 5892 Insert Done parsing in 0.063 seconds
Search
ENG
09:18
06-12-2020

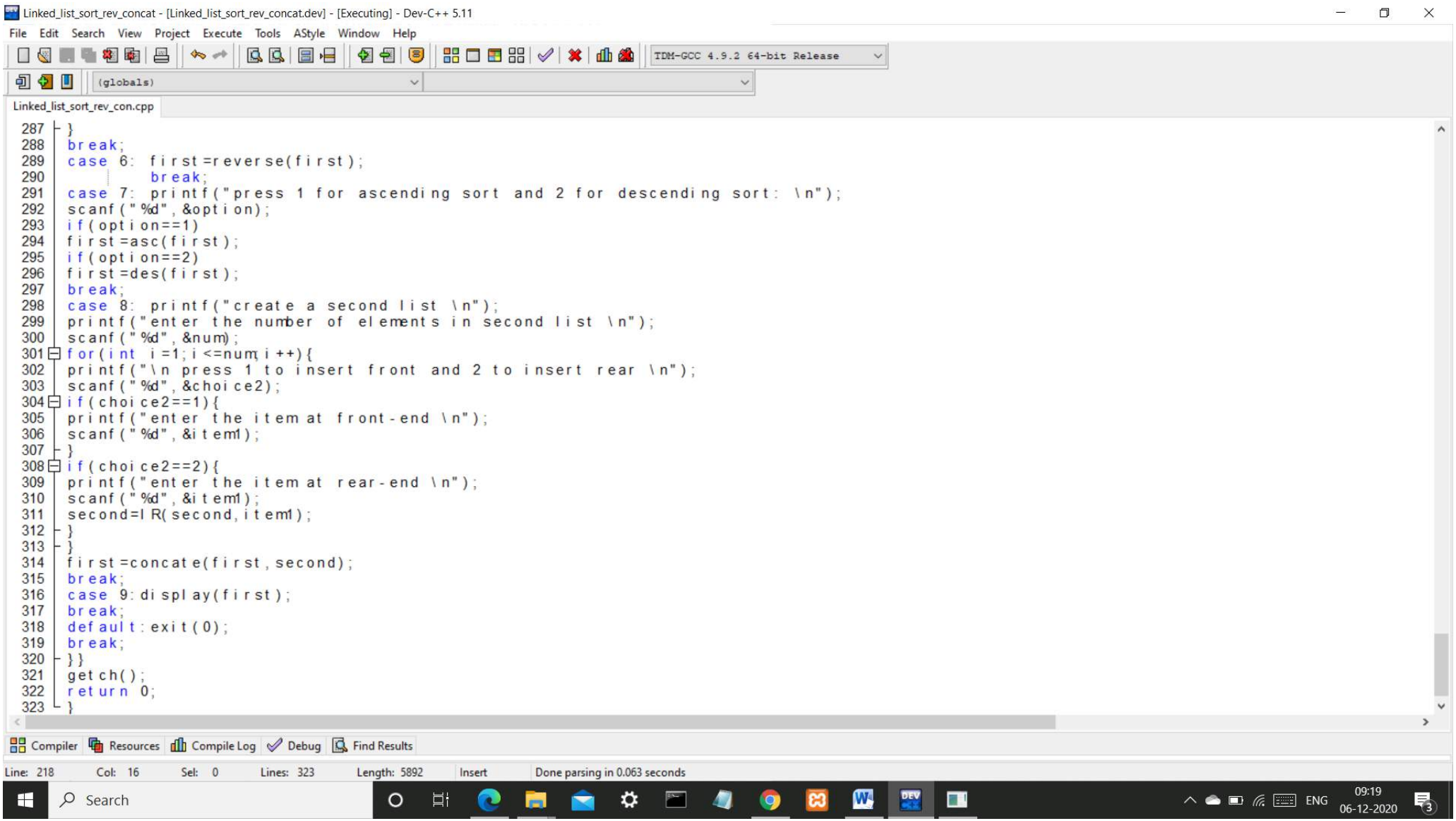

Linked_list_sort_rev_concat - [Linked_list_sort_rev_concat.dev] - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Linked_list_sort_rev_con.cpp
109 int count;
110 temp=getnode();
111 temp->info=item;
112 temp->link=NULL;
113 if(first==NULL&&pos==1)
114 return temp;
115 if(first==NULL){
116 printf("invalid pos \n");
117 return first;
118 }
119 if(pos==1){
120 temp->link=first;
121 return temp;
122 }
123 count=1;
124 prev=NULL;
125 cur=first;
126 while(cur!=NULL&&count!=pos){
127 prev=cur;
128 cur=cur->link;
129 count++;
130 }
131 if(count==pos){
132 prev->link=temp;
133 temp->link=cur;
134 return first;
135 }
136 printf("Invalid position \n");
137 return first;
138 }
139 NODE delete_pos(int pos,NODE first){
140 NODE cur;
141 NODE prev;
142 int count;
143 if(first==NULL||pos<=0){
144 printf("invalid position \n");
145 return NULL;
Compiler Resources Compile Log Debug Find Results
Line: 218 Col: 16 Sel: 0 Lines: 323 Length: 5892 Insert Done parsing in 0.063 seconds
Search
ENG
09:18
06-12-2020


Linked_list_sort_rev_concat - [Linked_list_sort_rev_concat.dev] - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Linked_list_sort_rev_con.cpp
return NULL;
}
if(pos==1){
cur=first;
first=first->link;
freenode(cur);
return first;
}
prev=NULL;
cur=first;
count=1;
while(cur!=NULL){
if(count==pos)
break;
prev=cur;
cur=cur->link;
count++;
}
if(count!=pos){
printf("invalid position \n");
return first;
}
if(count!=pos){
printf("invalid position specified \n");
return first;
}
prev->link=cur->link;
freenode(cur);
return first;
}
NODE reverse(NODE first){
NODE cur,temp;
cur=NULL;
while(first!=NULL){
temp=first;
first=first->link;
temp->link=cur;
Compiler Resources Compile Log Debug Find Results
Line: 218 Col: 16 Sel: 0 Lines: 323 Length: 5892 Insert Done parsing in 0.063 seconds
Search
ENG
09:18
06-12-2020


Linked_list_sort_rev_concat - [Linked_list_sort_rev_concat.dev] - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Linked_list_sort_rev_con.cpp
temp->link=cur;
cur=temp;
}
return cur;
}
NODE asc(NODE first){
NODE prev=first;
NODE cur=NULL;
int temp;
if(first==NULL){
return 0;
}
else{
while(prev!=NULL){
cur=prev->link;
while(cur!=NULL){
if(prev->info > cur->info){
temp=prev->info;
prev->info=cur->info;
cur->info=temp;
}
cur=cur->link;
}
prev=prev->link;
}
}
return first;
}
NODE des(NODE first){
NODE prev=first;
NODE cur=NULL;
int temp;
if(first==NULL){
return 0;
}
else{
while(prev!=NULL){
Compiler Resources Compile Log Debug Find Results
Line: 218 Col: 16 Sel: 0 Lines: 323 Length: 5892 Insert Done parsing in 0.063 seconds
Search
ENG
09:19
06-12-2020


Linked_list_sort_rev_concat - [Linked_list_sort_rev_concat.dev] - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Linked_list_sort_rev_con.cpp
else{
while(prev!=NULL){
cur=prev->link;
while(cur!=NULL){
if(prev->info < cur->info){
temp=prev->info;
prev->info=cur->info;
cur->info=temp;
}
cur=cur->link;
}
prev=prev->link;
}}
return first;
}
NODE concate(NODE first,NODE second){
NODE cur;
if(first==NULL)
return second;
if(second==NULL)
return first;
cur=first;
while(cur->link!=NULL){
cur=cur->link;
}
cur->link=second;
return first;
}
void display(NODE first){
NODE temp;
if(first==NULL)
printf("list empty cannot display items \n");
for(temp=first;temp!=NULL;temp=temp->link){
printf("%d \n",temp->info);
}
}
Compiler Resources Compile Log Debug Find Results
Line: 218 Col: 16 Sel: 0 Lines: 323 Length: 5892 Insert Done parsing in 0.063 seconds
Search
ENG
09:19
06-12-2020


Linked_list_sort_rev_concat - [Linked_list_sort_rev_concat.dev] - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Linked_list_sort_rev_con.cpp
252 }
253 int main(){
254 int item,choice,pos,element,option,choice2,item1,num;
255 system("cls");
256 NODE first=NULL;
257 NODE second=NULL;
258 for(;;){
259 printf("\n 1:insert_front \n 2: delete_front \n 3: insert_rear \n 4: delete_rear \n 5: random_position \n 6: reverse \n 7: sort \n 8
260 printf("enter the choice \n");
261 scanf("%d",&choice);
262 switch(choice){
263 case 1: printf("enter the item at front-end \n");
264 scanf("%d",&item);
265 first=insert_front(first,item);
266 break;
267 case 2:first=delete_front(first);
268 break;
269 case 3:printf("enter the item at rear-end \n");
270 scanf("%d",&item);
271 first=insert_rear(first,item);
272 break;
273 case 4:first=delete_rear(first);
274 break;
275 case 5:printf("press 1 to insert or 2 to delete at any desired position \n");
276 scanf("%d",&element);
277 if(element==1){
278 printf("enter the position to inset \n");
279 scanf("%d",&pos);
280 printf("enter the item to inset \n");
281 scanf("%d",&item);
282 first=insert_pos(item,pos,first);}
283 if(element==2){
284 printf("enter the position to delete \n");
285 scanf("%d",&pos);
286 first=delete_pos(pos,first);
287 }
288 break;
Compiler Resources Compile Log Debug Find Results
Line: 218 Col: 16 Sel: 0 Lines: 323 Length: 5892 Insert Done parsing in 0.063 seconds
Search
ENG
09:19
06-12-2020


Linked_list_sort_rev_concat - [Linked_list_sort_rev_concat.dev] - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
Linked_list_sort_rev_con.cpp
}
break;
case 6: first=reverse(first);
break;
case 7: printf("press 1 for ascending sort and 2 for descending sort: \n");
scanf("%d",&option);
if(option==1)
first=asc(first);
if(option==2)
first=des(first);
break;
case 8: printf("create a second list \n");
printf("enter the number of elements in second list \n");
scanf("%d",&num);
for(int i=1;i<=num;i++){
printf("\n press 1 to insert front and 2 to insert rear \n");
scanf("%d",&choice2);
if(choice2==1){
printf("enter the item at front-end \n");
scanf("%d",&item1);
}
if(choice2==2){
printf("enter the item at rear-end \n");
scanf("%d",&item1);
second=IR(second,item1);
}
}
first=concate(first,second);
break;
case 9:display(first);
break;
default:exit(0);
break;
}}
getch();
return 0;
}
Compiler Resources Compile Log Debug Find Results
Line: 218 Col: 16 Sel: 0 Lines: 323 Length: 5892 Insert Done parsing in 0.063 seconds
Search
ENG
09:19
06-12-2020

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\linked list sort rev concatenation\Linked_list_sort_rev_concat.exe

```
 1:insert_front
 2: delete_front
 3: insert_rear
 4: delete_rear
 5: random_position
 6: reverse
 7: sort
 8: concate
 9: display_list
 10: exit
enter the choice
1
enter the item at front-end
3

 1:insert_front
 2: delete_front
 3: insert_rear
 4: delete_rear
 5: random_position
 6: reverse
 7: sort
 8: concate
 9: display_list
 10: exit
enter the choice
1
enter the item at front-end
5

 1:insert_front
 2: delete_front
 3: insert_rear
 4: delete_rear
 5: random_position
 6: reverse
 7: sort
 8: concate
 9: display_list
 10: exit
enter the choice
3
enter the item at rear-end
2

 1:insert_front
 2: delete_front
 3: insert_rear
 4: delete_rear
```

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\linked list sort rev concatenation\Linked_list_sort_rev_concat.exe

```
3
enter the item at rear-end
2

 1:insert_front
 2: delete_front
 3: insert_rear
 4: delete_rear
 5: random_position
 6: reverse
 7: sort
 8: concate
 9: display_list
 10: exit
enter the choice
3
enter the item at rear-end
4

 1:insert_front
 2: delete_front
 3: insert_rear
 4: delete_rear
 5: random_position
 6: reverse
 7: sort
 8: concate
 9: display_list
 10: exit
enter the choice
6

 1:insert_front
 2: delete_front
 3: insert_rear
 4: delete_rear
 5: random_position
 6: reverse
 7: sort
 8: concate
 9: display_list
 10: exit
enter the choice
9
4
2
3
5

 1:insert_front
```

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\linked list sort rev concatenation\Linked_list_sort_rev_concat.exe

```
 1:insert_front
 2: delete_front
 3: insert_rear
 4: delete_rear
 5: random_position
 6: reverse
 7: sort
 8: concate
 9: display_list
 10: exit
enter the choice
7
press 1 for ascending sort and 2 for descending sort:
1

 1:insert_front
 2: delete_front
 3: insert_rear
 4: delete_rear
 5: random_position
 6: reverse
 7: sort
 8: concate
 9: display_list
 10: exit
enter the choice
9
2
3
4
5

 1:insert_front
 2: delete_front
 3: insert_rear
 4: delete_rear
 5: random_position
 6: reverse
 7: sort
 8: concate
 9: display_list
 10: exit
enter the choice
8
create a second list
enter the number of elements in second list
3

 press 1 to insert front and 2 to insert rear
```

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\linked list sort rev concatenation\Linked_list_sort_rev_concat.exe

```
1
enter the item at front-end
33

 press 1 to insert front and 2 to insert rear
2
enter the item at rear-end
22

 press 1 to insert front and 2 to insert rear
1
enter the item at front-end
11

 1:insert_front
 2: delete_front
 3: insert_rear
 4: delete_rear
 5: random_position
 6: reverse
 7: sort
 8: concate
 9: display_list
 10: exit
enter the choice
7
press 1 for ascending sort and 2 for descending sort:
2

 1:insert_front
 2: delete_front
 3: insert_rear
 4: delete_rear
 5: random_position
 6: reverse
 7: sort
 8: concate
 9: display_list
 10: exit
enter the choice
9
22
5
4
3
2

 1:insert_front
 2: delete_front
 3: insert_rear
```

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\linked list sort rev concatenation\Linked_list_sort_rev_concat.exe

```
 10: exit
enter the choice
7
press 1 for ascending sort and 2 for descending sort:
2

 1:insert_front
 2: delete_front
 3: insert_rear
 4: delete_rear
 5: random_position
 6: reverse
 7: sort
 8: concate
 9: display_list
 10: exit
enter the choice
9
22
5
4
3
2

 1:insert_front
 2: delete_front
 3: insert_rear
 4: delete_rear
 5: random_position
 6: reverse
 7: sort
 8: concate
 9: display_list
 10: exit
enter the choice
10

--------------------------------
Process exited after 54.1 seconds with return value 0
Press any key to continue . . .
```

Lab Program 8 :-

8) Write a program to implement Stack & Queue using linked representation.

