**TUGAS JOBSHEET 04**

**PRAKTIKUM STRUKTUR DATA**

****

**DOSEN PENGAMPU:**

**Vera Irma Delianti, S.Pd., M.Pd.T.**

**OLEH:**

**M. ilham**

**23343008**

**PROGRAM STUDI INFORMATIKA  
DEPARTEMEN TEKNIK ELEKTRONIKA**

**FAKULTAS TEKNIK**

**UNIVERSITAS NEGERI PADANG**

**2024**

1. **implementasi-array-dari-stack.c**
   1. **SOURCE CODE**

/\* Nama file : implementasi array dari stack

Pembuat : M. Ilham 23343008

Tgl pembuatan : 26 February 2024\*/

#include <stdio.h>

int stack[100], i, j, choice = 0, n, top = -1;

void push();

void pop();

void show();

int main(){

int choice = 0;

printf("\*\*\*\*\*\*\*\*\*Stack operations using array\*\*\*\*\*\*\*\*\*\n");

printf("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

printf("\nEnter the number of elements in the stack : ");

scanf("%d", &n);

printf("\n--------------------------------------------------");

while(choice != 4){

printf("\n\nChoose one from the below options...\n");

printf("\n 1. Push\n 2. Pop\n 3. Show\n 4. Exit");

printf("\nEnter your choice : ");

scanf("%d", &choice);

switch(choice){

case 1:

push();

break;

case 2:

pop();

break;

case 3:

show();

break;

case 4:

printf("Exiting...");

break;

default:

printf("Please Enter valid choice");

break;

}

}

return 0;

}

void push(){

int val;

if(top == n - 1){

printf("\nOverflow");

}

else {

printf("Enter the value?");

scanf("%d", &val);

top = top + 1;

stack[top] = val;

}

}

void pop(){

if(top == -1){

printf("Underflow");

}

else {

top = top - 1;

}

}

void show(){

for(i = top; i >= 0; i--){

printf("%d\n", stack[i]);

