**LAPORAN PRATIKUM STRUKTUR DATA**

**JOBSHEET 11**



**Disusun Oleh :**

**NAJWA ELDIARA OWILIA TIKSA**

**BP/NIM : 2023/23343079**

**Dosen Pengampu :**

**Randi Proska Sandra, S.Pd, M.Sc**

**Kode Kelas : 202323430157**

**PROGRAM STUDI S1 INFORMATIKA**

**DEPARTEMEN ELEKTRONIKA**

**FAKULITAS TEKNIK**

**UNIVERSITAS NEGERI PADANG**

**2023**

1. **Source Code**

//Created by 23343079\_Najwa Eldiara Owilia Tiksa

#include <stdio.h>

#include <string.h>

// Struktur untuk menyimpan data mahasiswa

typedef struct {

int NIM;

char Nama[50];

char TTL[50];

float IPK;

} Mahasiswa;

// Fungsi untuk mencetak data mahasiswa

void cetakMahasiswa(Mahasiswa m) {

printf("NIM: %d\n", m.NIM);

printf("Nama: %s\n", m.Nama);

printf("TTL: %s\n", m.TTL);

printf("IPK: %.2f\n", m.IPK);

}

// Fungsi sequential search

int sequentialSearch(Mahasiswa mhs[], int ukuran, int kunci) {

for (int i = 0; i < ukuran; i++) {

if (mhs[i].NIM == kunci) {

return i;

}

}

return -1;

}

// Fungsi binary search (array harus terurut berdasarkan NIM)

int binarySearch(Mahasiswa mhs[], int ukuran, int kunci) {

int kiri = 0, kanan = ukuran - 1;

while (kiri <= kanan) {

int tengah = kiri + (kanan - kiri) / 2;

if (mhs[tengah].NIM == kunci) {

return tengah;

} else if (mhs[tengah].NIM < kunci) {

kiri = tengah + 1;

} else {

kanan = tengah - 1;

}

}

return -1;

}

// Fungsi untuk mengurutkan array mahasiswa berdasarkan NIM menggunakan bubble sort

void urutkanMahasiswa(Mahasiswa mhs[], int ukuran) {

for (int i = 0; i < ukuran - 1; i++) {

for (int j = 0; j < ukuran - i - 1; j++) {

if (mhs[j].NIM > mhs[j + 1].NIM) {

Mahasiswa temp = mhs[j];

mhs[j] = mhs[j + 1];

mhs[j + 1] = temp;

}

}

}

}

int main() {

int ukuran;

printf("Masukkan jumlah mahasiswa: ");

scanf("%d", &ukuran);

getchar(); // Mengkonsumsi newline character

Mahasiswa mhs[ukuran];

for (int i = 0; i < ukuran; i++) {

printf("Masukkan data mahasiswa ke-%d:\n", i + 1);

printf("NIM: ");

scanf("%d", &mhs[i].NIM);

getchar(); // Mengkonsumsi newline character

printf("Nama: ");

fgets(mhs[i].Nama, sizeof(mhs[i].Nama), stdin);

mhs[i].Nama[strcspn(mhs[i].Nama, "\n")] = '\0'; // Menghilangkan newline di akhir input

printf("TTL: ");

fgets(mhs[i].TTL, sizeof(mhs[i].TTL), stdin);

mhs[i].TTL[strcspn(mhs[i].TTL, "\n")] = '\0'; // Menghilangkan newline di akhir input

printf("IPK: ");

scanf("%f", &mhs[i].IPK);

getchar(); // Mengkonsumsi newline character

}

printf("\nData mahasiswa:\n");

for (int i = 0; i < ukuran; i++) {

cetakMahasiswa(mhs[i]);

printf("\n");

}

// Pencarian data

int kunci;

printf("Masukkan NIM untuk pencarian: ");

scanf("%d", &kunci);

// Sequential Search

int indeksSeq = sequentialSearch(mhs, ukuran, kunci);

if (indeksSeq != -1) {

printf("\nHasil pencarian (Sequential Search):\n");

cetakMahasiswa(mhs[indeksSeq]);

} else {

printf("\nMahasiswa dengan NIM %d tidak ditemukan (Sequential Search).\n", kunci);

}

// Mengurutkan data mahasiswa berdasarkan NIM untuk Binary Search

urutkanMahasiswa(mhs, ukuran);

// Binary Search

int indeksBin = binarySearch(mhs, ukuran, kunci);

if (indeksBin != -1) {

printf("\nHasil pencarian (Binary Search):\n");

cetakMahasiswa(mhs[indeksBin]);

} else {

printf("\nMahasiswa dengan NIM %d tidak ditemukan (Binary Search).\n", kunci);

}

return 0;

}

1. **Penjelasan Program**

Program ini adalah sebuah aplikasi berbasis C yang digunakan untuk menyimpan, mengurutkan, dan mencari data mahasiswa. Program ini memanfaatkan beberapa algoritma dan fungsi untuk mencapai tujuannya.Algoritma yang Digunakan

Sequential Search (Pencarian Linear)

Algoritma ini mencari elemen dengan memeriksa setiap elemen dalam array satu per satu hingga elemen yang dicari ditemukan atau akhir array tercapai.