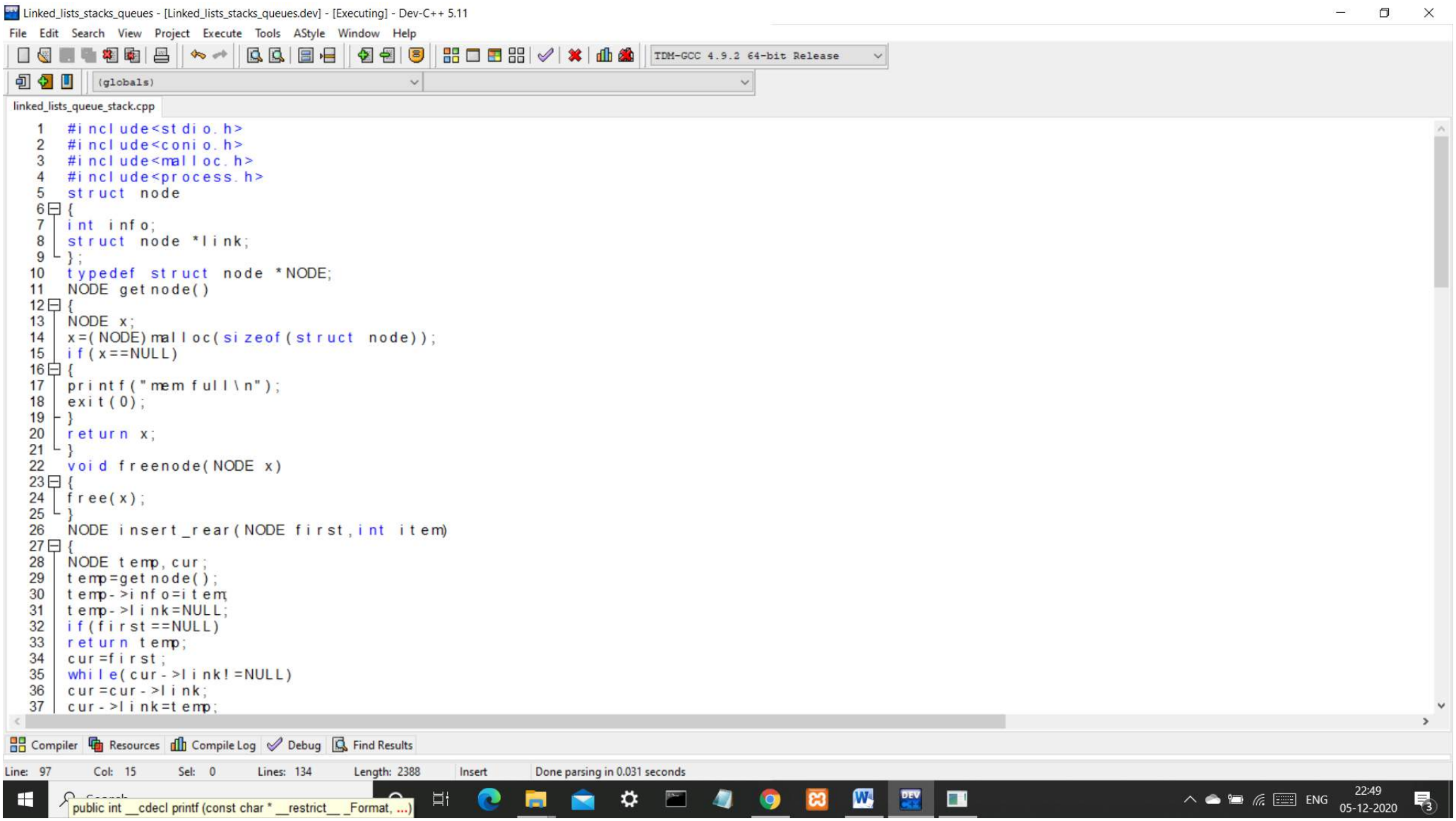
A

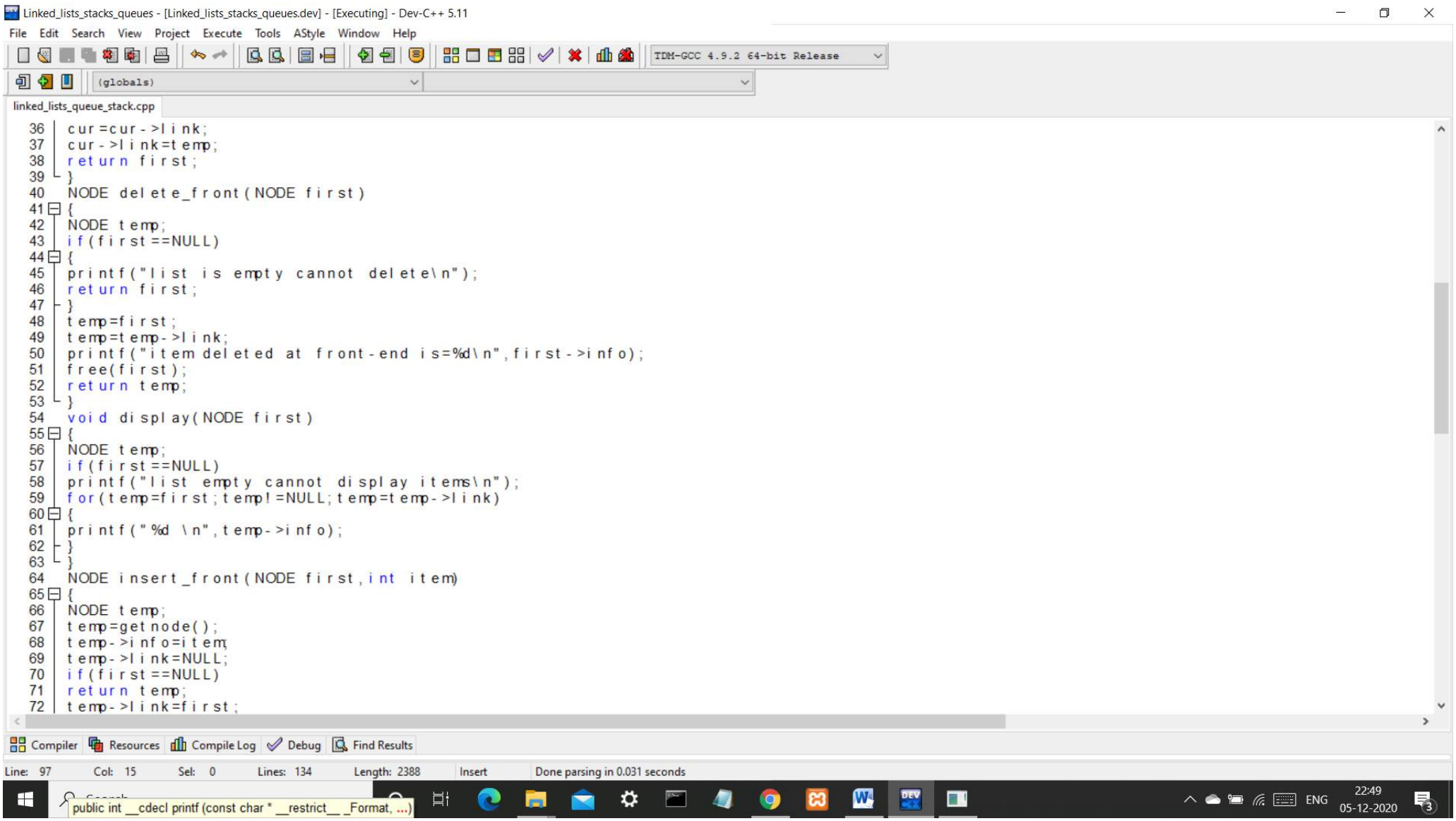
```
#include<stdio.h>
#include<conio.h>
#include<malloc.h>
#include<process.h>
Struct node
{
int info;
struct node *link;
};
typedef struct struct node *NODE;
NODE getnode()
{
NODE x;
x = (NODE) malloc(sizeof(struct node));
if(x==NULL)
{
printf("mem full \n");
exit(0);
}
return x;
}
void freenode(NODE x)
{
free(x);
}
NODE insert_rear(NODE first, int item)
{
NODE temp, cur;
temp = getnode();
temp->info = item;
```

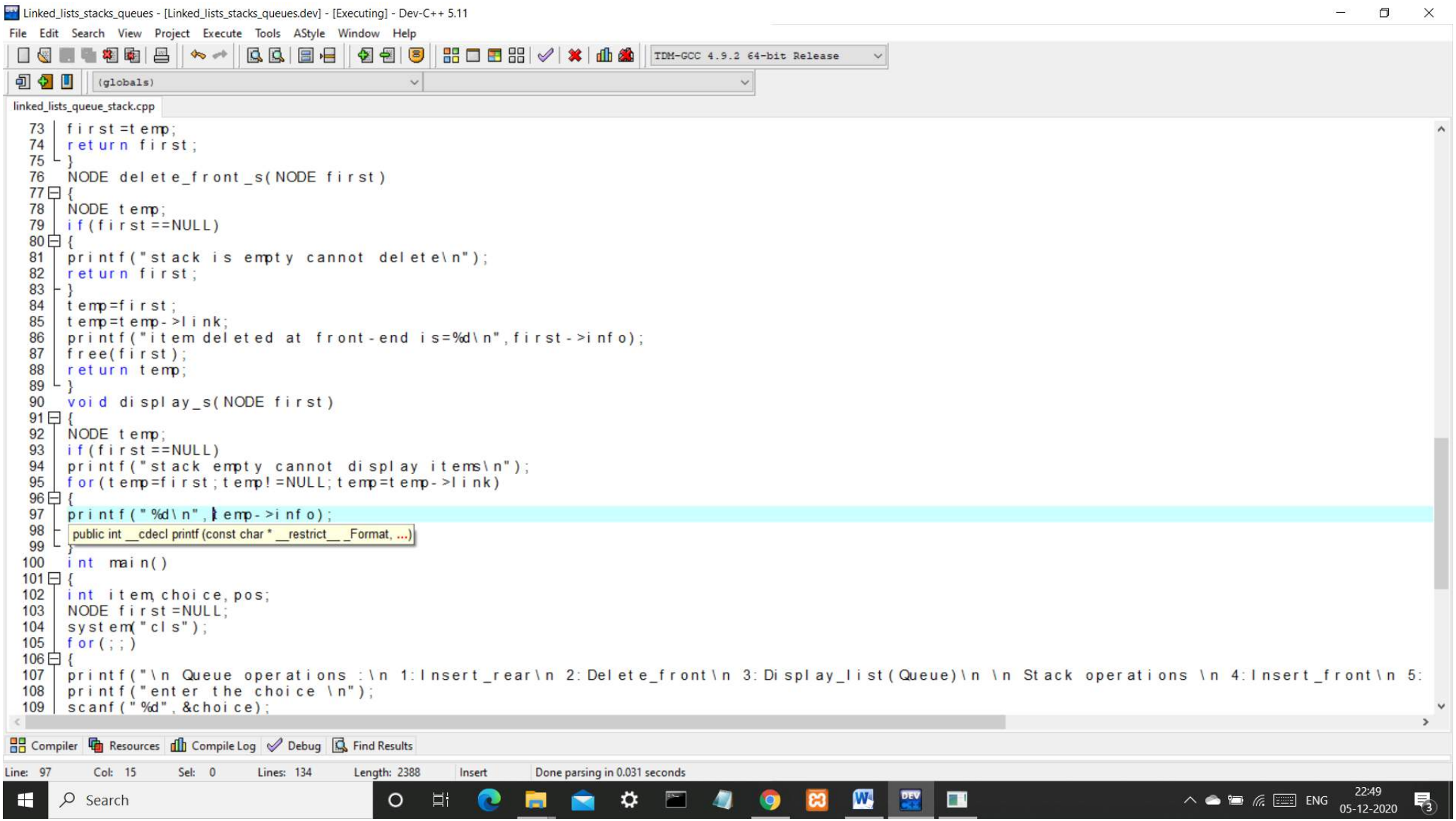
```
temp -> link = NULL;
if (first == NULL)
return temp;
cur = first;
while (cur -> link != NULL)
cur = cur -> link;
cur -> link = temp;
return first;
}
NODE delete_front (NODE first){
NODE temp;
if (first == NULL){
printf ("list is empty cannot delete \n");
return first;
}
temp = first;
temp = temp -> link;
printf ("item deleted at front-end is = %d \n", first -> info);
free (first);
return temp;
}
void display (NODE first) {
NODE temp;
if (first == NULL)
printf ("list empty cannot display items \n");
for (temp = first; temp != NULL; temp = temp -> link).
{
printf ("%d \n", temp -> info);
}}
NODE insert_front (NODE first, int item){
NODE temp;
temp = get_node();
temp -> info = item;
temp -> link = NULL;
```

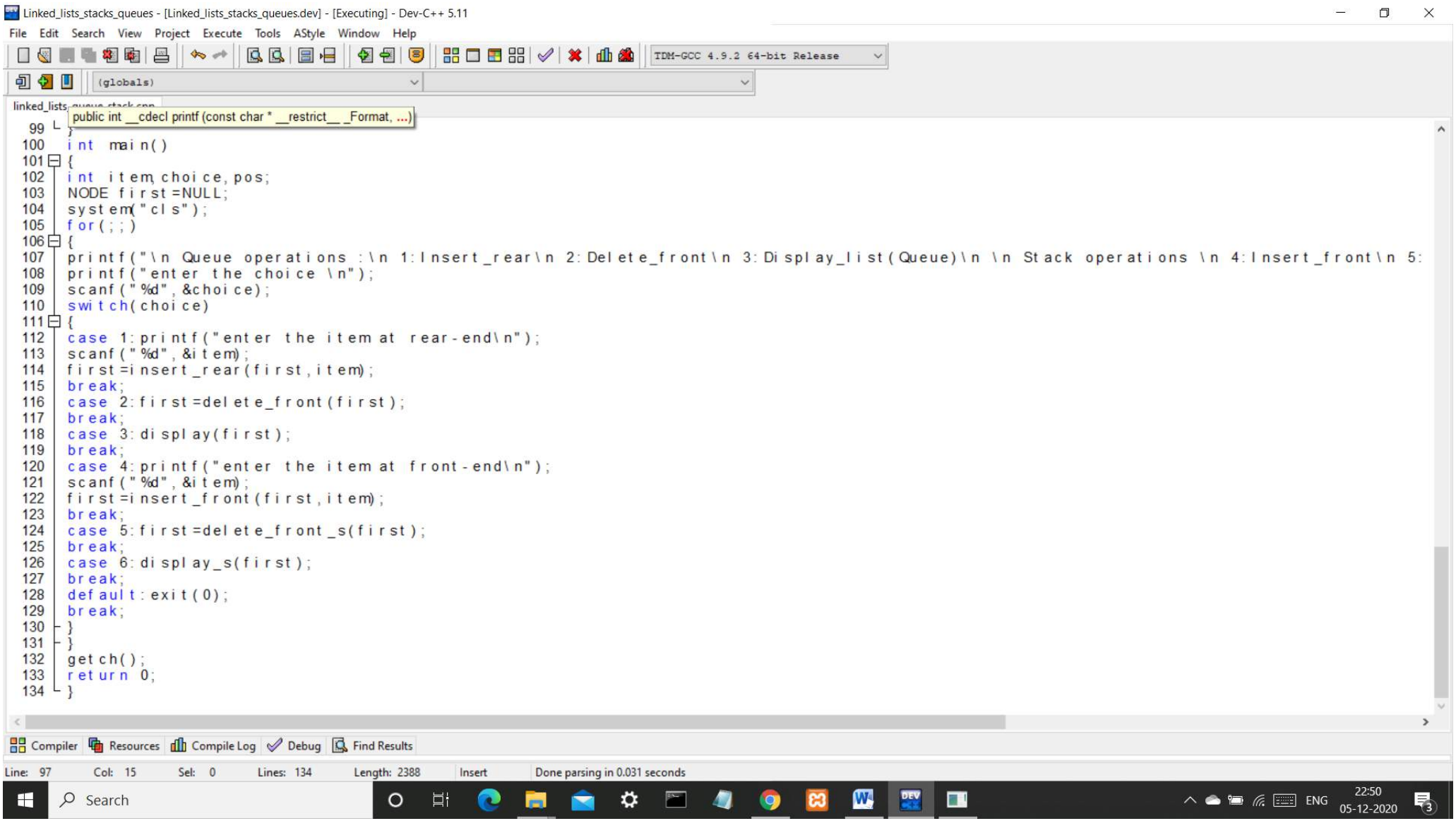
```
if (first == NULL)
return temp;
temp->link = first;
first = temp;
return first;
}
NODE delete_front_s(NODE first) {
NODE temp;
if (first == NULL) {
printf ("stack is empty cannot delete \n");
return first;
}
temp = first;
temp = temp -> link;
printf ("item deleted at front-end is = %d \n", first->info);
free (first);
return temp;
}
void display_s (NODE first) {
NODE temp;
if ( first == NULL)
printf ("stack empty cannot display items \n");
for(temp = first; temp != NULL; temp = temp->link)
{
  printf("%d \n", temp->info );
}
int main(){
int item, choice, pos;
NODE first = NULL;
system("cls");
for (;;){
printf("\n Queue operation : \n 1: insert rear \n 2: Delete_front
 \n 3: Display_list(Queue) \n \n Stack operations \n
4: Insert front \n 5: Delete_front \n 6: Display_list (Stack) \n
```

```
7: Exit \n");
printf("enter the choice \n");
scanf("%d",&choice);
switch(choice)
{
case 1: printf("enter the item at rear-end \n");
        scanf("%d",&item);
        first = insert_rear(first,item);
        break;
case 2: first = delete_front(first);
        break;
case 3 : display(first);
break;
case 4 : printf("enter the item at front-end \n");
        scanf("%d",&item);
        first = insert_front(first,item);
        break;
case 5: first = delete_front_s(first);
        break;
case 6: display_s(first);
        break;
default: exit(0);
break;
}}
getch();
return 0;
}
```


Linked_lists_stacks_queues - [Linked_lists_stacks_queues.dev] - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
linked_lists_queue_stack.cpp
#include<stdio.h>
#include<conio.h>
#include<malloc.h>
#include<process.h>
struct node
{
int info;
struct node *link;
};
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 insert_rear(NODE first,int item)
{
NODE temp,cur;
temp=getnode();
temp->info=item;
temp->link=NULL;
if(first==NULL)
return temp;
cur=first;
while(cur->link!=NULL)
cur=cur->link;
cur->link=temp;
Compiler
Resources
Compile Log
Debug
Find Results
Line: 97 Col: 15 Sel: 0 Lines: 134 Length: 2388 Insert Done parsing in 0.031 seconds
public int __cdecl printf (const char * __restrict__ _Format, ...)
ENG
22:49
05-12-2020