}

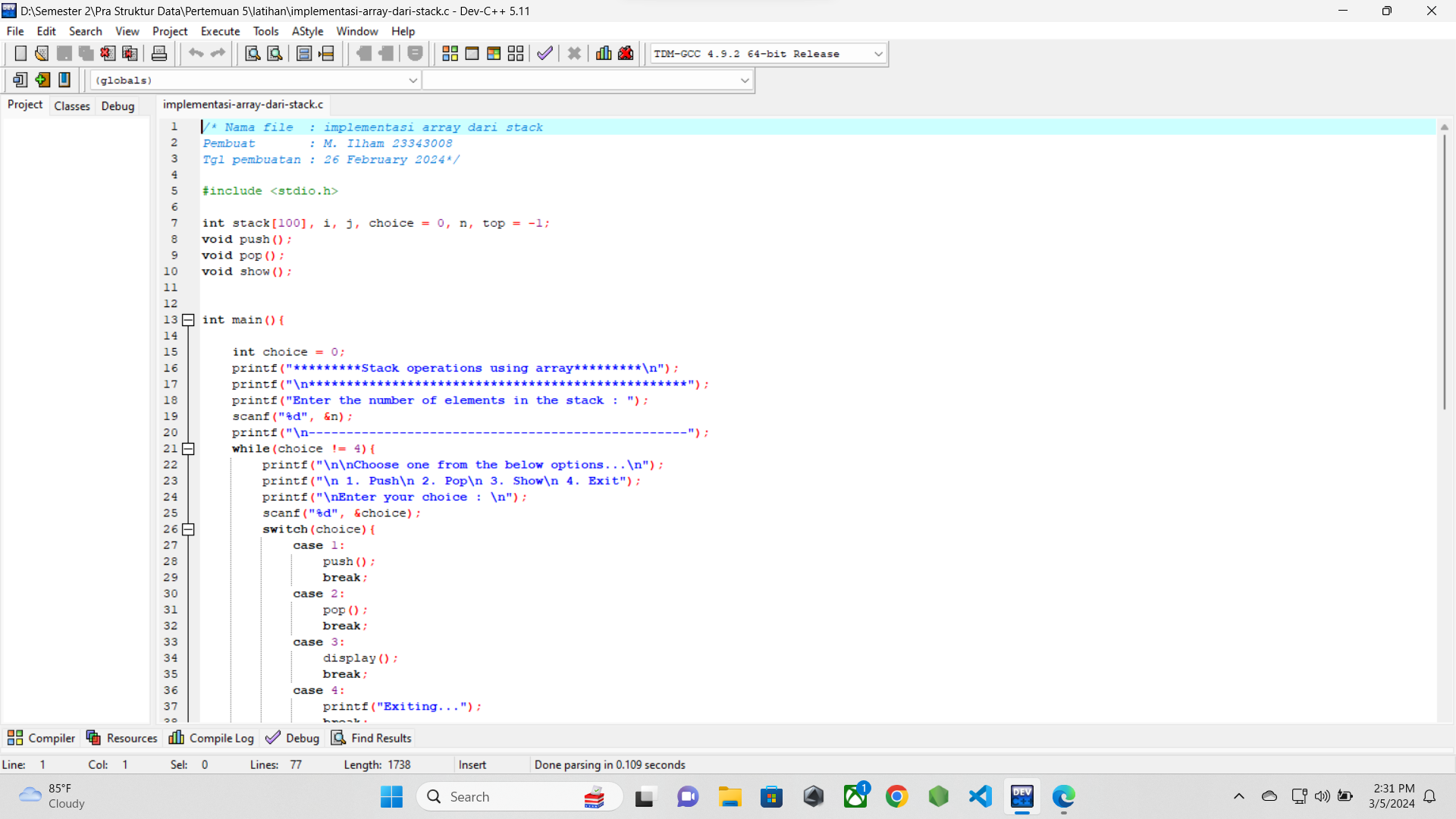
if(top == -1){

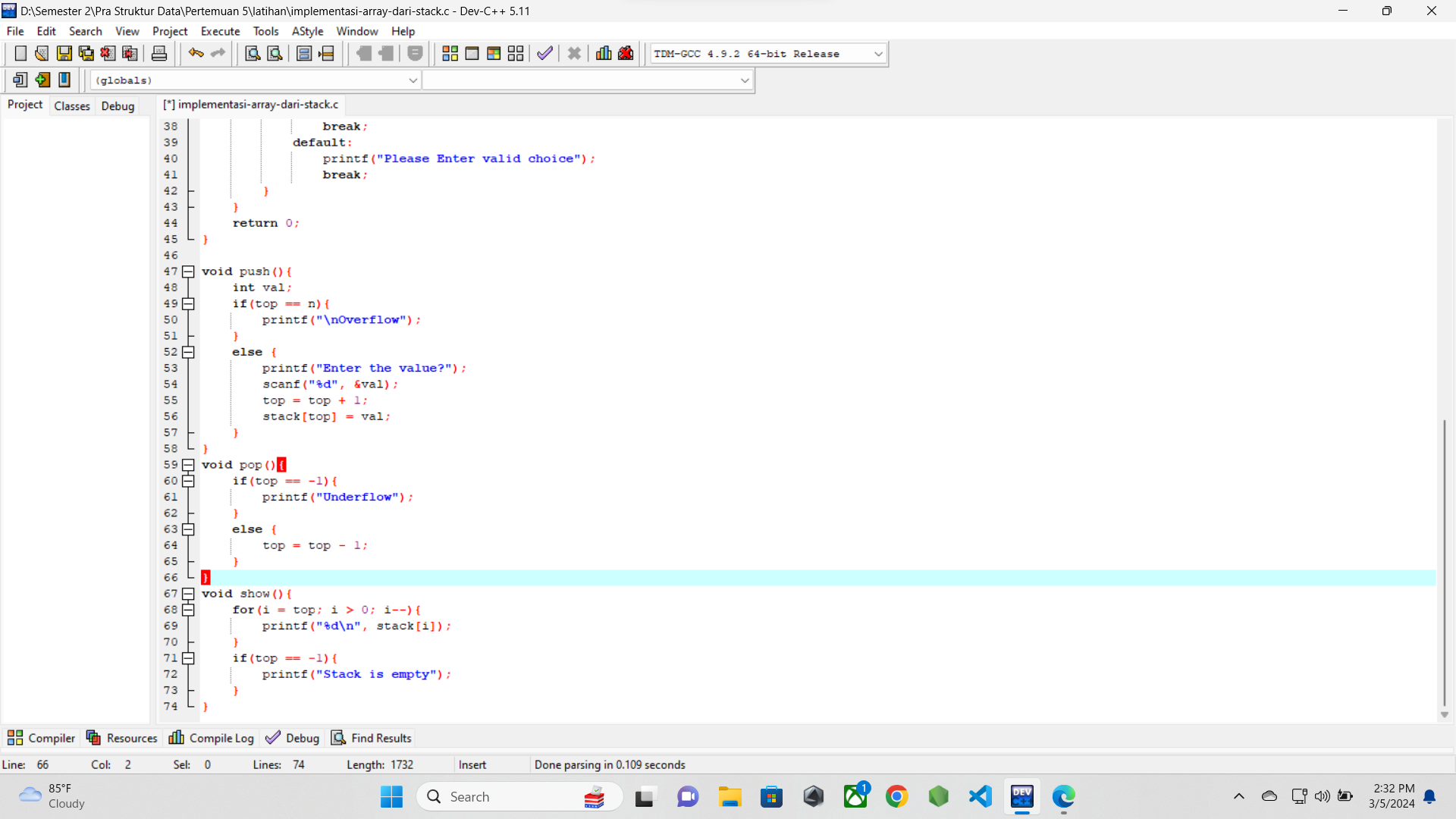
printf("Stack is empty");