Linked_lists_stacks_queues - [Linked_lists_stacks_queues.dev] - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
linked_lists_queue_stack.cpp
cur=cur->link;
cur->link=temp;
return first;
}
NODE delete_front(NODE first)
{
NODE temp;
if(first==NULL)
{
printf("list is empty cannot delete\n");
return first;
}
temp=first;
temp=temp->link;
printf("item deleted at front-end is=%d\n",first->info);
free(first);
return temp;
}
void display(NODE first)
{
NODE temp;
if(first==NULL)
printf("list empty cannot display items\n");
for(temp=first;temp!=NULL;temp=temp->link)
{
printf("%d \n",temp->info);
}
}
NODE insert_front(NODE first,int item)
{
NODE temp;
temp=getnode();
temp->info=item;
temp->link=NULL;
if(first==NULL)
return temp;
temp->link=first;
Compiler Resources Compile Log Debug Find Results
Line: 97 Col: 15 Sel: 0 Lines: 134 Length: 2388 Insert Done parsing in 0.031 seconds
public int __cdecl printf (const char * __restrict__ _Format, ...)
ENG 22:49 05-12-2020


Linked_lists_stacks_queues - [Linked_lists_stacks_queues.dev] - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
linked_lists_queue_stack.cpp
73 first=temp;
74 return first;
75 }
76 NODE delete_front_s(NODE first)
77 {
78 NODE temp;
79 if(first==NULL)
80 {
81 printf("stack is empty cannot delete\n");
82 return first;
83 }
84 temp=first;
85 temp=temp->link;
86 printf("item deleted at front-end is=%d\n",first->info);
87 free(first);
88 return temp;
89 }
90 void display_s(NODE first)
91 {
92 NODE temp;
93 if(first==NULL)
94 printf("stack empty cannot display items\n");
95 for(temp=first;temp!=NULL;temp=temp->link)
96 {
97 printf("%d\n",temp->info);
98 public int __cdecl printf (const char * __restrict__ _Format, ...)
99 }
100 int main()
101 {
102 int item,choice,pos;
103 NODE first=NULL;
104 system("cls");
105 for(;;)
106 {
107 printf("\n Queue operations :\n 1:Insert_rear\n 2:Delete_front\n 3:Display_list(Queue)\n \n Stack operations \n 4:Insert_front\n 5:
108 printf("enter the choice \n");
109 scanf("%d",&choice);
Compiler Resources Compile Log Debug Find Results
Line: 97 Col: 15 Sel: 0 Lines: 134 Length: 2388 Insert Done parsing in 0.031 seconds
Search
ENG
22:49
05-12-2020


Linked_lists_stacks_queues - [Linked_lists_stacks_queues.dev] - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
linked_lists_queue_stack.cpp
public int __cdecl printf (const char * __restrict__ _Format, ...)
}
int main()
{
int item,choice,pos;
NODE first=NULL;
system("cls");
for(;;)
{
printf("\n Queue operations :\n 1:Insert_rear\n 2:Delete_front\n 3:Display_list(Queue)\n \n Stack operations \n 4:Insert_front\n 5:
printf("enter the choice \n");
scanf("%d",&choice);
switch(choice)
{
case 1:printf("enter the item at rear-end\n");
scanf("%d",&item);
first=insert_rear(first,item);
break;
case 2:first=delete_front(first);
break;
case 3:display(first);
break;
case 4:printf("enter the item at front-end\n");
scanf("%d",&item);
first=insert_front(first,item);
break;
case 5:first=delete_front_s(first);
break;
case 6:display_s(first);
break;
default:exit(0);
break;
}
}
getch();
return 0;
}
Compiler
Resources
Compile Log
Debug
Find Results
Line: 97 Col: 15 Sel: 0 Lines: 134 Length: 2388 Insert Done parsing in 0.031 seconds
Search
ENG
22:50
05-12-2020

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\linked_list_stack_queues\Linked_lists_stacks_queues.exe

```
 Queue operations :
 1:Insert_rear
 2:Delete_front
 3:Display_list(Queue)

 Stack operations
 4:Insert_front
 5: Delete_front
 6:Dislay_list(Stack)
 7:Exit

enter the choice
1
enter the item at rear-end
34

 Queue operations :
 1:Insert_rear
 2:Delete_front
 3:Display_list(Queue)

 Stack operations
 4:Insert_front
 5: Delete_front
 6:Dislay_list(Stack)
 7:Exit

enter the choice
1
enter the item at rear-end
54

 Queue operations :
 1:Insert_rear
 2:Delete_front
 3:Display_list(Queue)

 Stack operations
 4:Insert_front
 5: Delete_front
 6:Dislay_list(Stack)
 7:Exit

enter the choice
2
item deleted at front-end is=34

 Queue operations :
 1:Insert_rear
```

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\linked_list_stack_queues\Linked_lists_stacks_queues.exe

```
enter the choice
2
item deleted at front-end is=34

 Queue operations :
 1:Insert_rear
 2:Delete_front
 3:Display_list(Queue)

 Stack operations
 4:Insert_front
 5: Delete_front
 6:Dislay_list(Stack)
 7:Exit

enter the choice
3
54

 Queue operations :
 1:Insert_rear
 2:Delete_front
 3:Display_list(Queue)

 Stack operations
 4:Insert_front
 5: Delete_front
 6:Dislay_list(Stack)
 7:Exit

enter the choice
4
enter the item at front-end
6

 Queue operations :
 1:Insert_rear
 2:Delete_front
 3:Display_list(Queue)

 Stack operations
 4:Insert_front
 5: Delete_front
 6:Dislay_list(Stack)
 7:Exit

enter the choice
4
enter the item at front-end
7
```

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\linked_list_stack_queues\Linked_lists_stacks_queues.exe

```
 7:Exit

enter the choice
4
enter the item at front-end
7

 Queue operations :
 1:Insert_rear
 2:Delete_front
 3:Display_list(Queue)

 Stack operations
 4:Insert_front
 5: Delete_front
 6:Dislay_list(Stack)
 7:Exit

enter the choice
4
enter the item at front-end
8

 Queue operations :
 1:Insert_rear
 2:Delete_front
 3:Display_list(Queue)

 Stack operations
 4:Insert_front
 5: Delete_front
 6:Dislay_list(Stack)
 7:Exit

enter the choice
5
item deleted at front-end is=8

 Queue operations :
 1:Insert_rear
 2:Delete_front
 3:Display_list(Queue)

 Stack operations
 4:Insert_front
 5: Delete_front
 6:Dislay_list(Stack)
 7:Exit

enter the choice
```

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\linked_list_stack_queues\Linked_lists_stacks_queues.exe

```
 2:Delete_front
 3:Display_list(Queue)

 Stack operations
 4:Insert_front
 5: Delete_front
 6:Dislay_list(Stack)
 7:Exit

enter the choice
5
item deleted at front-end is=8

 Queue operations :
 1:Insert_rear
 2:Delete_front
 3:Display_list(Queue)

 Stack operations
 4:Insert_front
 5: Delete_front
 6:Dislay_list(Stack)
 7:Exit

enter the choice
6
7
6
54

 Queue operations :
 1:Insert_rear
 2:Delete_front
 3:Display_list(Queue)

 Stack operations
 4:Insert_front
 5: Delete_front
 6:Dislay_list(Stack)
 7:Exit

enter the choice
7

--------------------------------
Process exited after 78.14 seconds with return value 0
Press any key to continue . . .
```

1BM19CS159

Sohan R Kumar.

Lab Program - 9

Q) Write a program to implement doubly linked list with following operations:

i) Insert front
ii) Insert rear
iii) Delete front
iv) Delete rear
v) Display
vi) Insert after/before key node
vii) Search
viii) Delete duplicate.

A)

```
#include<stdio.h>
#include<conio.h>
#include<process.h>
#include<stdlib.h>
#include<malloc.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;
}
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", temp->info);
temp=temp->rlink;
}
printf("\n");
}
NODE delete_all_key(int item, NODE head){
NODE prev, cur, next;
int count;
if(head->rlink==head){
printf("List Empty \n");
return head;
}
```

```
count 0;
cur = head -> rlink;
cur = cur -> 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 \n");
else
printf("key found at %d positions and are deleted \n", count);
return head;
}
NODE insert_leftpos(int item, NODE head){
NODE temp, cur, prev;
if(head -> rlink == head){
printf("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 insert_right(int item, NODE head){
NODE temp, cur, prev;
if(head->llink == head){
printf("Empty \n");
return head;
}
cur = head->llink;
while(cur != head){
if(item == cur->info) break;
cur = cur->llink;
}
if(cur == head){
printf("key not found \n");
return head;
}
prev = cur->rlink;
printf("Enter towards right of %d:", item);
temp = getnode();
scanf("%d", &temp->info);
prev->llink = temp;
temp->rlink = prev;
cur->rlink = temp;
```

```
temp->llink = cur;
return head;
}
NODE search(NODE head){
NODE temp = head->rlink;
int count=0, key, flag=0;
printf("Enter the key : ");
scanf("%d", &key);
while(temp!=head){
count++;
if(temp->info==key){
flag=1;
printf("key %d found in position %d ", key, count);
}
temp = temp->rlink;
}
if(flag==0){
printf("key is not found in list");
}
return head;
}
int main(){
NODE head, last;
int item, choice, option;
head = getnode();
head->rlink = head;
head->llink = head;
system("cls");
for(;;){
printf("\n1: insert front \n2: insert rear \n3: delete front \n
4: delete rear \n5: display \n6: Delete repeating occurences \n7:
search \n8: Insert node after/befor key node \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 item \n");
        scanf ("%d", &item);
        last = delete_all_key (item, head);
        break;
case 8: printf (" press 1: for insert behind 2: for insert after ");
        scanf ("%d", &option);
        if (option == 1) {
        printf (" enter # key node \n");
        scanf ("%d", &item);
        last = insert_leftpos (item, head); }
        else if (option == 2) {
        printf (" enter key node \n");
        scanf ("%d", &item);
        last = insert_rightpos (item, head);
        break;
    }
 case 7: search (head);
         break;
 default: exit(0);
```

```
}}
getch();
return(0);
}
```

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\doubly linked list\doubly_linked_lists.cpp - Dev-C++ 5.11

File Edit Search View Project Execute Tools AStyle Window Help

TDM-GCC 4.9.2 64-bit Release

(globals)

doubly_linked_lists.cpp

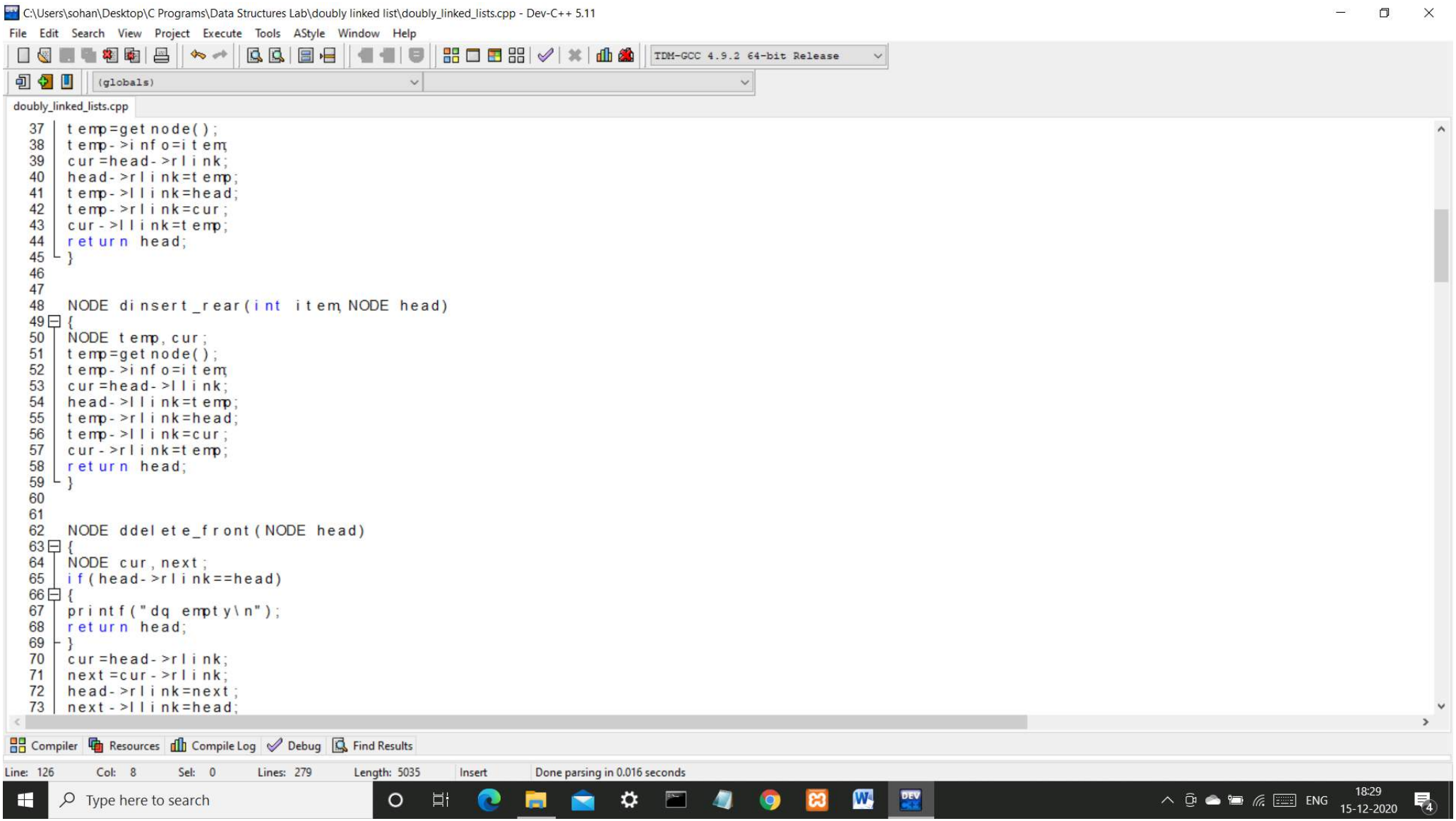
```
#include<stdio.h>
#include<conio.h>
#include<process.h>
#include<stdlib.h>
#include <malloc.h>
struct node
{
      int info;
      struct node *llink;
      struct node *rlink;
      };
typedef struct node *NODE;


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();
```

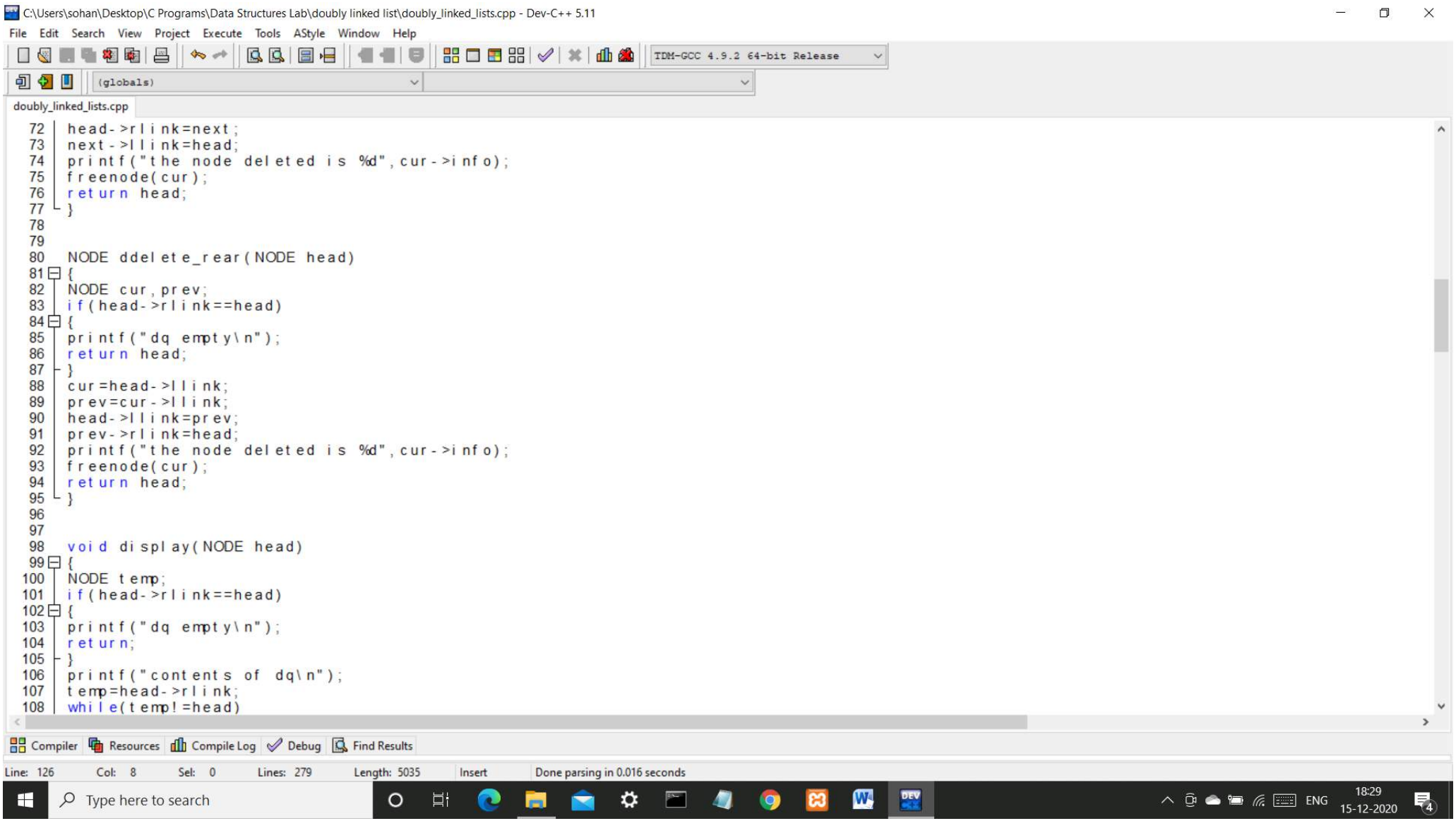
Compiler Resources Compile Log Debug Find Results

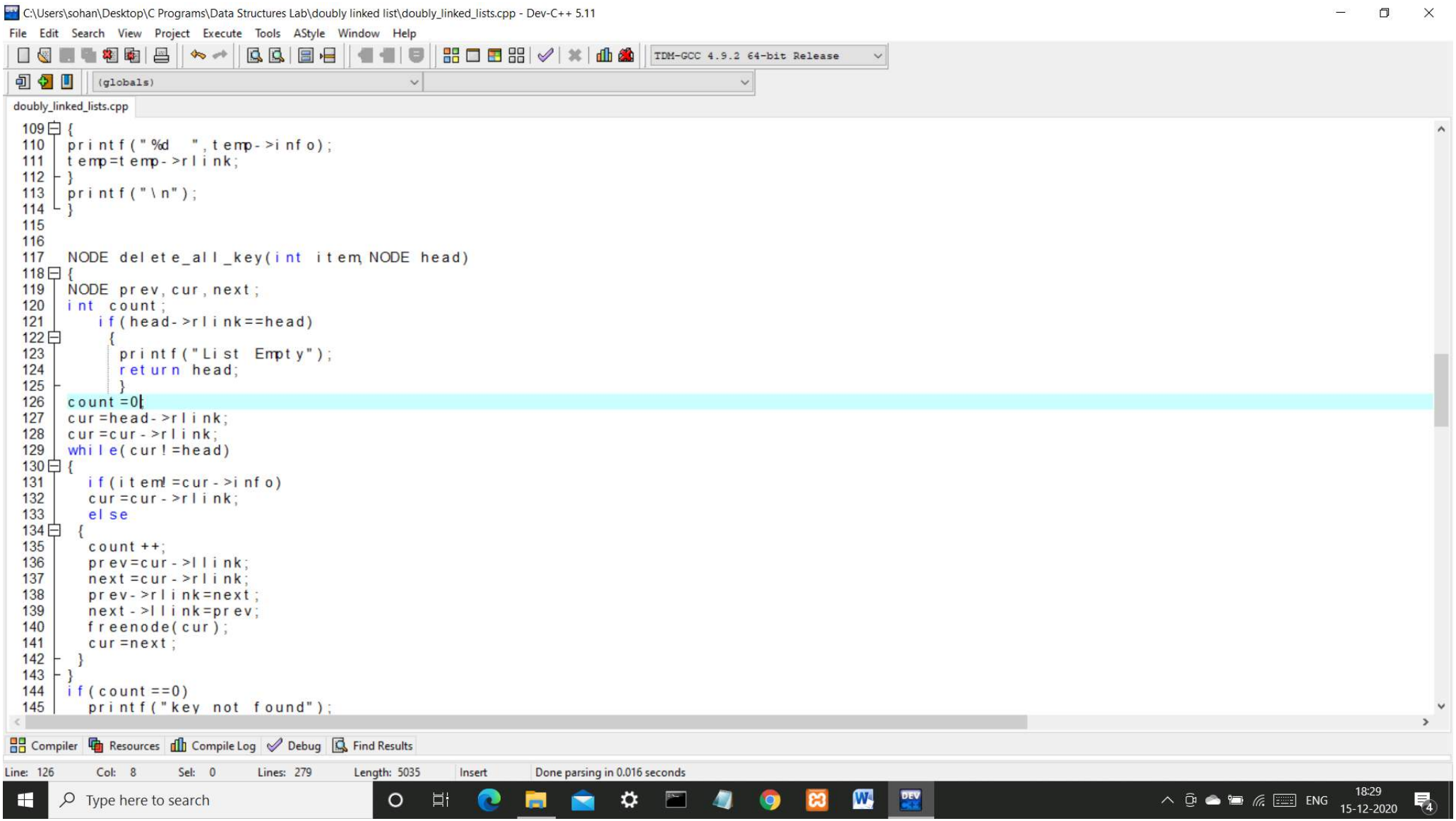
Line: 126 Col: 8 Sel: 0 Lines: 279 Length: 5035 Insert Done parsing in 0.016 seconds

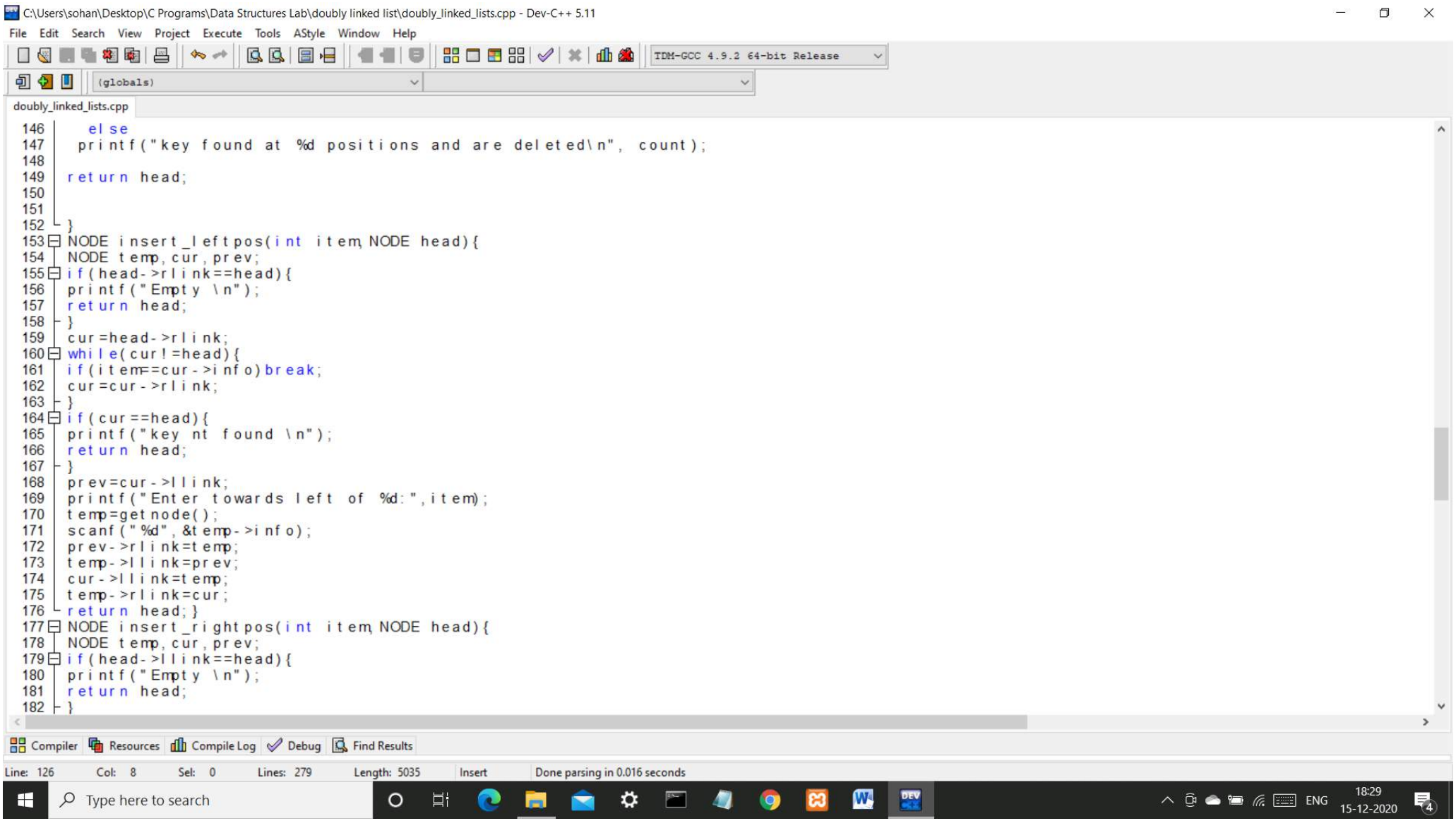
Type here to search

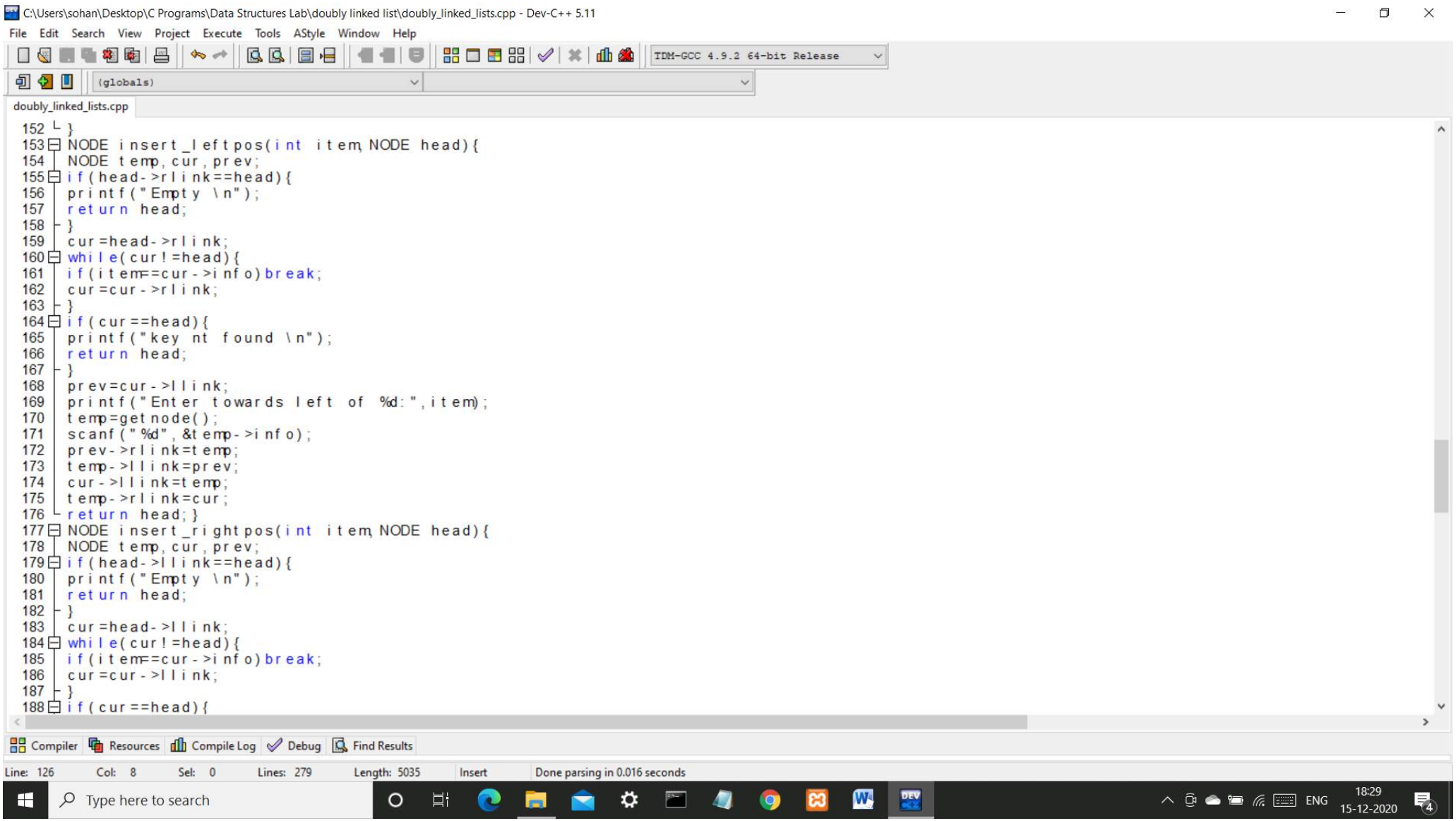
ENG 18:29 15-12-2020

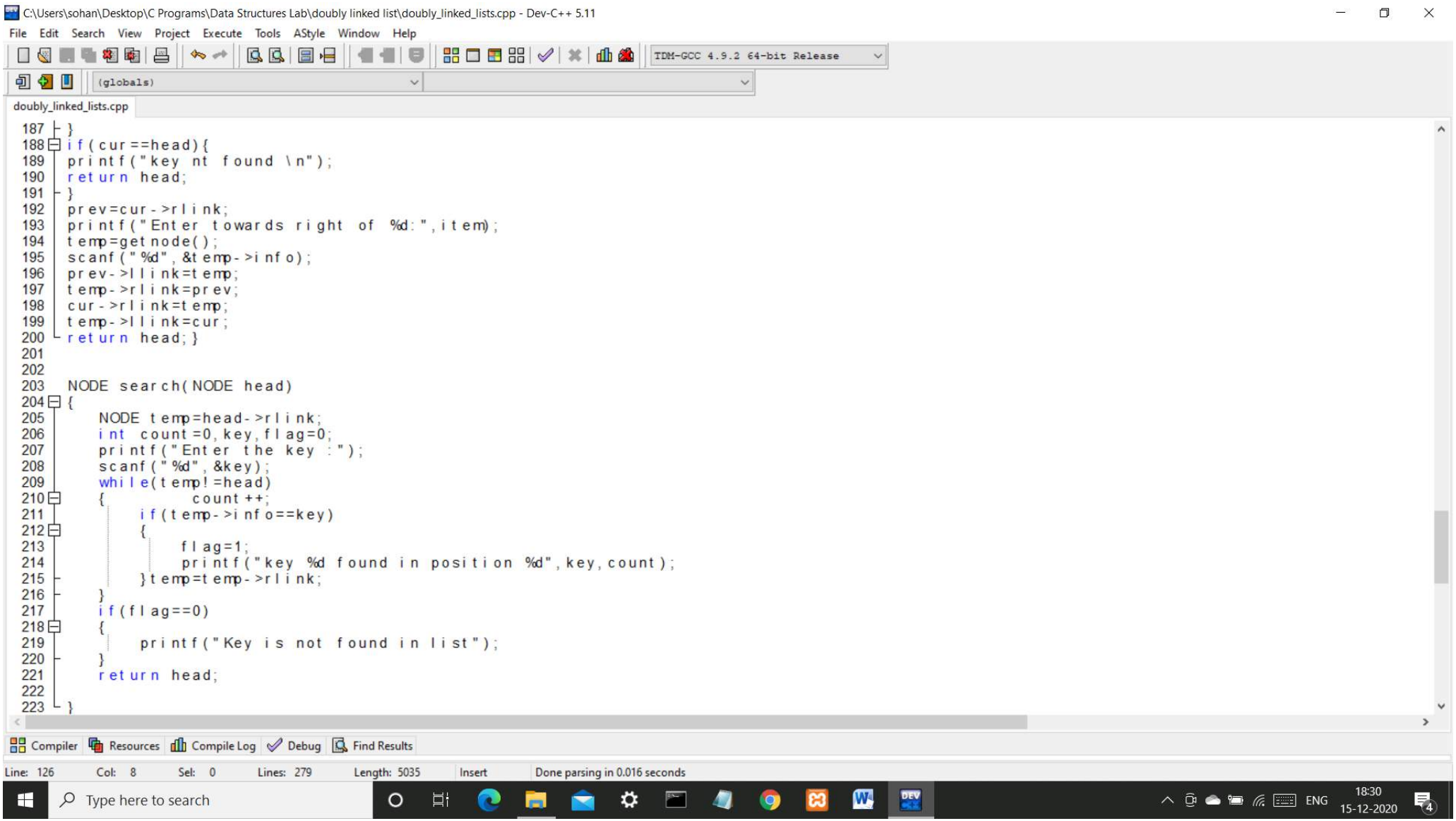

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\doubly linked list\doubly_linked_lists.cpp - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
doubly_linked_lists.cpp
temp=getnode();
temp->info=item;
cur=head->rlink;
head->rlink=temp;
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;
Compiler Resources Compile Log Debug Find Results
Line: 126 Col: 8 Sel: 0 Lines: 279 Length: 5035 Insert Done parsing in 0.016 seconds
Type here to search
ENG
18:29
15-12-2020