}

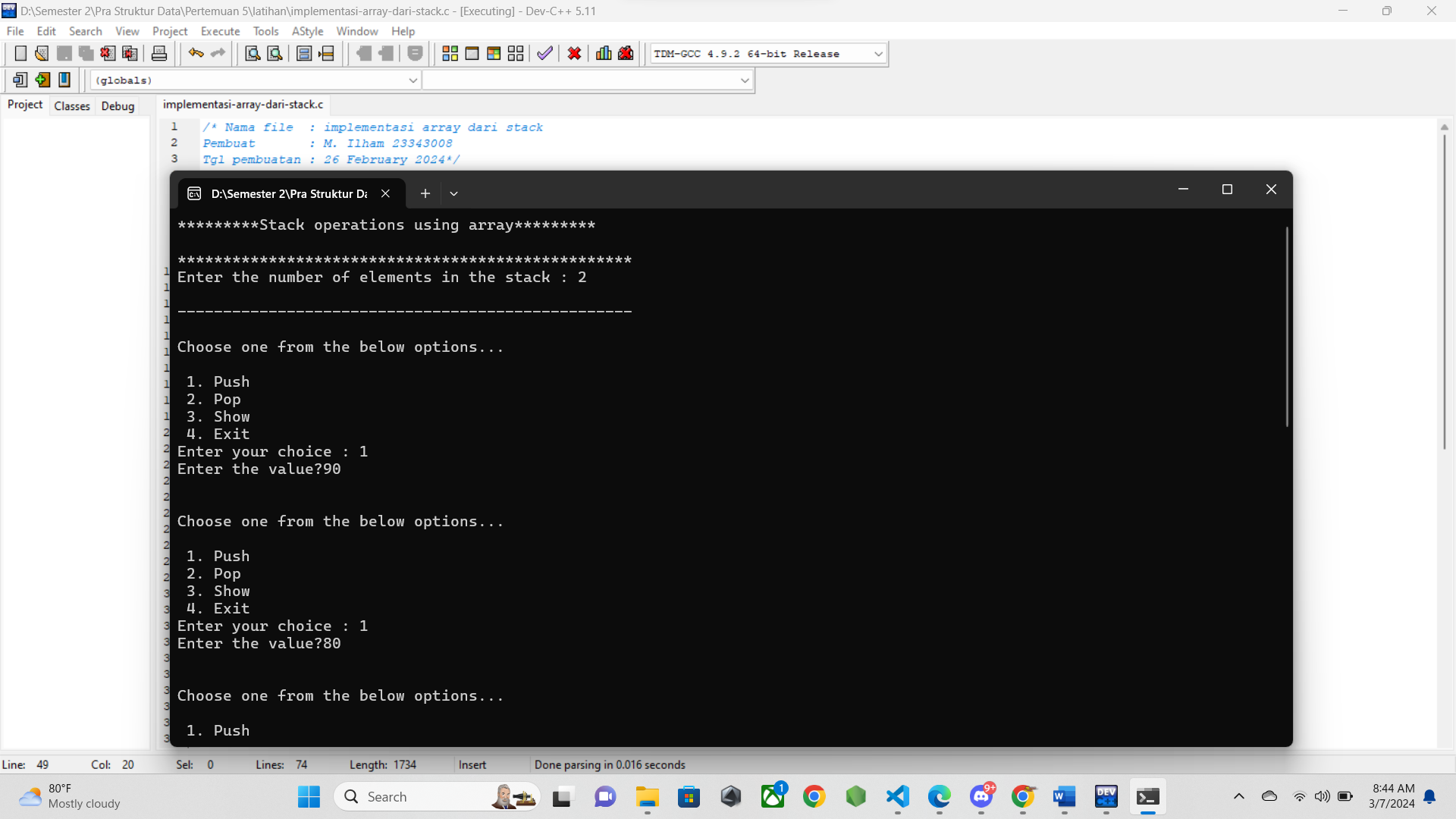
}

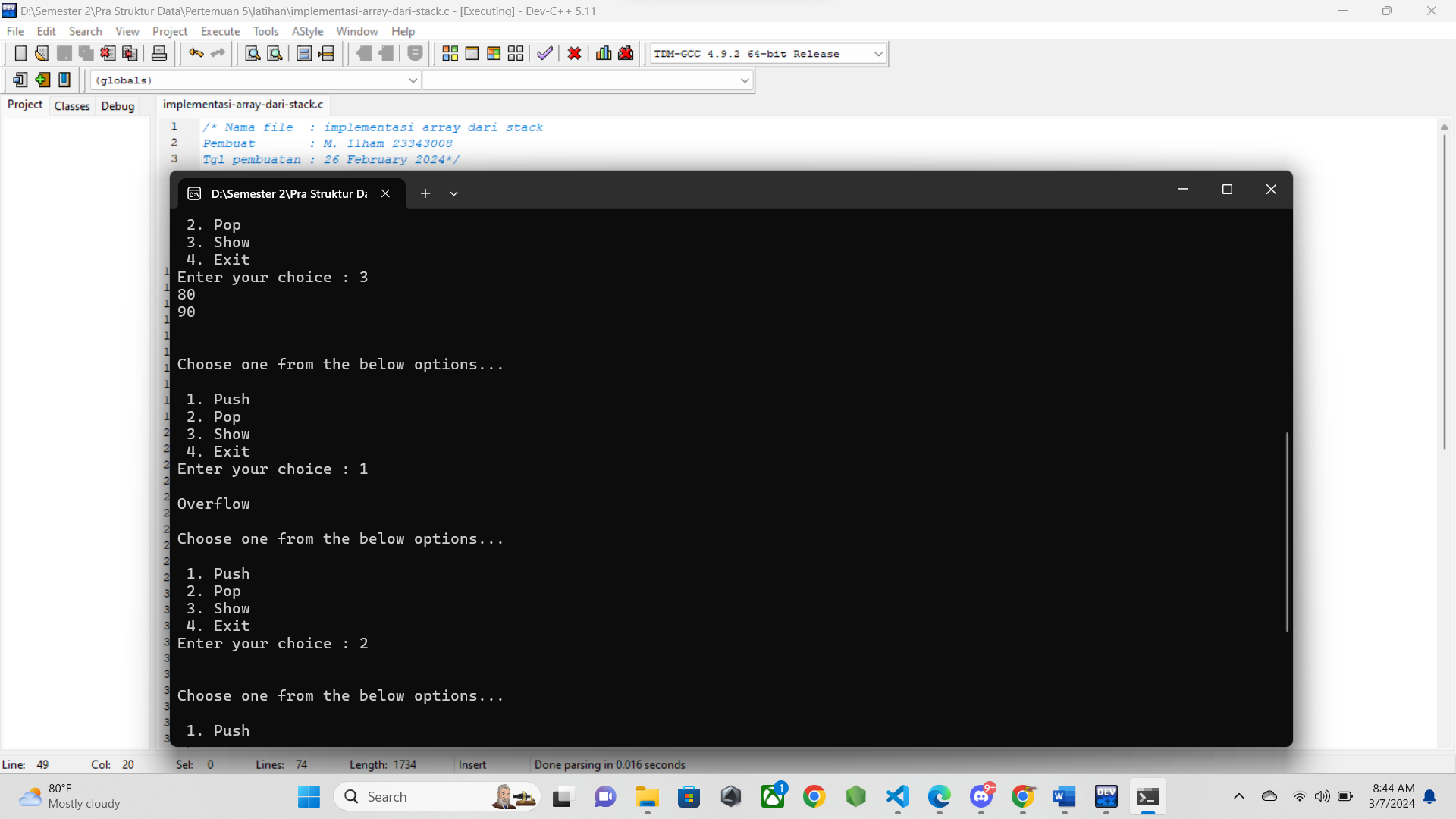
* 1. **SCREENSHOT PROGRAM**

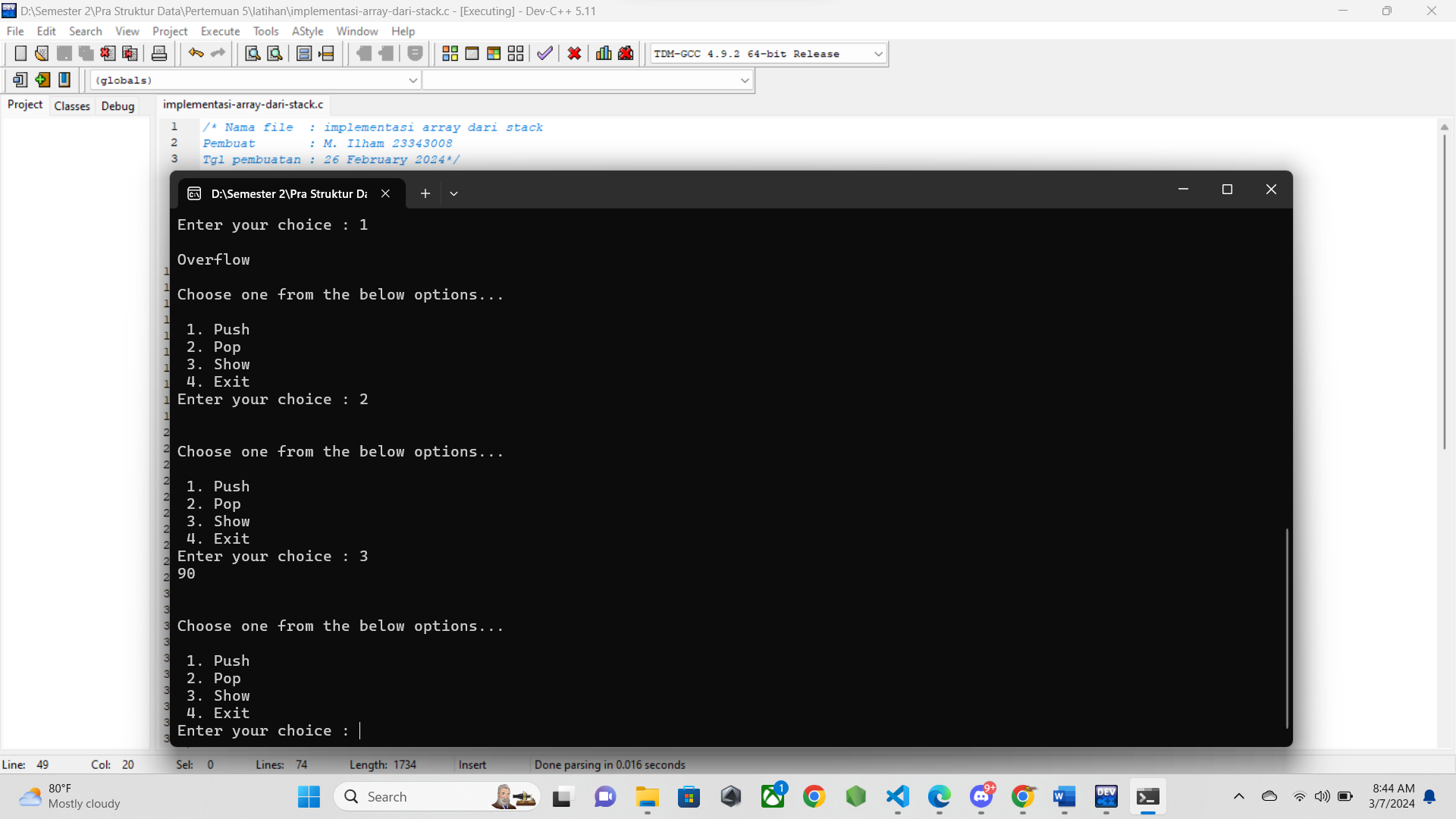




* 1. **SCREENSHOT OUTPUT**







* 1. **PENJELASAN PROGRAM**

Pada program ini, akan di implementasikan stack dengan array, saat element baru di tambahkan, ia akan diletakkan di element paling belakang. Saat element di hapus, element di paling belakang jugalah yang di hapus, namun karena ini menggunakan alokasi memori otomatis berarti memori akan di alokasikan di stack sehingga tidak bisa melakukan pembebasan memori, karena itu pada fungsi pop hanya akan menghapus akses ke element paling belakang.

Saat melakukan show, maka akan di tampilkan satu persatu dari element paling akhir hingga paling awal sehingga terkesan ditampilkan dari element paling atas tumpukan.