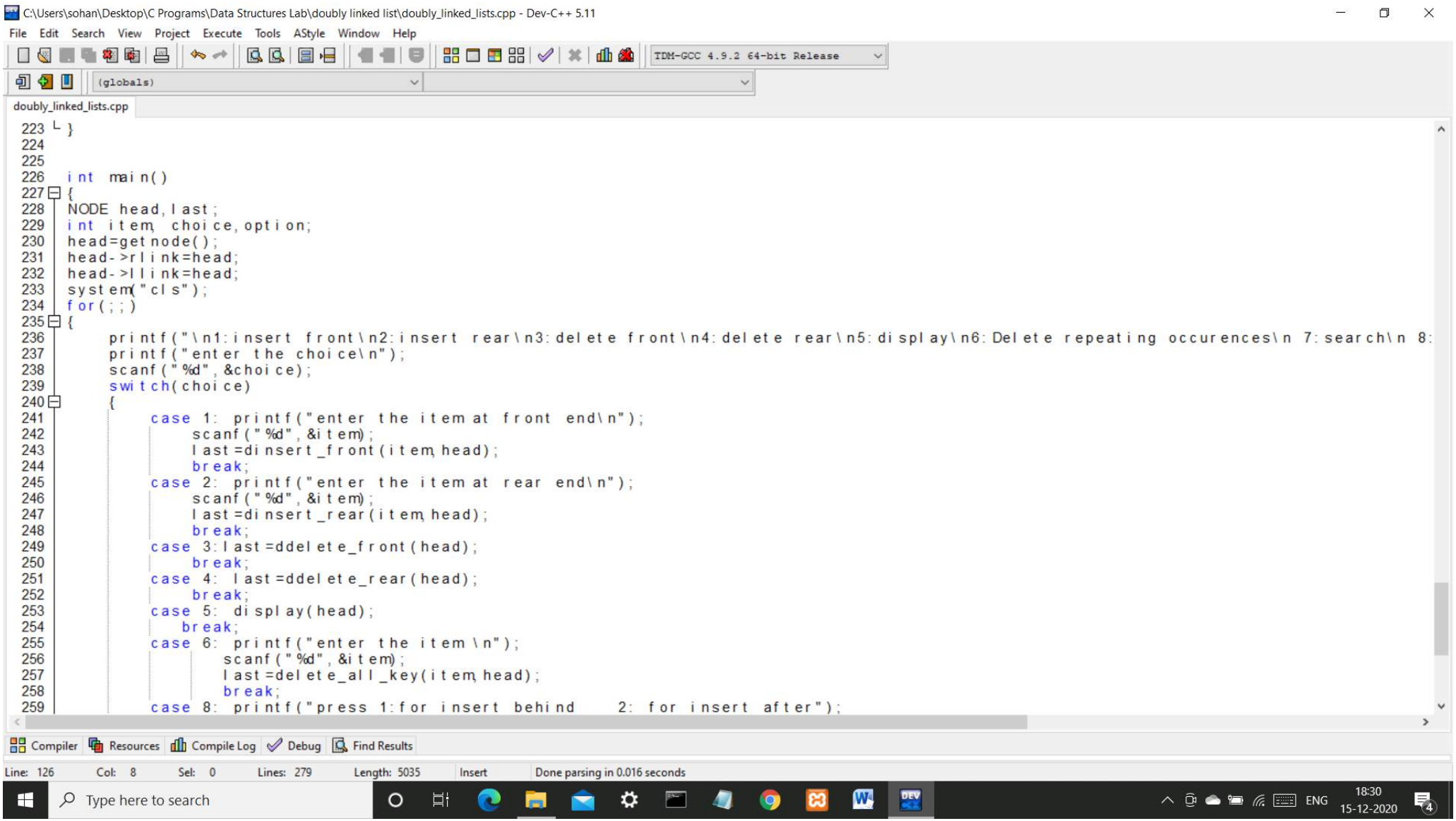

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\doubly linked list\doubly_linked_lists.cpp - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
doubly_linked_lists.cpp
head->rlink=next;
next->llink=head;
printf("the node deleted is %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;
}
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)
Compiler Resources Compile Log Debug Find Results
Line: 126 Col: 8 Sel: 0 Lines: 279 Length: 5035 Insert Done parsing in 0.016 seconds
Type here to search
ENG
18:29
15-12-2020

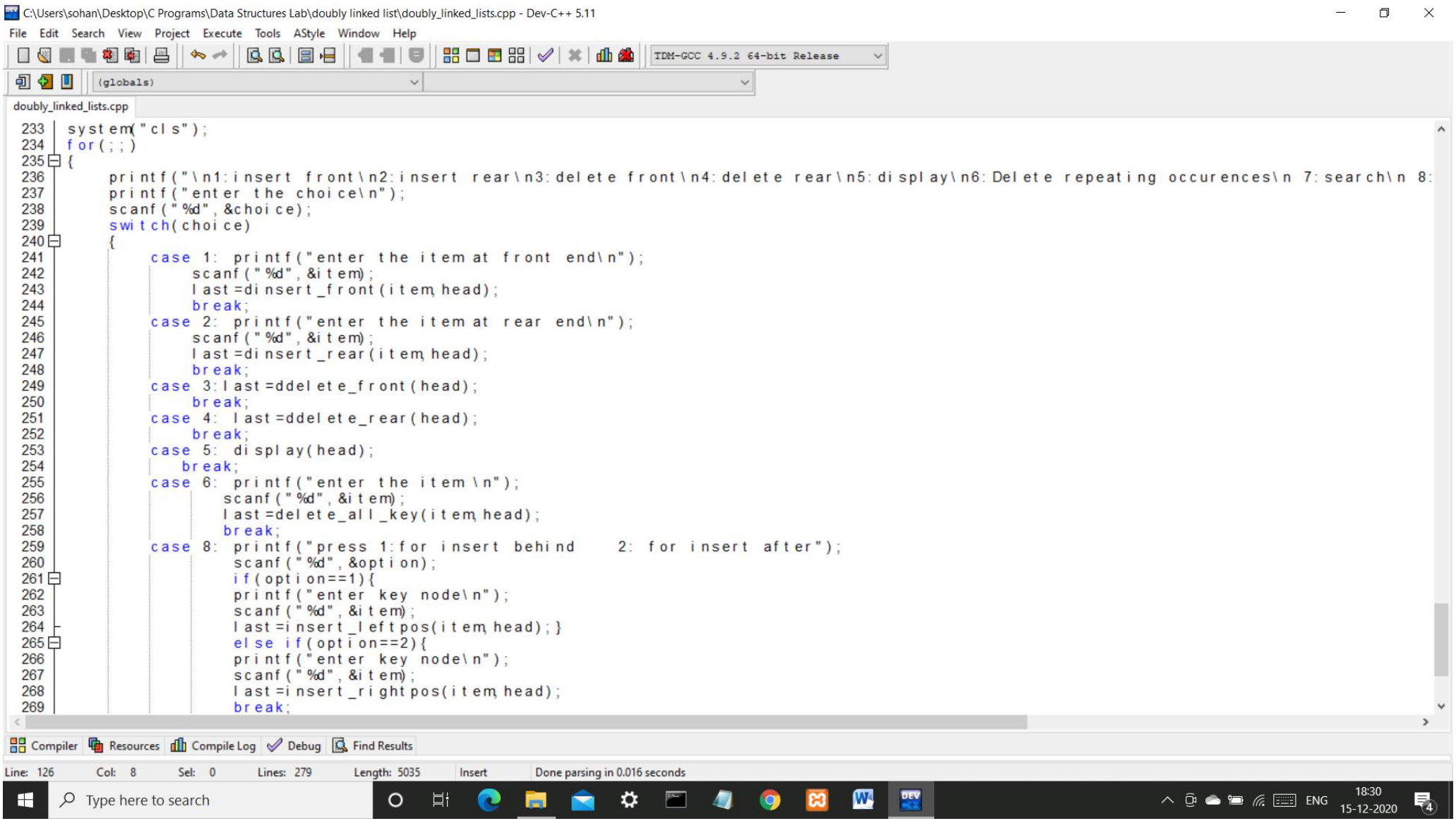

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\doubly linked list\doubly_linked_lists.cpp - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
doubly_linked_lists.cpp
{
printf("%d ",temp->info);
temp=temp->rlink;
}
printf("\n");
}
NODE delete_all_key(int item,NODE head)
{
NODE prev,cur,next;
int count;
if(head->rlink==head)
{
printf("List Empty");
return head;
}
count=0;
cur=head->rlink;
cur=cur->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");
Compiler Resources Compile Log Debug Find Results
Line: 126 Col: 8 Sel: 0 Lines: 279 Length: 5035 Insert Done parsing in 0.016 seconds
Type here to search
ENG
18:29
15-12-2020

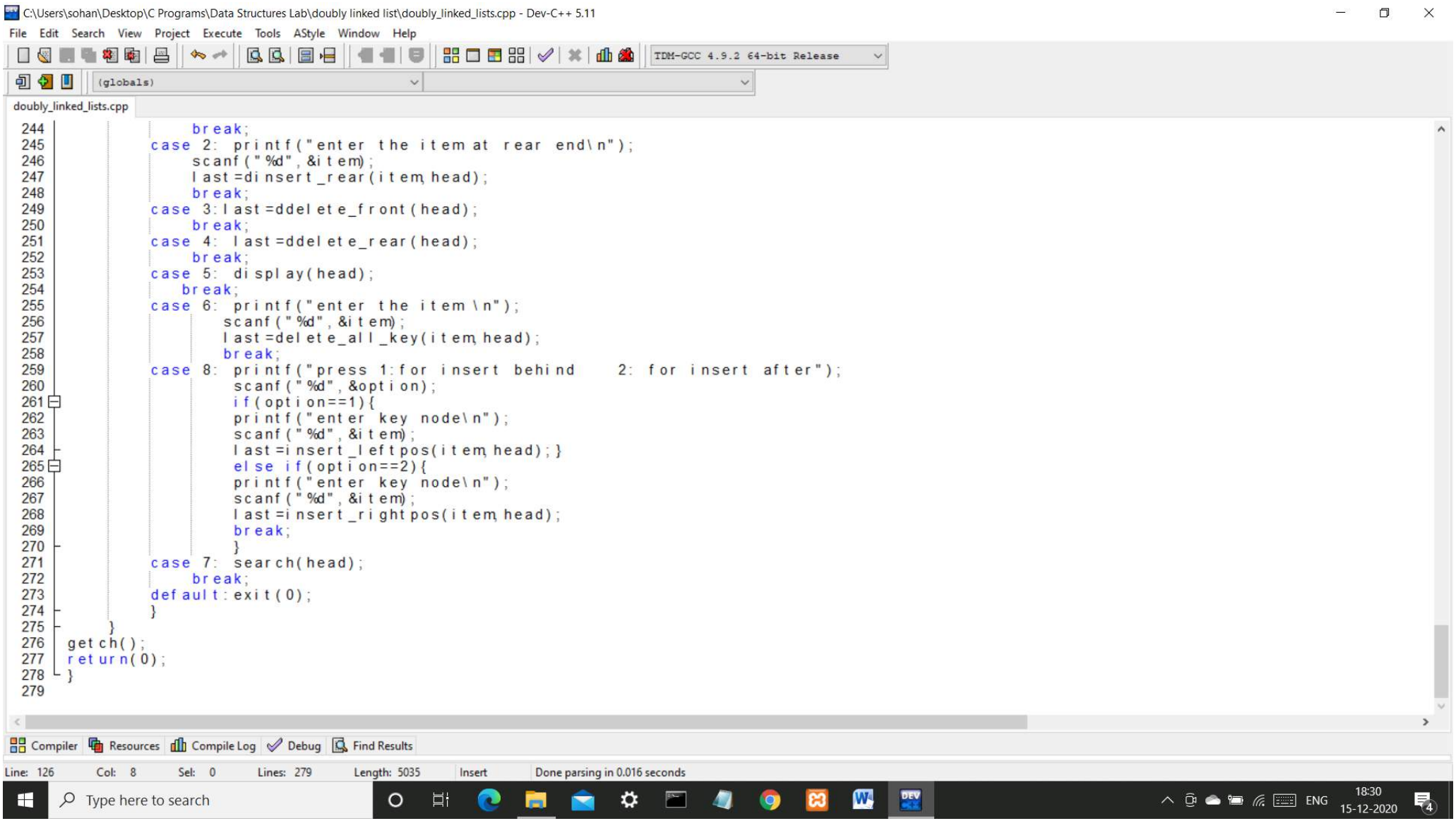

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\doubly linked list\doubly_linked_lists.cpp - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
doubly_linked_lists.cpp
146 else
147 printf("key found at %d positions and are deleted\n", count);
148
149 return head;
150
151
152 }
153 NODE insert_leftpos(int item,NODE head){
154 NODE temp,cur,prev;
155 if(head->rlink==head){
156 printf("Empty \n");
157 return head;
158 }
159 cur=head->rlink;
160 while(cur!=head){
161 if(item==cur->info)break;
162 cur=cur->rlink;
163 }
164 if(cur==head){
165 printf("key nt found \n");
166 return head;
167 }
168 prev=cur->llink;
169 printf("Enter towards left of %d:",item);
170 temp=getnode();
171 scanf("%d",&temp->info);
172 prev->rlink=temp;
173 temp->llink=prev;
174 cur->llink=temp;
175 temp->rlink=cur;
176 return head;}
177 NODE insert_rightpos(int item,NODE head){
178 NODE temp,cur,prev;
179 if(head->llink==head){
180 printf("Empty \n");
181 return head;
182 }
Compiler Resources Compile Log Debug Find Results
Line: 126 Col: 8 Sel: 0 Lines: 279 Length: 5035 Insert Done parsing in 0.016 seconds
Type here to search
ENG
18:29
15-12-2020

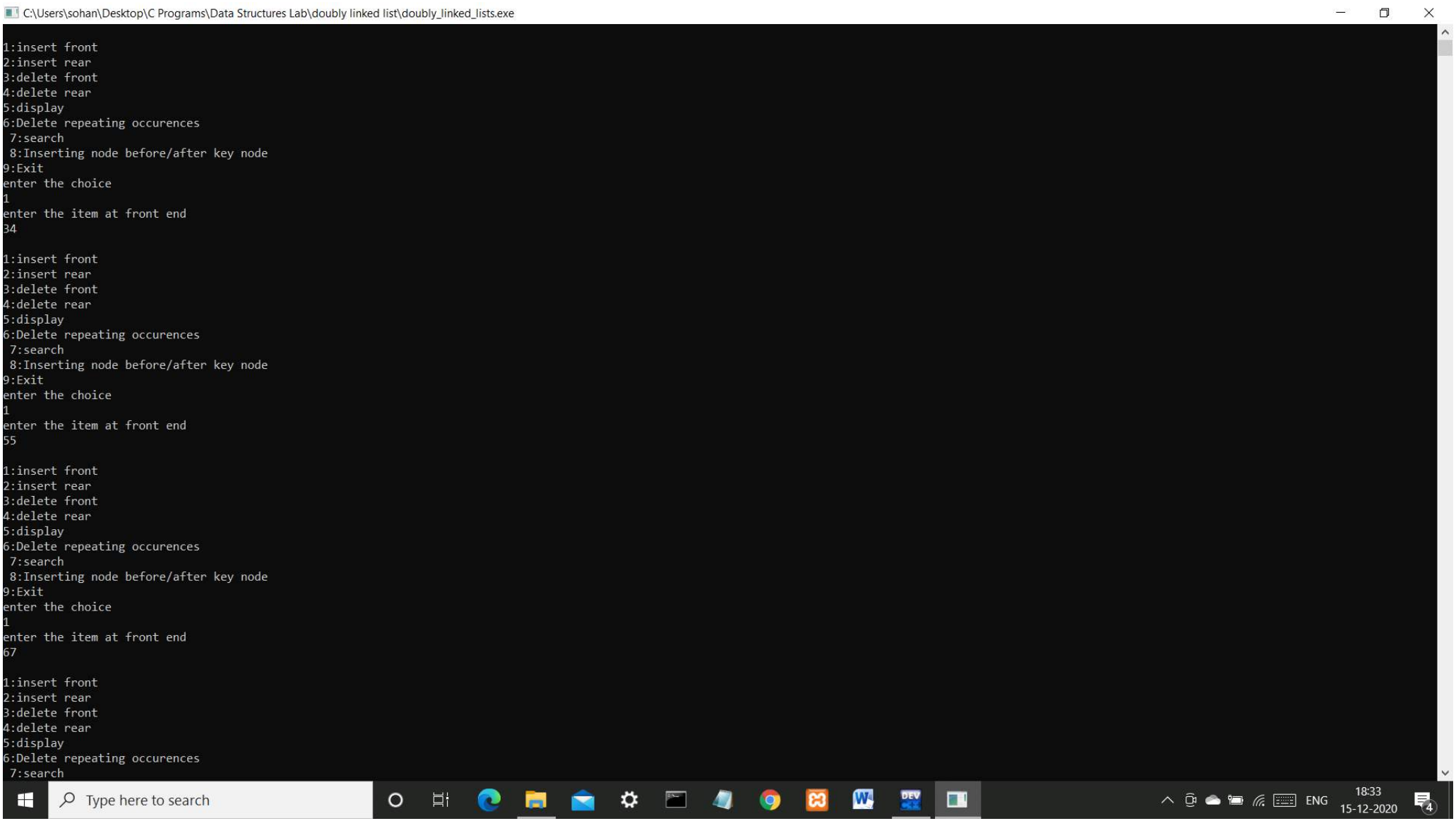

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\doubly linked list\doubly_linked_lists.cpp - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
doubly_linked_lists.cpp
152 }
153 NODE insert_leftpos(int item,NODE head){
154 NODE temp,cur,prev;
155 if(head->rlink==head){
156 printf("Empty \n");
157 return head;
158 }
159 cur=head->rlink;
160 while(cur!=head){
161 if(item==cur->info)break;
162 cur=cur->rlink;
163 }
164 if(cur==head){
165 printf("key nt found \n");
166 return head;
167 }
168 prev=cur->llink;
169 printf("Enter towards left of %d:",item);
170 temp=getnode();
171 scanf("%d",&temp->info);
172 prev->rlink=temp;
173 temp->llink=prev;
174 cur->llink=temp;
175 temp->rlink=cur;
176 return head;}
177 NODE insert_rightpos(int item,NODE head){
178 NODE temp,cur,prev;
179 if(head->llink==head){
180 printf("Empty \n");
181 return head;
182 }
183 cur=head->llink;
184 while(cur!=head){
185 if(item==cur->info)break;
186 cur=cur->llink;
187 }
188 if(cur==head){
Compiler Resources Compile Log Debug Find Results
Line: 126 Col: 8 Sel: 0 Lines: 279 Length: 5035 Insert Done parsing in 0.016 seconds
Type here to search
ENG
18:29
15-12-2020

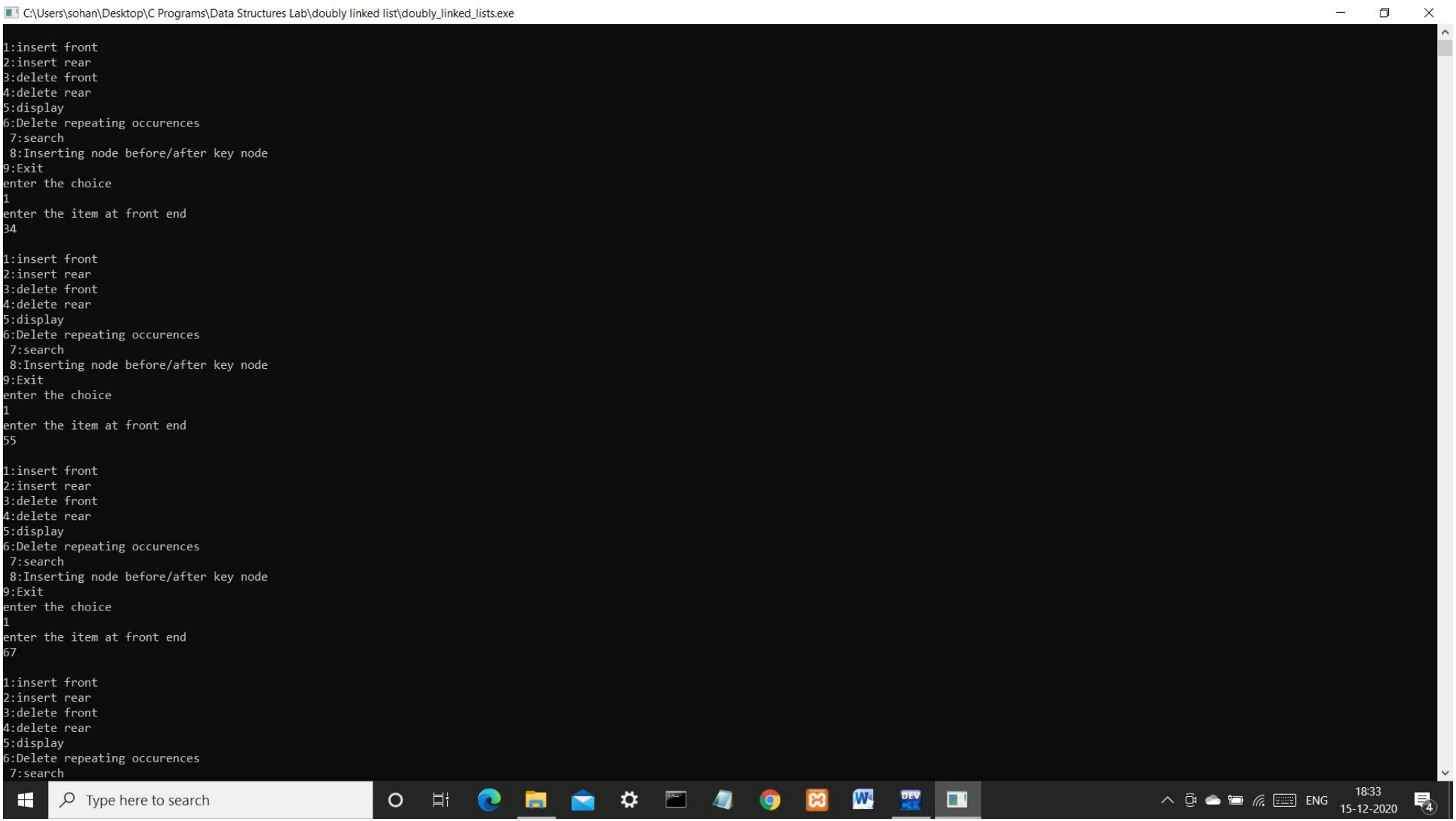

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\doubly linked list\doubly_linked_lists.cpp - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
doubly_linked_lists.cpp
}
if(cur==head){
printf("key nt found \n");
return head;
}
prev=cur->rlink;
printf("Enter towards right of %d:",item);
temp=getnode();
scanf("%d",&temp->info);
prev->llink=temp;
temp->rlink=prev;
cur->rlink=temp;
temp->llink=cur;
return head;}
NODE search(NODE head)
{
NODE temp=head->rlink;
int count=0,key,flag=0;
printf("Enter the key :");
scanf("%d",&key);
while(temp!=head)
{ count++;
if(temp->info==key)
{
flag=1;
printf("key %d found in position %d",key,count);
}temp=temp->rlink;
}
if(flag==0)
{
printf("Key is not found in list");
}
return head;
}
Compiler Resources Compile Log Debug Find Results
Line: 126 Col: 8 Sel: 0 Lines: 279 Length: 5035 Insert Done parsing in 0.016 seconds
Type here to search
ENG
18:30
15-12-2020


C:\Users\sohan\Desktop\C Programs\Data Structures Lab\doubly linked list\doubly_linked_lists.cpp - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
doubly_linked_lists.cpp
223 }
224
225
226 int main()
227 {
228 NODE head,last;
229 int item, choice,option;
230 head=getnode();
231 head->rlink=head;
232 head->llink=head;
233 system("cls");
234 for(;;)
235 {
236 printf("\n1:insert front\n2:insert rear\n3:delete front\n4:delete rear\n5:display\n6:Delete repeating occurences\n 7:search\n 8:
237 printf("enter the choice\n");
238 scanf("%d",&choice);
239 switch(choice)
240 {
241 case 1: printf("enter the item at front end\n");
242 scanf("%d",&item);
243 last=dinsert_front(item,head);
244 break;
245 case 2: printf("enter the item at rear end\n");
246 scanf("%d",&item);
247 last=dinsert_rear(item,head);
248 break;
249 case 3:last=ddelete_front(head);
250 break;
251 case 4: last=ddelete_rear(head);
252 break;
253 case 5: display(head);
254 break;
255 case 6: printf("enter the item\n");
256 scanf("%d",&item);
257 last=delete_all_key(item,head);
258 break;
259 case 8: printf("press 1:for insert behind 2: for insert after");
Compiler Resources Compile Log Debug Find Results
Line: 126 Col: 8 Sel: 0 Lines: 279 Length: 5035 Insert Done parsing in 0.016 seconds
Type here to search
ENG
18:30
15-12-2020


C:\Users\sohan\Desktop\C Programs\Data Structures Lab\doubly linked list\doubly_linked_lists.cpp - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
doubly_linked_lists.cpp
233 system("cls");
234 for(;;)
235 {
236 printf("\n1:insert front\n2:insert rear\n3:delete front\n4:delete rear\n5:display\n6:Delete repeating occurences\n 7:search\n 8:
237 printf("enter the choice\n");
238 scanf("%d",&choice);
239 switch(choice)
240 {
241 case 1: printf("enter the item at front end\n");
242 scanf("%d",&item);
243 last=dinsert_front(item,head);
244 break;
245 case 2: printf("enter the item at rear end\n");
246 scanf("%d",&item);
247 last=dinsert_rear(item,head);
248 break;
249 case 3:last=ddelete_front(head);
250 break;
251 case 4: last=ddelete_rear(head);
252 break;
253 case 5: display(head);
254 break;
255 case 6: printf("enter the item\n");
256 scanf("%d",&item);
257 last=delete_all_key(item,head);
258 break;
259 case 8: printf("press 1:for insert behind 2: for insert after");
260 scanf("%d",&option);
261 if(option==1){
262 printf("enter key node\n");
263 scanf("%d",&item);
264 last=insert_leftpos(item,head);}
265 else if(option==2){
266 printf("enter key node\n");
267 scanf("%d",&item);
268 last=insert_rightpos(item,head);
269 break;
Compiler Resources Compile Log Debug Find Results
Line: 126 Col: 8 Sel: 0 Lines: 279 Length: 5035 Insert Done parsing in 0.016 seconds
Type here to search
ENG
18:30
15-12-2020


C:\Users\sohan\Desktop\C Programs\Data Structures Lab\doubly linked list\doubly_linked_lists.cpp - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
doubly_linked_lists.cpp
244 break;
245 case 2: printf("enter the item at rear end\n");
246 scanf("%d",&item);
247 last=dinsert_rear(item,head);
248 break;
249 case 3:last=ddelete_front(head);
250 break;
251 case 4: last=ddelete_rear(head);
252 break;
253 case 5: display(head);
254 break;
255 case 6: printf("enter the item\n");
256 scanf("%d",&item);
257 last=delete_all_key(item,head);
258 break;
259 case 8: printf("press 1:for insert behind 2: for insert after");
260 scanf("%d",&option);
261 if(option==1){
262 printf("enter key node\n");
263 scanf("%d",&item);
264 last=insert_leftpos(item,head);}
265 else if(option==2){
266 printf("enter key node\n");
267 scanf("%d",&item);
268 last=insert_rightpos(item,head);
269 break;
270 }
271 case 7: search(head);
272 break;
273 default:exit(0);
274 }
275 }
276 getch();
277 return(0);
278 }
279
Compiler Resources Compile Log Debug Find Results
Line: 126 Col: 8 Sel: 0 Lines: 279 Length: 5035 Insert Done parsing in 0.016 seconds
Type here to search
ENG
18:30
15-12-2020


C:\Users\sohan\Desktop\C Programs\Data Structures Lab\doubly linked list\doubly_linked_lists.exe
1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:Delete repeating occurences
7:search
8:Inserting node before/after key node
9:Exit
enter the choice
1
enter the item at front end
34
1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:Delete repeating occurences
7:search
8:Inserting node before/after key node
9:Exit
enter the choice
1
enter the item at front end
55
1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:Delete repeating occurences
7:search
8:Inserting node before/after key node
9:Exit
enter the choice
1
enter the item at front end
67
1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:Delete repeating occurences
7:search
Type here to search
ENG
18:33
15-12-2020


C:\Users\sohan\Desktop\C Programs\Data Structures Lab\doubly linked list\doubly_linked_lists.exe
1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:Delete repeating occurences
7:search
8:Inserting node before/after key node
9:Exit
enter the choice
1
enter the item at front end
34
1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:Delete repeating occurences
7:search
8:Inserting node before/after key node
9:Exit
enter the choice
1
enter the item at front end
55
1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:Delete repeating occurences
7:search
8:Inserting node before/after key node
9:Exit
enter the choice
1
enter the item at front end
67
1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:Delete repeating occurences
7:search
Type here to search
ENG
18:33
15-12-2020

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\doubly linked list\doubly_linked_lists.exe

```
5:display
6:Delete repeating occurences
 7:search
 8:Inserting node before/after key node
9:Exit
enter the choice
2
enter the item at rear end
9

1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:Delete repeating occurences
 7:search
 8:Inserting node before/after key node
9:Exit
enter the choice
2
enter the item at rear end
13

1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:Delete repeating occurences
 7:search
 8:Inserting node before/after key node
9:Exit
enter the choice
5
contents of dq
67  55  34  9  13

1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:Delete repeating occurences
 7:search
 8:Inserting node before/after key node
9:Exit
enter the choice
7
Enter the key :55
```

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\doubly linked list\doubly_linked_lists.exe

```
7
Enter the key :55
key 55 found in position 2
1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:Delete repeating occurences
 7:search
 8:Inserting node before/after key node
9:Exit
enter the choice
2
enter the item at rear end
33

1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:Delete repeating occurences
 7:search
 8:Inserting node before/after key node
9:Exit
enter the choice
1
enter the item at front end
1

1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:Delete repeating occurences
 7:search
 8:Inserting node before/after key node
9:Exit
enter the choice
5
contents of dq
1  67  55  34  9  13  33

1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
```

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\doubly linked list\doubly_linked_lists.exe

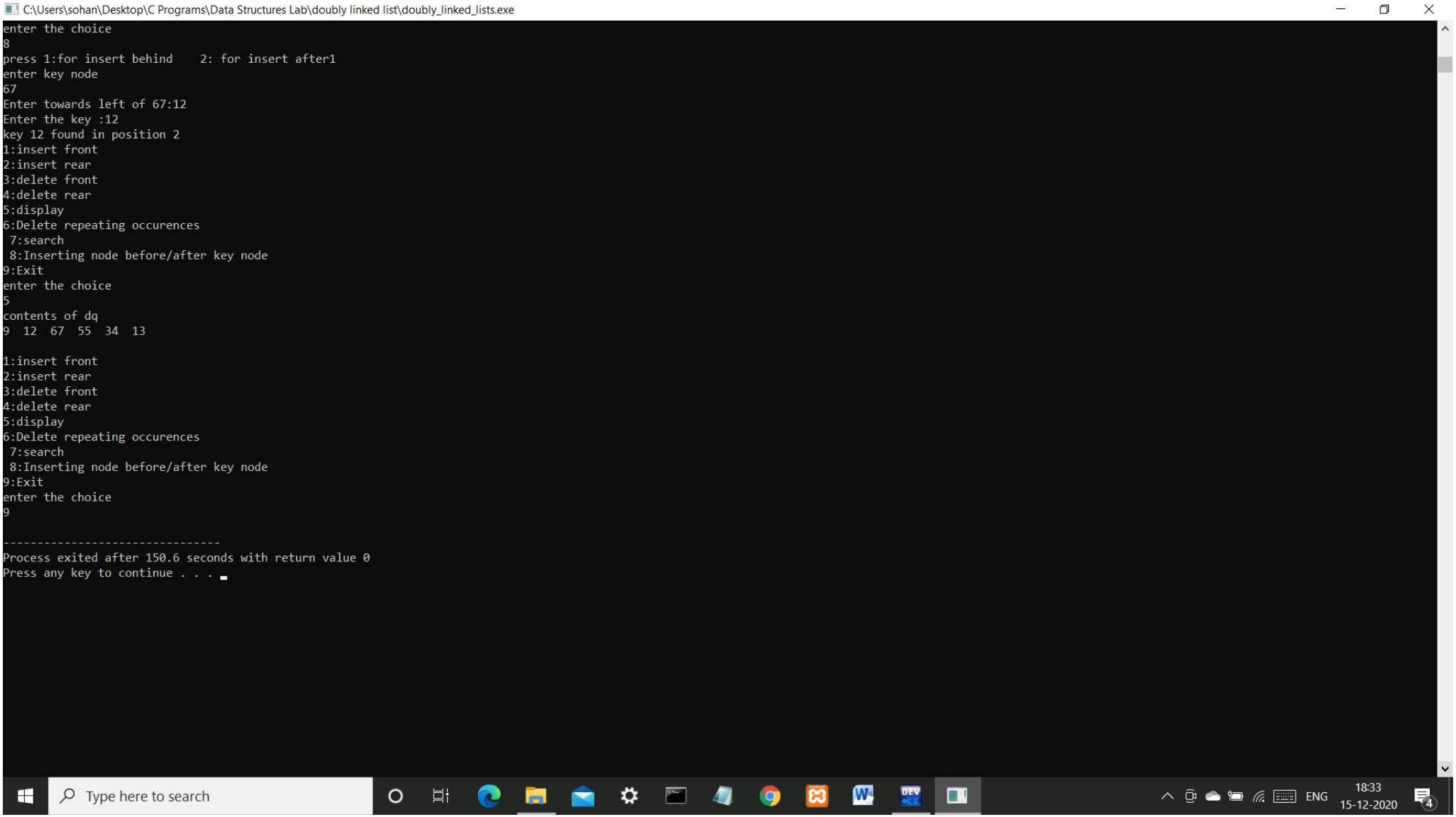
```
2:insert rear
3:delete front
4:delete rear
5:display
6:Delete repeating occurences
 7:search
 8:Inserting node before/after key node
9:Exit
enter the choice
3
the node deleted is 1
1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:Delete repeating occurences
 7:search
 8:Inserting node before/after key node
9:Exit
enter the choice
4
the node deleted is 33
1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:Delete repeating occurences
 7:search
 8:Inserting node before/after key node
9:Exit
enter the choice
5
contents of dq
67  55  34  9  13

1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:Delete repeating occurences
 7:search
 8:Inserting node before/after key node
9:Exit
enter the choice
2
enter the item at rear end
9
```