1. **implementasi-linked-list-dari-stack.c**
   1. **SOURCE CODE**

/\* Nama file : implementasi linked list dari Stack

Pembuat : M. Ilham 23343008

Tgl pembuatan : 26 February 2024\*/

#include <stdio.h>

void push();

void pop();

void display();

struct node {

int val;

struct node \*next;

};

struct node \*head;

int main(){

int choice = 0;

printf("\*\*\*\*\*\*\*\*\*Stack operations using linked list\*\*\*\*\*\*\*\*\*\n");

printf("\n--------------------------------------------------");

while(choice != 4){

printf("\n\nChoose one from the below options...\n");

printf("\n 1. Push\n 2. Pop\n 3. Show\n 4. Exit");

printf("\nEnter your choice : \n");

scanf("%d", &choice);

switch(choice){

case 1:

push();

break;

case 2:

pop();

break;

case 3:

display();

break;

case 4:

printf("Exiting...");

break;

default:

printf("Please Enter valid choice");

break;

}

}

return 0;

}

void push(){

int val;

struct node \*ptr = (struct node \*)malloc(sizeof(struct node));

if(ptr == NULL){

printf("just and only able to push the element");

}

else {

printf("Enter the value : ");

scanf("%d", &val);

if(head == NULL){

ptr->val = val;

ptr->next = NULL;

head = ptr;

}

else {

ptr->val = val;

ptr->next = head;

head = ptr;

}

printf("Item pushed");

}

}

void pop(){

int item;

struct node \*ptr;

if(head == NULL){

printf("Underflow");

}

else {

item = head->val;

ptr = head;

head = head->next;

free(ptr);

printf("Item popped");

}

}

void display(){

int i;

struct node \*ptr;

ptr = head;

if(ptr == NULL){

printf("Stack if empty\n");

}

else {

printf("Printing Stack elements\n");

while(ptr != NULL){

printf("%d\n", ptr->val);

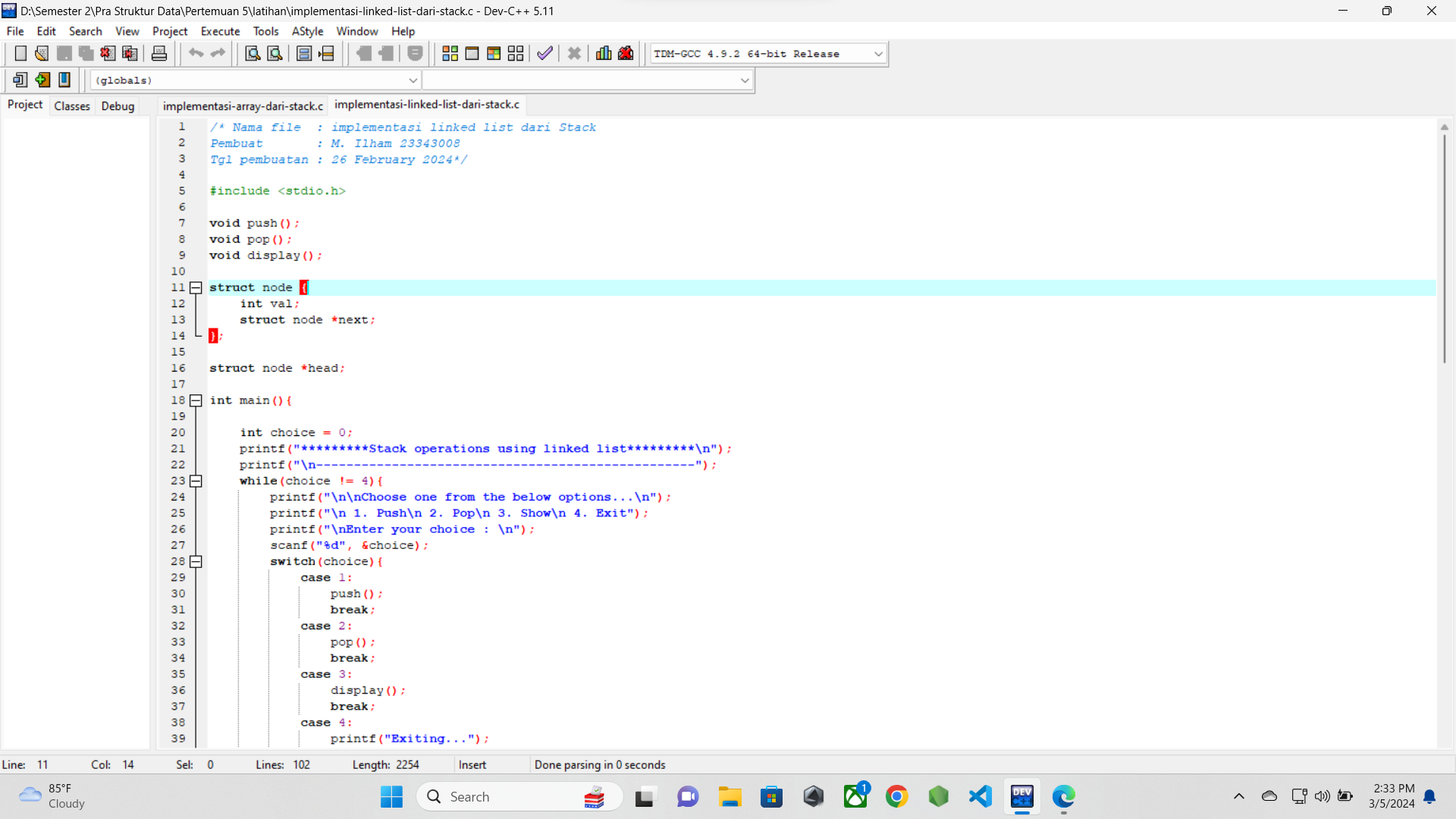
ptr = ptr->next;

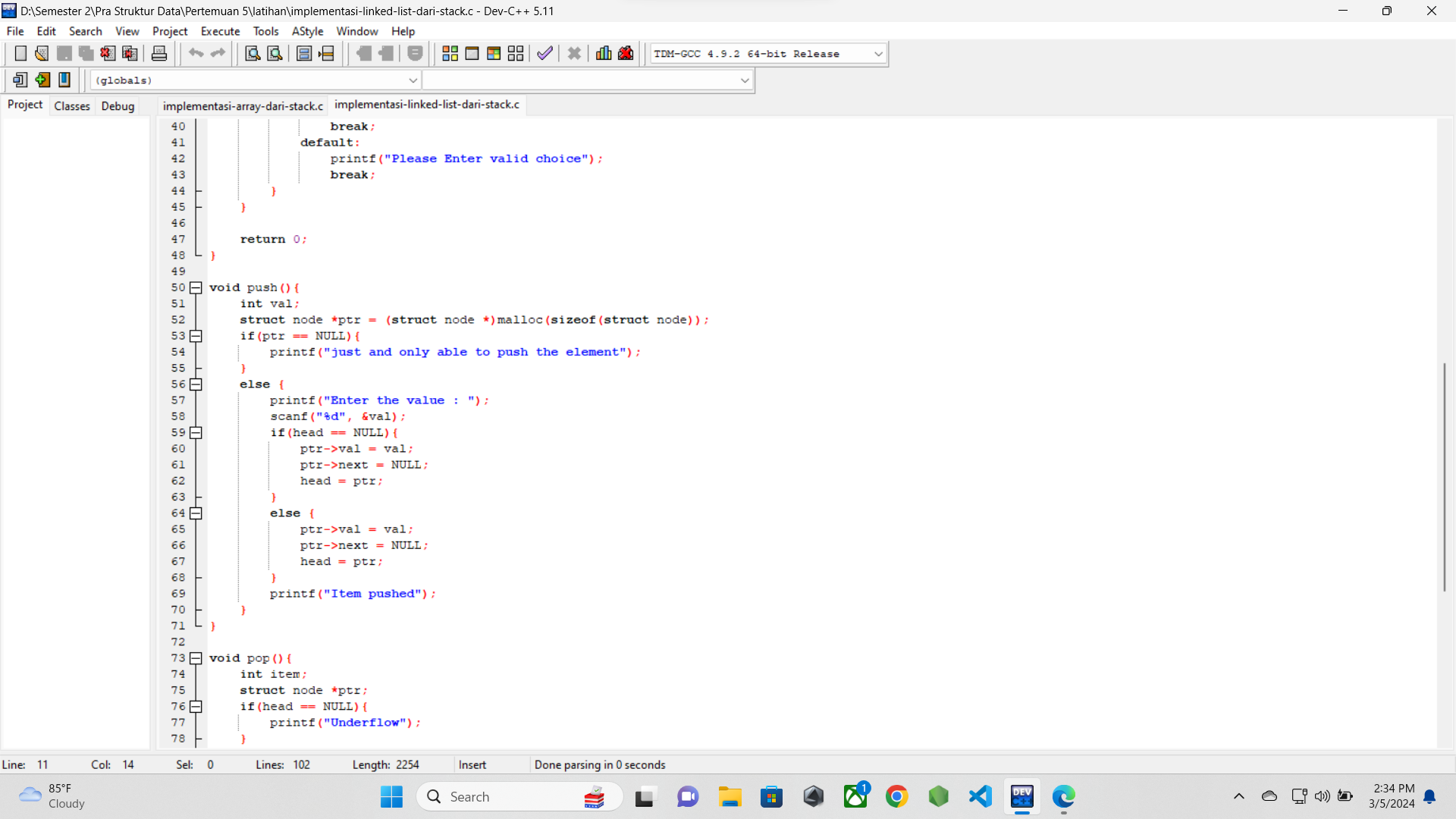
}

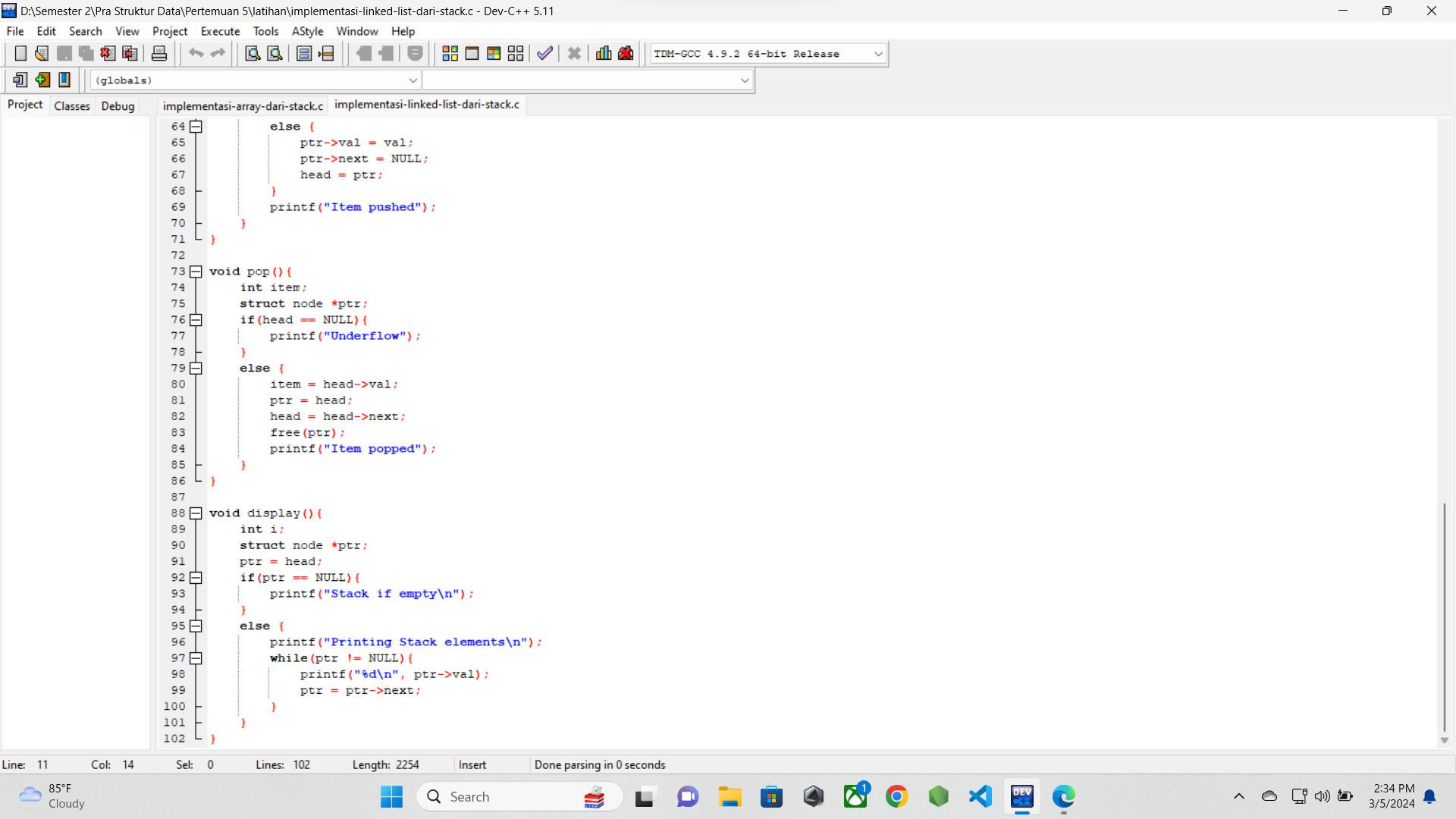
}

}

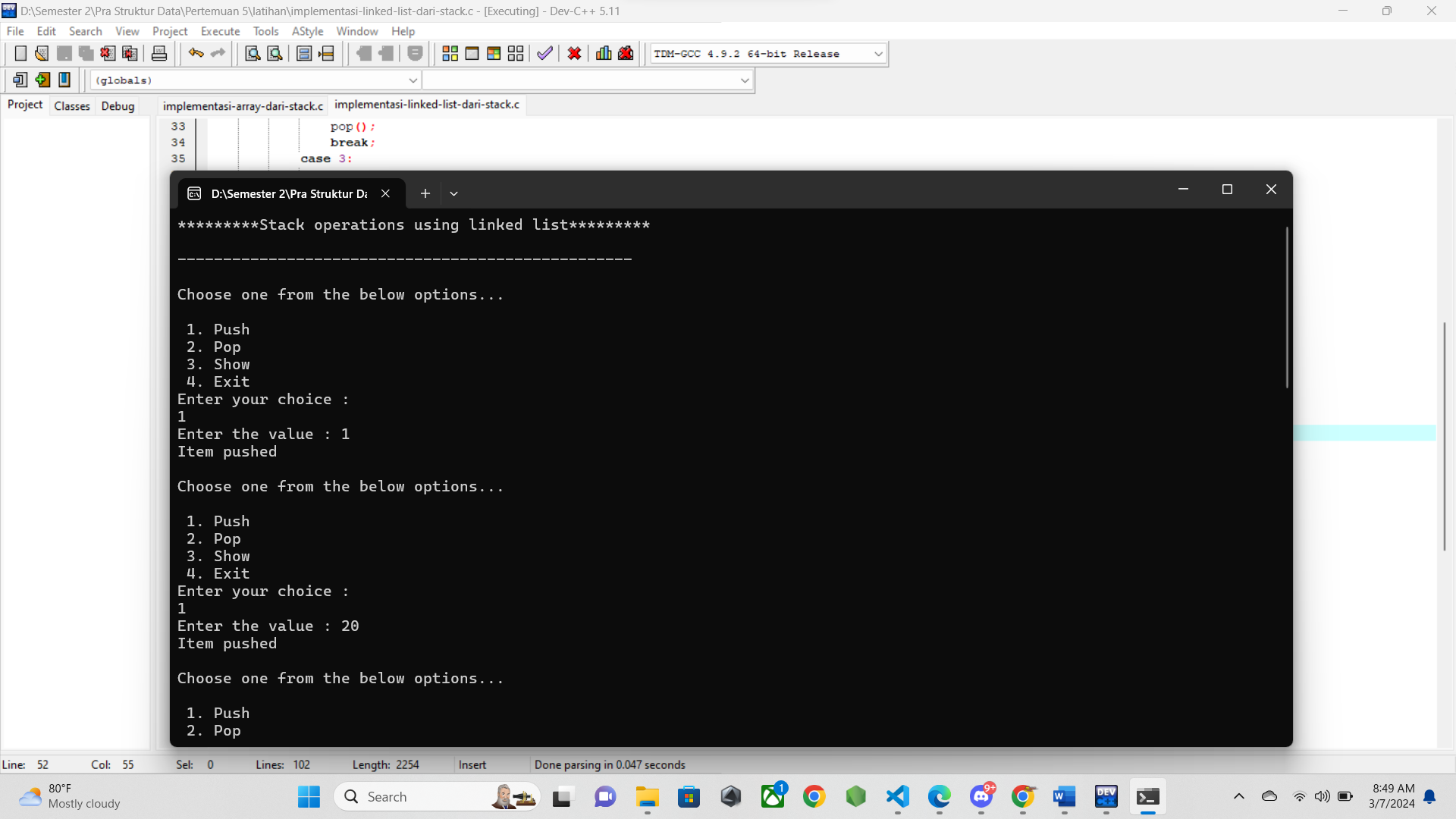
* 1. **SCREENSHOT PROGRAM**

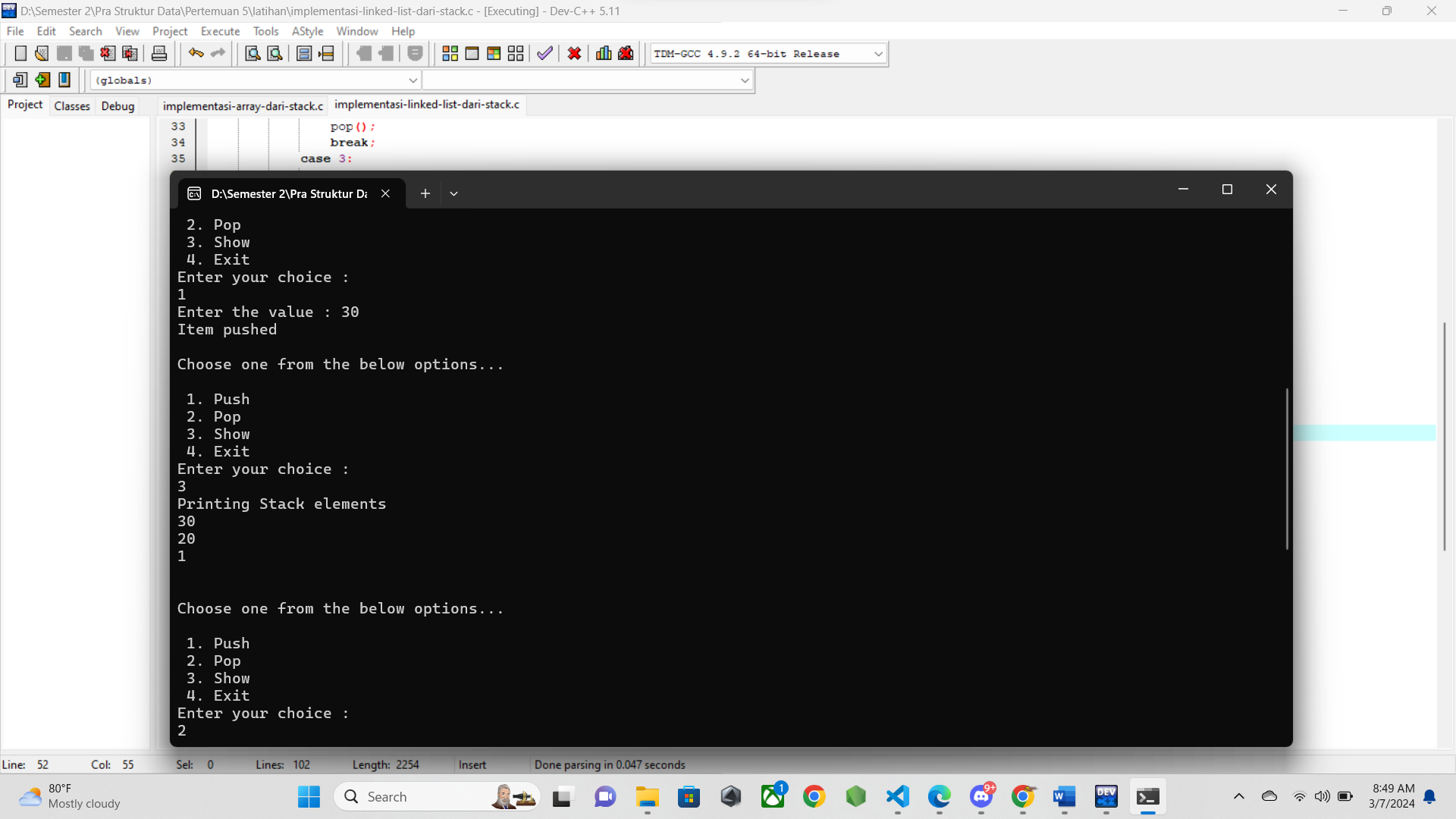


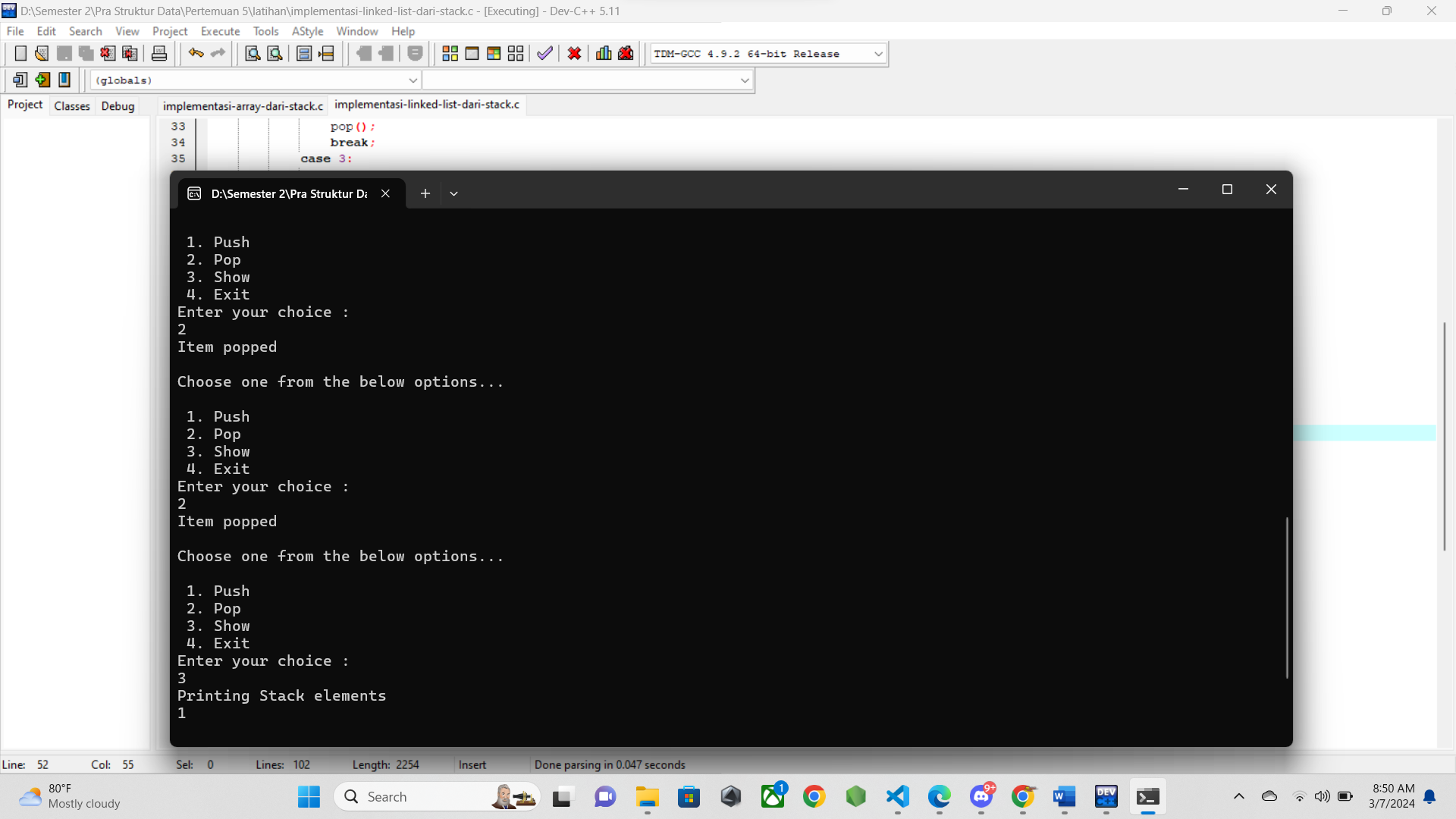




* 1. **SCREENSHOT OUTPUT**







* 1. **PENJELASAN PROGRAM**

Pada program ini digunakan linkedlist, sehingga saat melakukan push element baru, element akan diletakkan di node paling depan karena dengan linkedlist kita bisa lebih leluasa meletakkan posisi dari node. Pada saat pop, element paling depan yaitu headnya akan di hapus sehingga nodenya tidak bisa di akses lagi. Lalu posisi head di atur ke node selanjutnya.