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\doubly linked list\doubly_linked_lists.exe

```
4:delete rear
5:display
6:Delete repeating occurences
 7:search
 8:Inserting node before/after key node
9:Exit
enter the choice
2
enter the item at rear end
9

1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:Delete repeating occurences
 7:search
 8:Inserting node before/after key node
9:Exit
enter the choice
1
enter the item at front end
9

1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:Delete repeating occurences
 7:search
 8:Inserting node before/after key node
9:Exit
enter the choice
5
contents of dq
9  67  55  34  9  13  9

1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:Delete repeating occurences
 7:search
 8:Inserting node before/after key node
9:Exit
enter the choice
6
```

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\doubly linked list\doubly_linked_lists.exe

```
9:Exit
enter the choice
6
enter the item
9
key found at 2 positions and are deleted

1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:Delete repeating occurences
 7:search
 8:Inserting node before/after key node
9:Exit
enter the choice
5
contents of dq
9  67  55  34  13

1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:Delete repeating occurences
 7:search
 8:Inserting node before/after key node
9:Exit
enter the choice
8
press 1:for insert behind    2: for insert after1
enter key node
67
Enter towards left of 67:12
Enter the key :12
key 12 found in position 2
1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:Delete repeating occurences
 7:search
 8:Inserting node before/after key node
9:Exit
enter the choice
5
contents of dq
```


C:\Users\sohan\Desktop\C Programs\Data Structures Lab\doubly linked list\doubly_linked_lists.exe
enter the choice
8
press 1:for insert behind 2: for insert after1
enter key node
67
Enter towards left of 67:12
Enter the key :12
key 12 found in position 2
1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:Delete repeating occurences
7:search
8:Inserting node before/after key node
9:Exit
enter the choice
5
contents of dq
9 12 67 55 34 13
1:insert front
2:insert rear
3:delete front
4:delete rear
5:display
6:Delete repeating occurences
7:search
8:Inserting node before/after key node
9:Exit
enter the choice
9
Process exited after 150.6 seconds with return value 0
Press any key to continue . . .
Type here to search
ENG
18:33
15-12-2020

Lab Program - 10

10) Write a program
a) To construct a binary search tree.
b) To traverse the tree using all methods i.e, in-order, preorder & post-order.
c) To display the elements in the tree.

A.

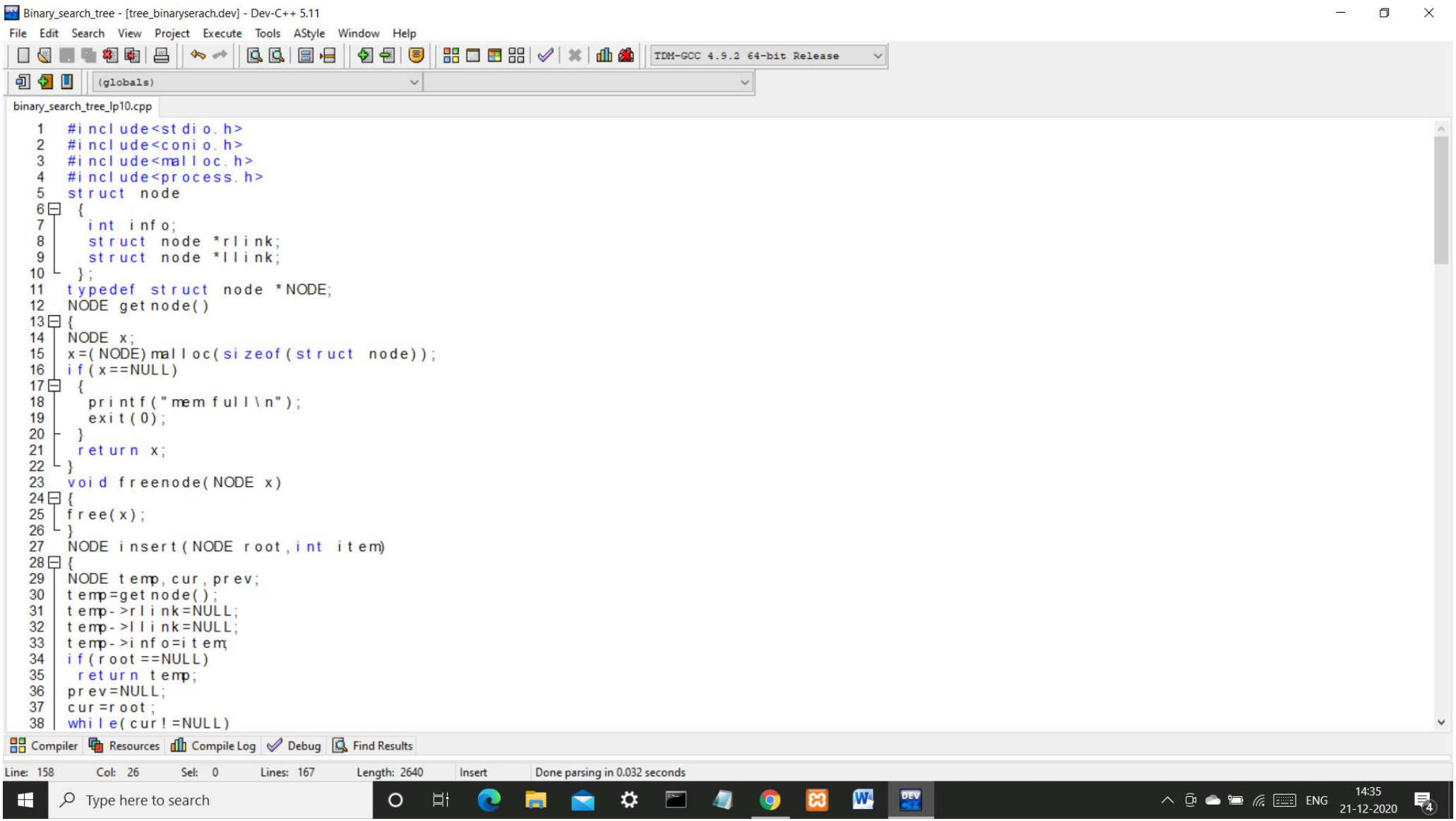
```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
struct node{
int info;
struct node *llink;
struct node *rlink;
};
typedef struct node *NODE;
NODE getnode(){
NODE x;
x=(NODE)malloc(sizeof(struct node));
if(x==NULL){
printf("Memory not available\n");
exit(0);
}
return x;
}
void freenode(NODE x){
free(x);
}
NODE insert(int item, NODE root){
NODE temp, cur, prev;
char direction[10];
int i;
temp=getnode();
```

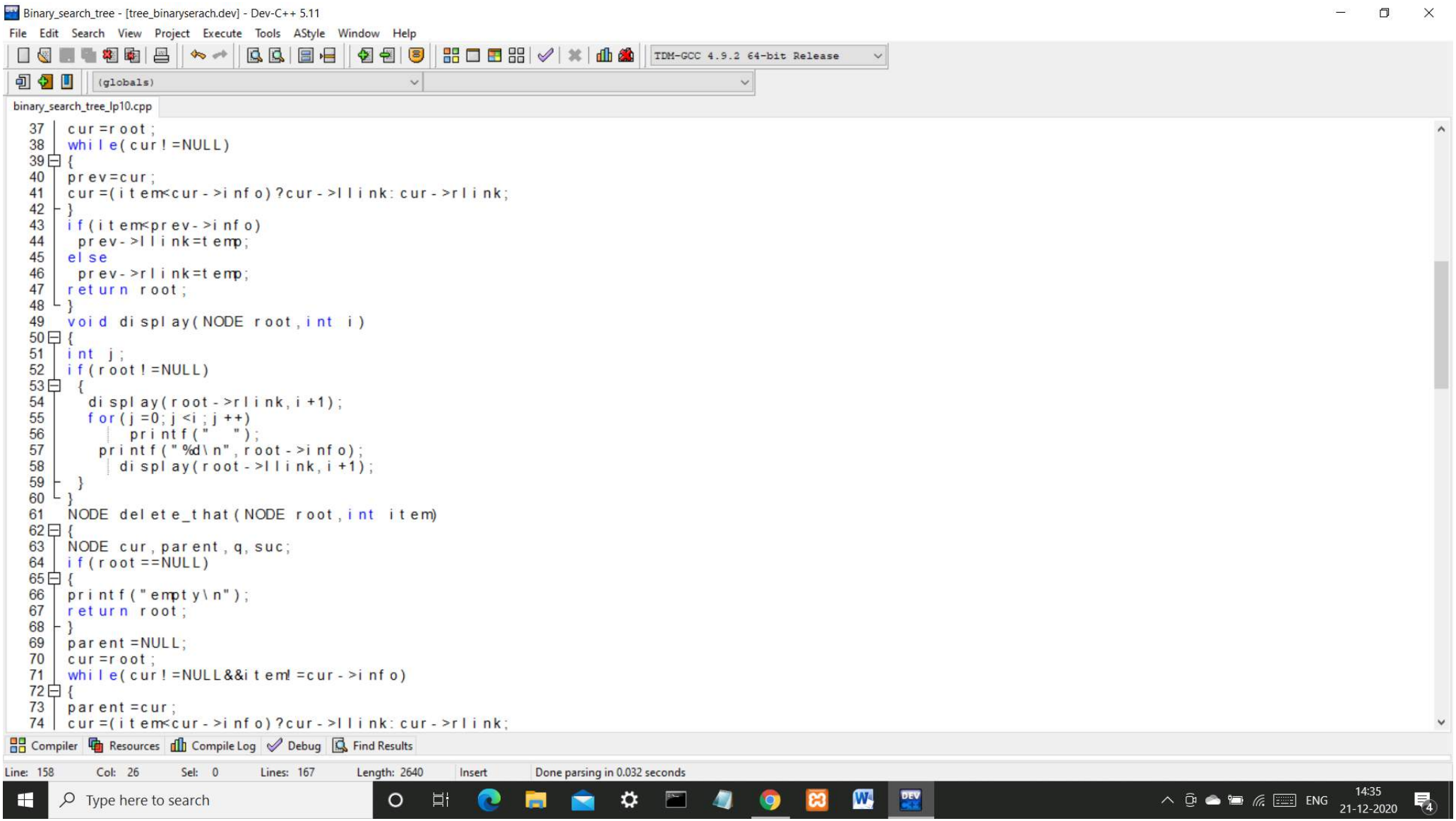
```
temp->info = item;
temp->llink = NULL;
temp->rlink = NULL;
if(root == NULL)
return temp;
printf("Give direction to insert \n");
scanf("%s", direction);
prev = NULL;
cur = root;
for(i=0; i<strlen(direction) && cur!=NULL; i++){
prev = cur;
if(direction[i] == 'l')
cur = cur->llink;
else
cur = cur->rlink;
}
if(cur!=NULL || i!=strlen(direction)){
printf("Insertion not possible \n");
freenode(temp);
return(root);
}
if(cur == NULL){
if(direction[i-1] == 'l')
prev->llink = temp;
else
prev->rlink = temp;
}return(root);
}
void preorder(NODE root){
if(root!=NULL){
printf("the item is %d \n", root->info);
preorder(root->llink);
preorder(root->rlink);
}}
```

```
void inorder(NODE root){
if(root != NULL){
inorder(root->llink);
printf("The item is %d \n", root->info);
inorder(root->rlink);
}}

void preorder(NODE root){
if(root != NULL){
preorder(root->llink);
preorder(root->rlink);
printf("The item is %d \n", root->info);
}}

void display(NODE root, int i){
int j;
if(root != NULL){
display(root->rlink, i+1);
for(j=1; j<=i; j++)
printf("  ");
printf("%d \n", root->info);
display(root->llink, i+1);
}}

int main(){
NODE root = NULL;
int choice, i, item;
for(;;){
printf("1. Insert \n 2. Preorder \n 3. Inorder \n 4. Postorder \n 5.
    Display \n");
printf("Enter the choice \n");
scanf("%d \n", &choice);
switch(choice){
case 1: printf("Enter the item: \n");
        scanf("%d", &item);
        root = insert(item, root);
        break;
```

```
case 2: if(root==NULL){
        printf("Tree is empty \n");
      }
       else{
      printf("Given tree is... ");
      display(root, 1);
      printf("The preorder traversal is: \n");
       preorder(root);
    }
      break;
case 3: if(root==NULL){
        printf("Tree is empty \n");
        }
        else{
        printf("Given tree is..");
        display(root,1);
        printf("The inorder transversal is \n");
        inorder(root);
        }
        break;
case 4: if(root==NULL){
        printf("Tree is empty \n");
    }
        else{
        printf("Given tree is..");
        display(root,1);
        printf("The postorder transversal is \n");
        postorder(root);
    }
      break;
case 5: display(root,1);
        break;
default: printf("Invalid choice entered \n");
        exit(0);
```

```
    }}
    return 0;
}
```


Binary_search_tree - [tree_binaryserach.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
binary_search_tree_lp10.cpp
#include<stdio.h>
#include<conio.h>
#include<malloc.h>
#include<process.h>
struct node
{
int info;
struct node *rlink;
struct node *llink;
};
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 insert(NODE root,int item)
{
NODE temp,cur,prev;
temp=getnode();
temp->rlink=NULL;
temp->llink=NULL;
temp->info=item;
if(root==NULL)
return temp;
prev=NULL;
cur=root;
while(cur!=NULL)
Compiler Resources Compile Log Debug Find Results
Line: 158 Col: 26 Sel: 0 Lines: 167 Length: 2640 Insert Done parsing in 0.032 seconds
Type here to search
ENG
14:35
21-12-2020


Binary_search_tree - [tree_binaryserach.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
binary_search_tree_lp10.cpp
cur=root;
while(cur!=NULL)
{
prev=cur;
cur=(item<cur->info)?cur->llink:cur->rlink;
}
if(item<prev->info)
prev->llink=temp;
else
prev->rlink=temp;
return root;
}
void display(NODE root,int i)
{
int j;
if(root!=NULL)
{
display(root->rlink,i+1);
for(j=0;j<i;j++)
printf(" ");
printf("%d\n",root->info);
display(root->llink,i+1);
}
}
NODE delete_that(NODE root,int item)
{
NODE cur,parent,q,suc;
if(root==NULL)
{
printf("empty\n");
return root;
}
parent=NULL;
cur=root;
while(cur!=NULL&&item!=cur->info)
{
parent=cur;
cur=(item<cur->info)?cur->llink:cur->rlink;
Compiler Resources Compile Log Debug Find Results
Line: 158 Col: 26 Sel: 0 Lines: 167 Length: 2640 Insert Done parsing in 0.032 seconds
Type here to search
ENG
14:35
21-12-2020