Pada saat show, akan ditampilkan element secara berurutan dengan menggunakan while loop sehingga node ditampilkan dari yang terbaru atau paling atas ke yang paling bawah

1. **polindrom.c**
   1. **SOURCE CODE**

/\* Nama file : polindrom

Pembuat : M. Ilham 23343008

Tgl pembuatan : 26 February 2024\*/

#include <stdio.h>

#include <string.h>

#define MAXSTACK 100

#define MAXCOLUMN 30

char stack[MAXSTACK][MAXCOLUMN];

int top = -1;

void check(){

int i, j;

int isPolyndrome = 1;

int length = strlen(stack[top]);

for(i = 0, j = length - 1; i < j; i++, j--){

if(stack[top][i] != stack[top][j]){

isPolyndrome = 0;

break;

}

}

if(!isPolyndrome) printf("Kalimat tersebut bukan polyndrom\n");

else printf("Kalimat tersebut adalah polyndrom\n");

}

void push(char kalimat[]){

if(top == MAXSTACK) puts("Penuh");

else {

if (kalimat[strlen(kalimat) - 1] == '\n')

kalimat[strlen(kalimat) - 1] = '\0';

++top;

strcpy(stack[top], kalimat);

}

}

int main(){

char kalimat[MAXCOLUMN];

char run;

do {

printf("\nSelamat datang di program menentukan kalimat polyndrom dengan stack\n");

printf("Masukkan kalimat : ");

fgets(kalimat, sizeof(kalimat), stdin);

push(kalimat);

check();

printf("Ulang ? y/n : ");

scanf("%c", &run);

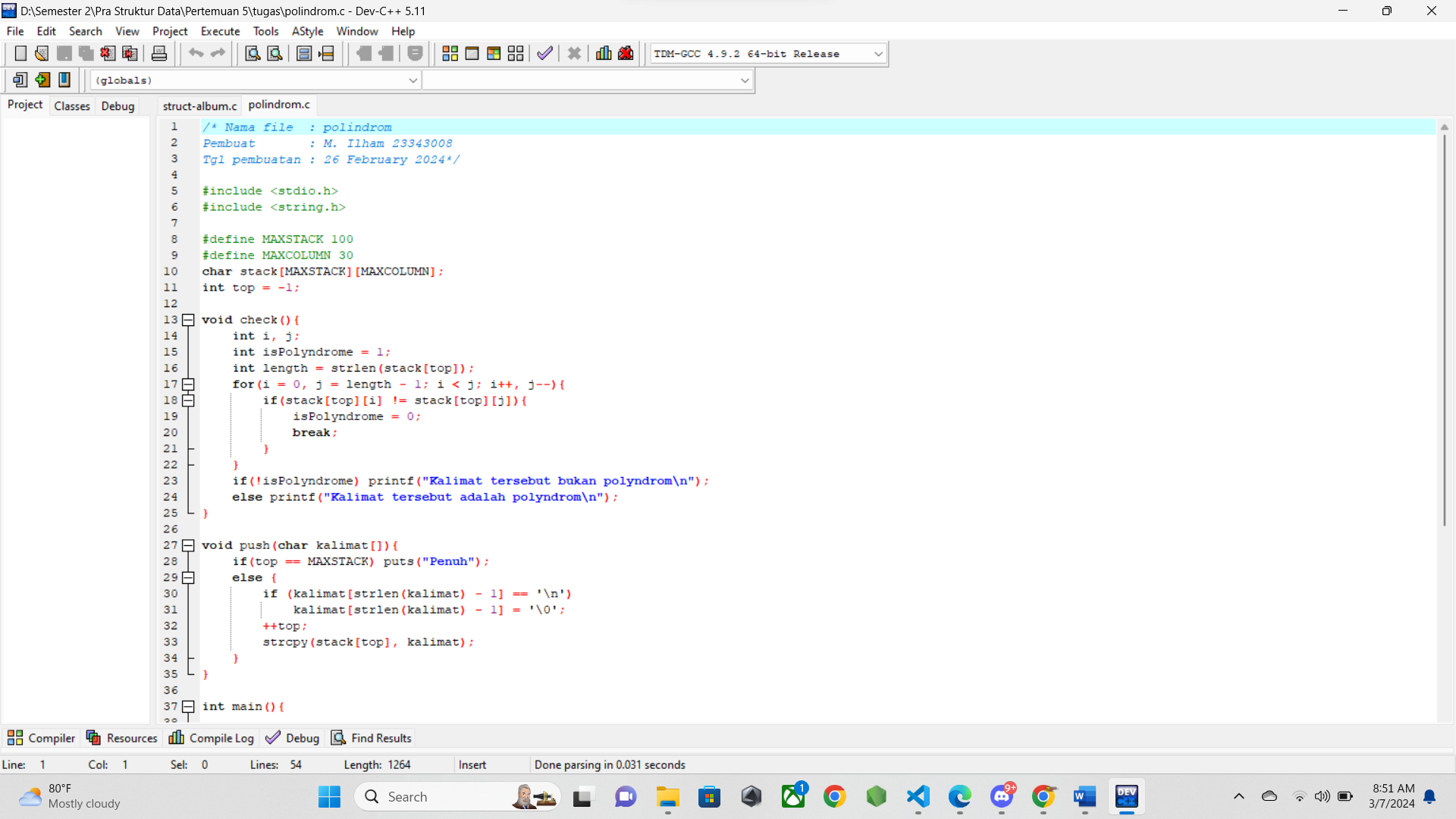
getchar();

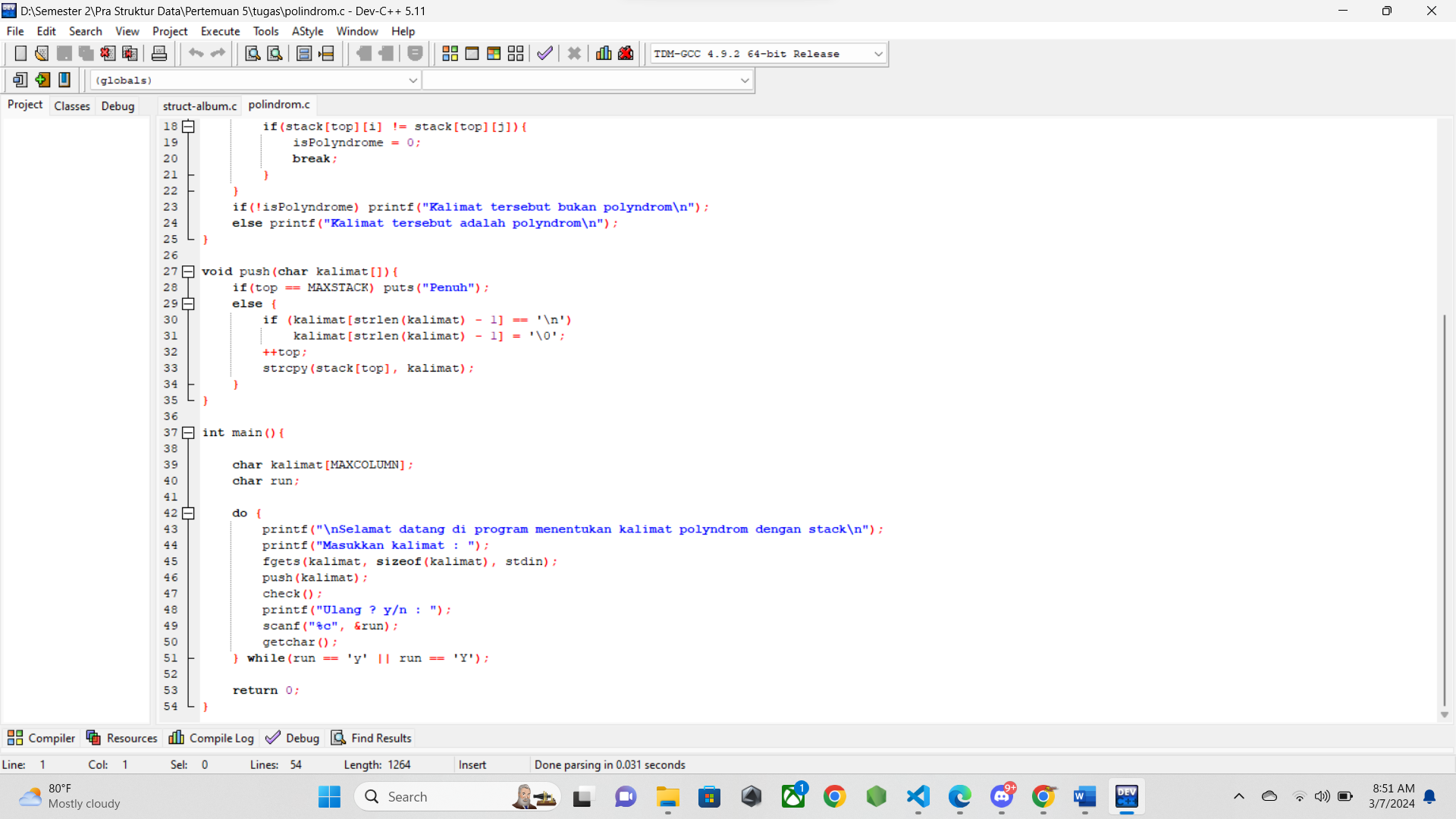
} while(run == 'y' || run == 'Y');

return 0;

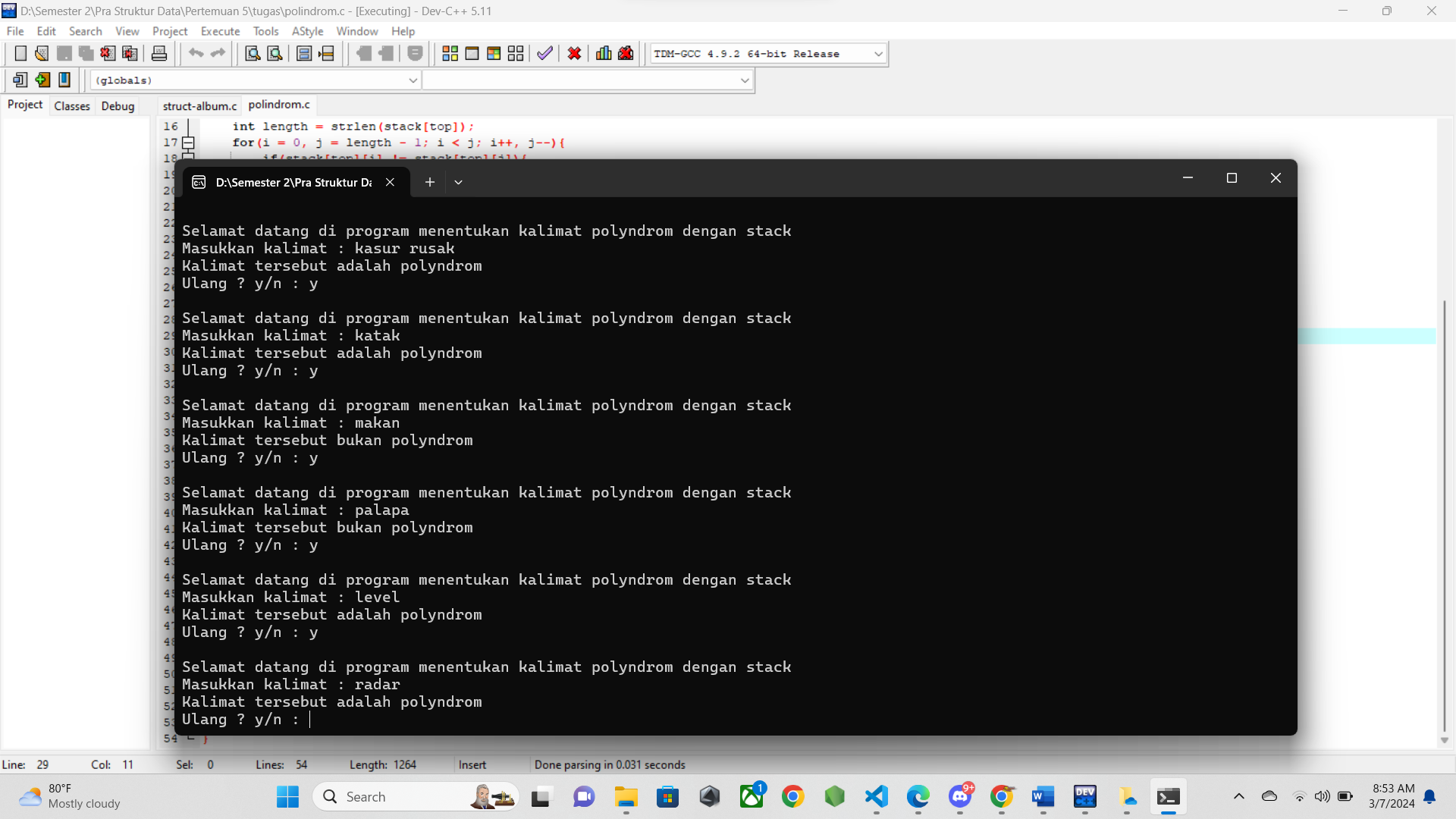
}

* 1. **SCREENSHOT PROGRAM**





* 1. **SCREENSHOT OUTPUT**



* 1. **PENJELASAN PROGRAM**

Pada program ini untuk mencari sebuah kalimat polindrom atau tidak. Pertama akan diminta kalimat yang akan diperiksa, lalu tiap karakter dari string yang di input akan dimasukkan kedalam variabel array dua dimensi bernama stack. Lalu untuk memeriksa apakah kalimatnya polindrom dengan method check().

Didalam method check akan diperiksa apakah element awal dan akhirnya sama, lalu akan dicek setiap iterasi, jika ada kalimat yang berbeda program pengecekan dengan loop akan berhenti dan di dapatkan hasil bukan polindrom, jika sampai akhir berjalannya program looping tidak ada kalimat yang berbeda, kalimat dinyatakan sebaai polindrom.