Binary_search_tree - [tree_binaryserach.dev] - Dev-C++ 5.11

File Edit Search View Project Execute Tools AStyle Window Help

TDM-GCC 4.9.2 64-bit Release

(globals)

binary_search_tree_lp10.cpp

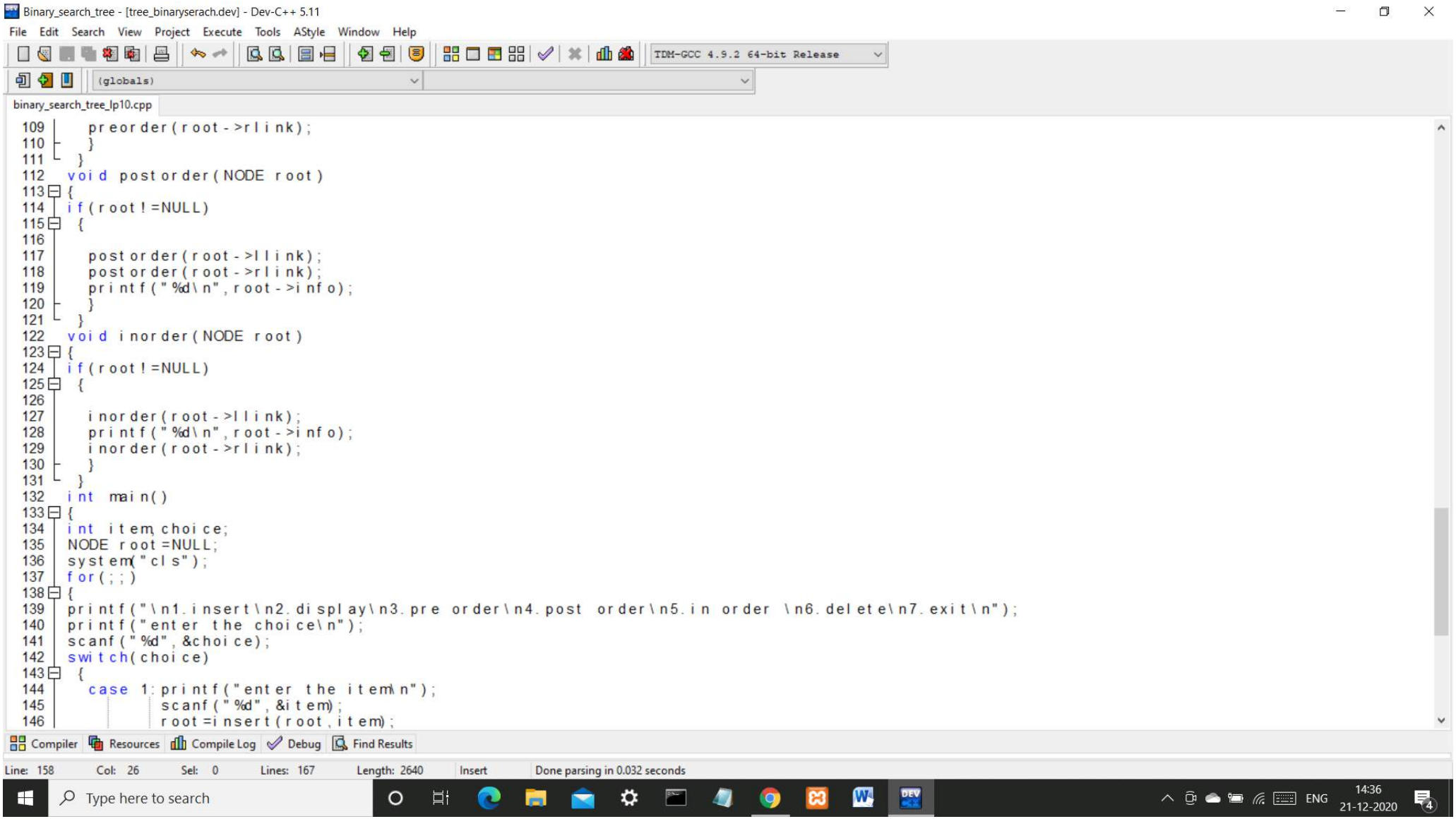
```
 73  parent=cur;
 74  cur=(item<cur->info)?cur->llink:cur->rlink;
 75  }
 76  if(cur==NULL)
 77  {
 78   printf("not found\n");
 79   return root;
 80  }
 81  if(cur->llink==NULL)
 82   q=cur->rlink;
 83  else if(cur->rlink==NULL)
 84   q=cur->llink;
 85  else
 86   {
 87   suc=cur->rlink;
 88   while(suc->llink!=NULL)
 89    suc=suc->llink;
 90   suc->llink=cur->llink;
 91   q=cur->rlink;
 92   }
 93   if(parent==NULL)
 94    return q;
 95   if(cur==parent->llink)
 96    parent->llink=q;
 97   else
 98    parent->rlink=q;
 99   freenode(cur);
100   return root;
101   }
102
103  void preorder(NODE root)
104  {
105  if(root!=NULL)
106   {
107    printf("%d\n",root->info);
108    preorder(root->llink);
109    preorder(root->rlink);
110    }
```

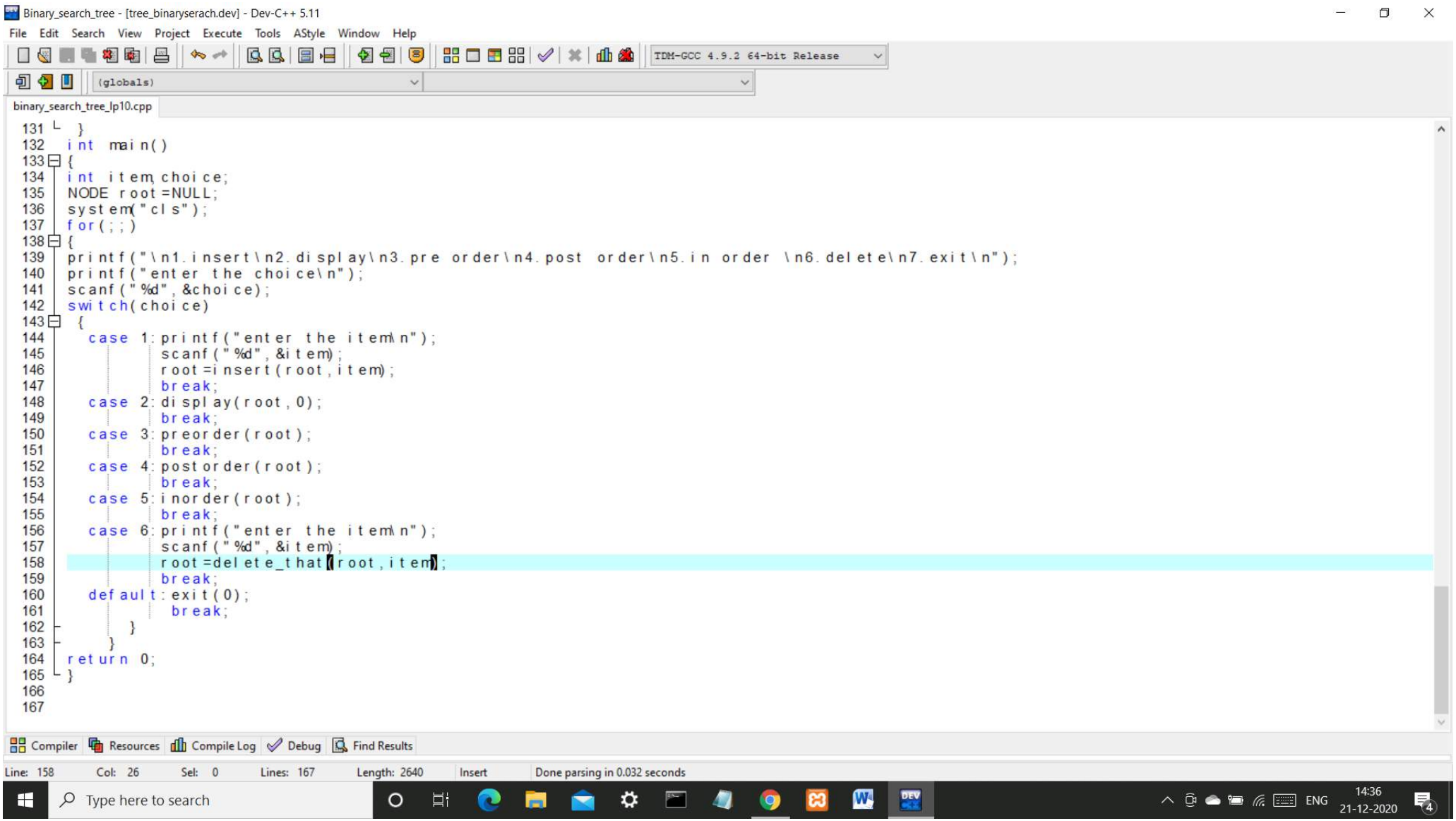
Compiler Resources Compile Log Debug Find Results

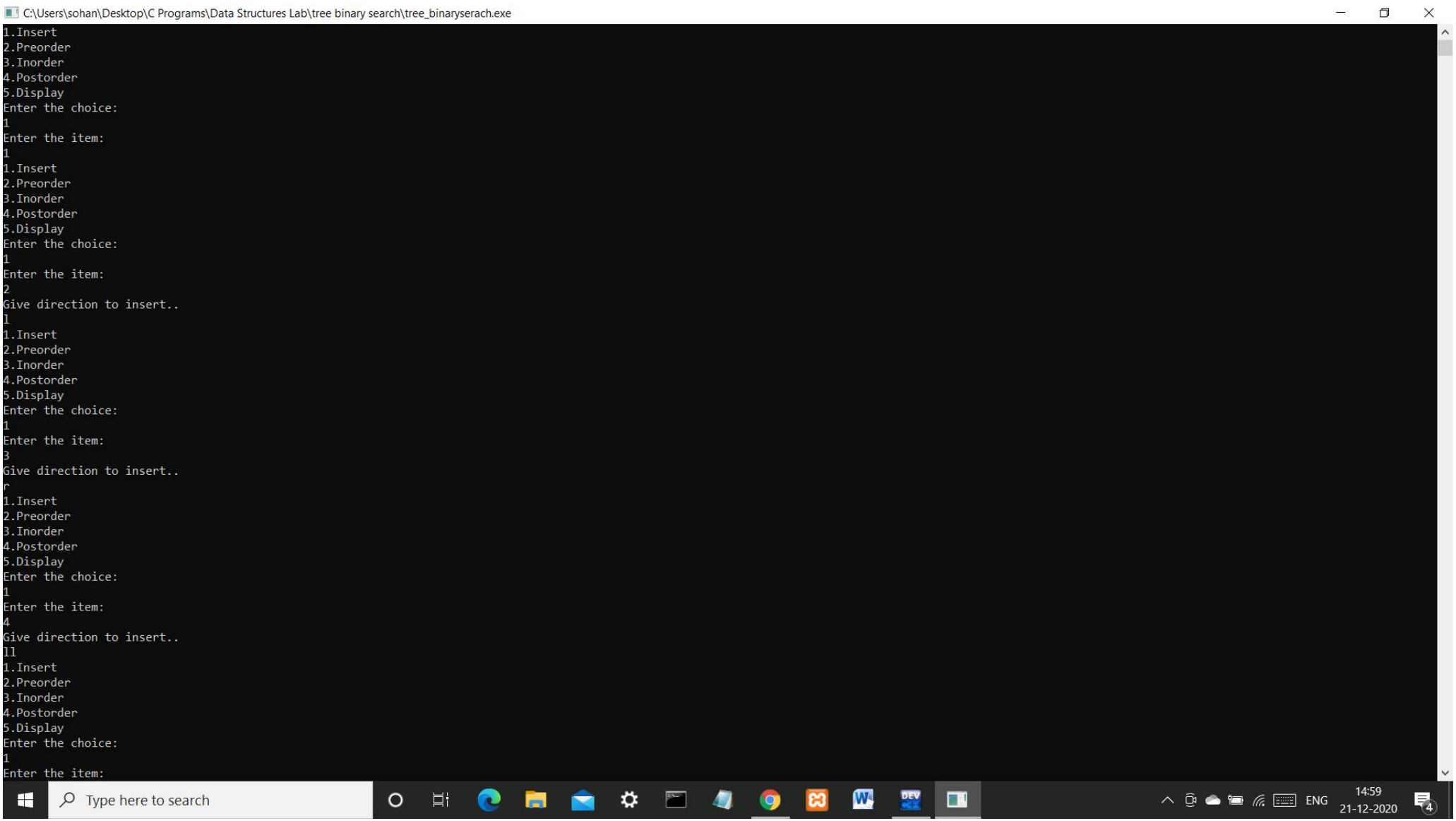
Line: 158 Col: 26 Sel: 0 Lines: 167 Length: 2640 Insert Done parsing in 0.032 seconds

Type here to search

ENG 14:36 21-12-2020


Binary_search_tree - [tree_binaryserach.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
binary_search_tree_lp10.cpp
preorder(root->rlink);
}
}
void postorder(NODE root)
{
if(root!=NULL)
{
postorder(root->llink);
postorder(root->rlink);
printf("%d\n",root->info);
}
}
void inorder(NODE root)
{
if(root!=NULL)
{
inorder(root->llink);
printf("%d\n",root->info);
inorder(root->rlink);
}
}
int main()
{
int item,choice;
NODE root=NULL;
system("cls");
for(;;)
{
printf("\n1.insert\n2.display\n3.pre order\n4.post order\n5.in order \n6.delete\n7.exit\n");
printf("enter the choice\n");
scanf("%d",&choice);
switch(choice)
{
case 1:printf("enter the item\n");
scanf("%d",&item);
root=insert(root,item);
Compiler Resources Compile Log Debug Find Results
Line: 158 Col: 26 Sel: 0 Lines: 167 Length: 2640 Insert Done parsing in 0.032 seconds
Type here to search
ENG
14:36
21-12-2020


Binary_search_tree - [tree_binaryserach.dev] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
TDM-GCC 4.9.2 64-bit Release
(globals)
binary_search_tree_lp10.cpp
}
int main()
{
int item,choice;
NODE root=NULL;
system("cls");
for(;;)
{
printf("\n1.insert\n2.display\n3.pre order\n4.post order\n5.in order \n6.delete\n7.exit\n");
printf("enter the choice\n");
scanf("%d",&choice);
switch(choice)
{
case 1:printf("enter the item\n");
scanf("%d",&item);
root=insert(root,item);
break;
case 2:display(root,0);
break;
case 3:preorder(root);
break;
case 4:postorder(root);
break;
case 5:inorder(root);
break;
case 6:printf("enter the item\n");
scanf("%d",&item);
root=delete_that(root,item);
break;
default:exit(0);
break;
}
}
return 0;
}
Compiler
Resources
Compile Log
Debug
Find Results
Line: 158 Col: 26 Sel: 0 Lines: 167 Length: 2640 Insert Done parsing in 0.032 seconds
Type here to search
ENG
14:36
21-12-2020


C:\Users\sohan\Desktop\C Programs\Data Structures Lab\tree binary search\tree_binaryserach.exe
1.Insert
2.Preorder
3.Inorder
4.Postorder
5.Display
Enter the choice:
1
Enter the item:
1
1.Insert
2.Preorder
3.Inorder
4.Postorder
5.Display
Enter the choice:
1
Enter the item:
2
Give direction to insert..
l
1.Insert
2.Preorder
3.Inorder
4.Postorder
5.Display
Enter the choice:
1
Enter the item:
3
Give direction to insert..
r
1.Insert
2.Preorder
3.Inorder
4.Postorder
5.Display
Enter the choice:
1
Enter the item:
4
Give direction to insert..
ll
1.Insert
2.Preorder
3.Inorder
4.Postorder
5.Display
Enter the choice:
1
Enter the item:
Type here to search
ENG
14:59
21-12-2020

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\tree binary search\tree_binaryserach.exe

```
3.Inorder
4.Postorder
5.Display
Enter the choice:
1
Enter the item:
5
Give direction to insert..
lr
1.Insert
2.Preorder
3.Inorder
4.Postorder
5.Display
Enter the choice:
1
Enter the item:
6
Give direction to insert..
rl
1.Insert
2.Preorder
3.Inorder
4.Postorder
5.Display
Enter the choice:
1
Enter the item:
7
Give direction to insert..
rr
1.Insert
2.Preorder
3.Inorder
4.Postorder
5.Display
Enter the choice:
5
      7
    3
      6
  1
      5
    2
      4
1.Insert
2.Preorder
3.Inorder
4.Postorder
5.Display
```

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\tree binary search\tree_binaryserach.exe

```
2.Preorder
3.Inorder
4.Postorder
5.Display
Enter the choice:
2
Given tree is..      7
    3
      6
  1
      5
    2
      4
The preorder traversal is:
the item is 1
the item is 2
the item is 4
the item is 5
the item is 3
the item is 6
the item is 7
1.Insert
2.Preorder
3.Inorder
4.Postorder
5.Display
Enter the choice:
3
Given tree is..      7
    3
      6
  1
      5
    2
      4
The inorder traversal is
The item is4
The item is2
The item is5
The item is1
The item is6
The item is3
The item is7
1.Insert
2.Preorder
3.Inorder
4.Postorder
5.Display
Enter the choice:
4
```

C:\Users\sohan\Desktop\C Programs\Data Structures Lab\tree binary search\tree_binaryserach.exe

```
3.Inorder
4.Postorder
5.Display
Enter the choice:
4
Given tree is..      7
    3
      6
  1
      5
    2
      4
The postorder traversal is
The item is4
The item is5
The item is2
The item is6
The item is7
The item is3
The item is1
1.Insert
2.Preorder
3.Inorder
4.Postorder
5.Display
Enter the choice:
5
      7
    3
      6
  1
      5
    2
      4
1.Insert
2.Preorder
3.Inorder
4.Postorder
5.Display
Enter the choice:
6
Invalid choice entered.

--------------------------------
Process exited after 226.4 seconds with return value 0
Press any key to continue . . .
```