Kompleksitas waktu: O(n), di mana n adalah jumlah elemen dalam array.

Bubble Sort

Algoritma ini mengurutkan array dengan membandingkan setiap elemen dengan elemen berikutnya dan menukar mereka jika urutannya salah. Proses ini diulang hingga array benar-benar terurut.

Kompleksitas waktu: O(n^2) dalam kasus terburuk, di mana n adalah jumlah elemen dalam array.

Binary Search (Pencarian Biner)

Algoritma ini mencari elemen dengan membagi array yang sudah terurut menjadi dua bagian secara berulang dan memeriksa apakah elemen yang dicari berada di bagian kiri atau kanan. Proses ini diulang pada subarray yang sesuai hingga elemen ditemukan atau subarray menjadi kosong.

Kompleksitas waktu: O(log n), di mana n adalah jumlah elemen dalam array.

Proses Utama dalam Program

Input Data Mahasiswa: Program meminta pengguna untuk memasukkan jumlah mahasiswa dan kemudian data masing-masing mahasiswa.

Mencetak Data Mahasiswa: Program mencetak semua data mahasiswa yang sudah dimasukkan.

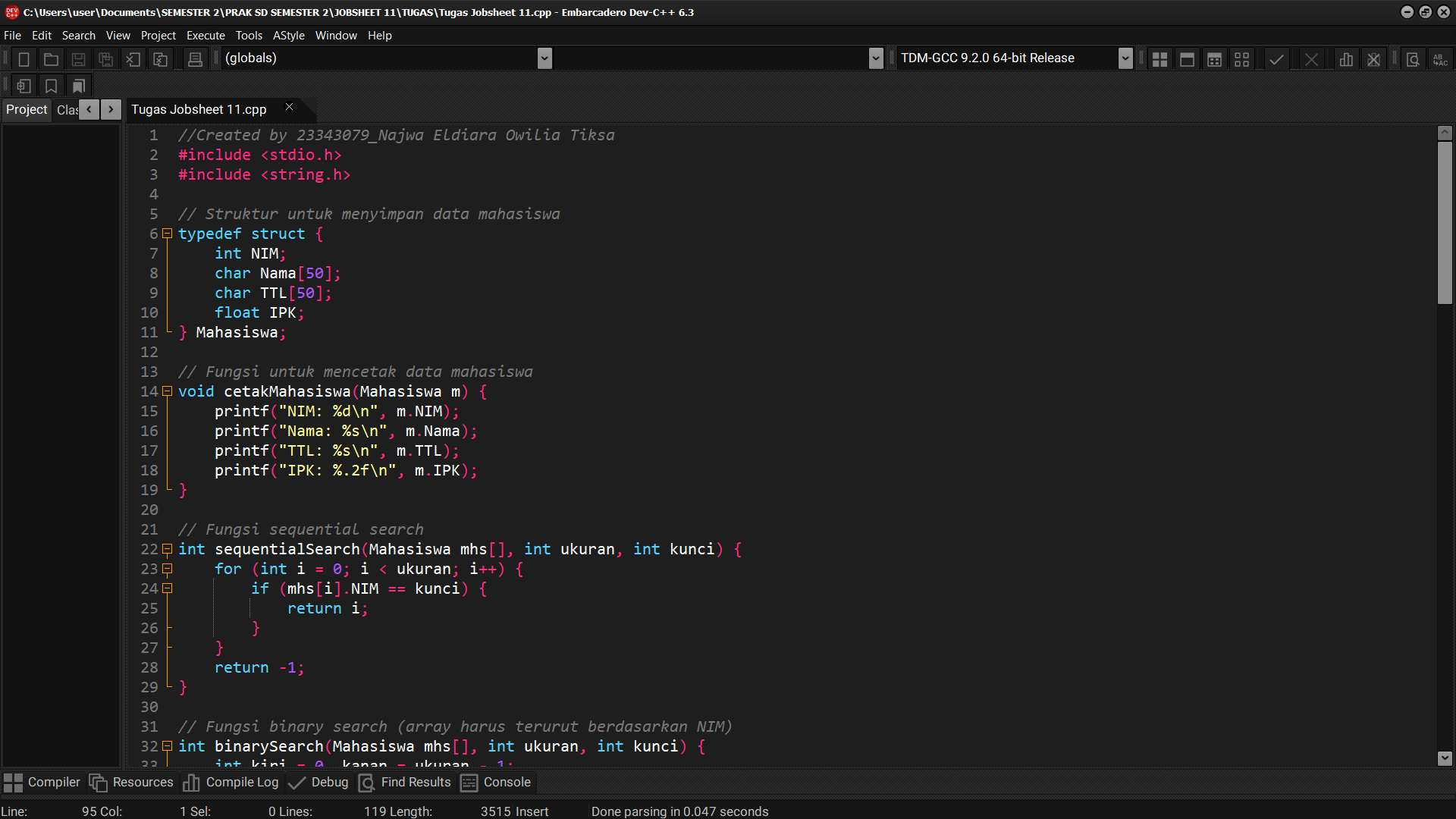
Pencarian Data dengan Sequential Search: Program meminta pengguna untuk memasukkan NIM mahasiswa yang ingin dicari, dan kemudian menggunakan sequential search untuk mencari mahasiswa tersebut.

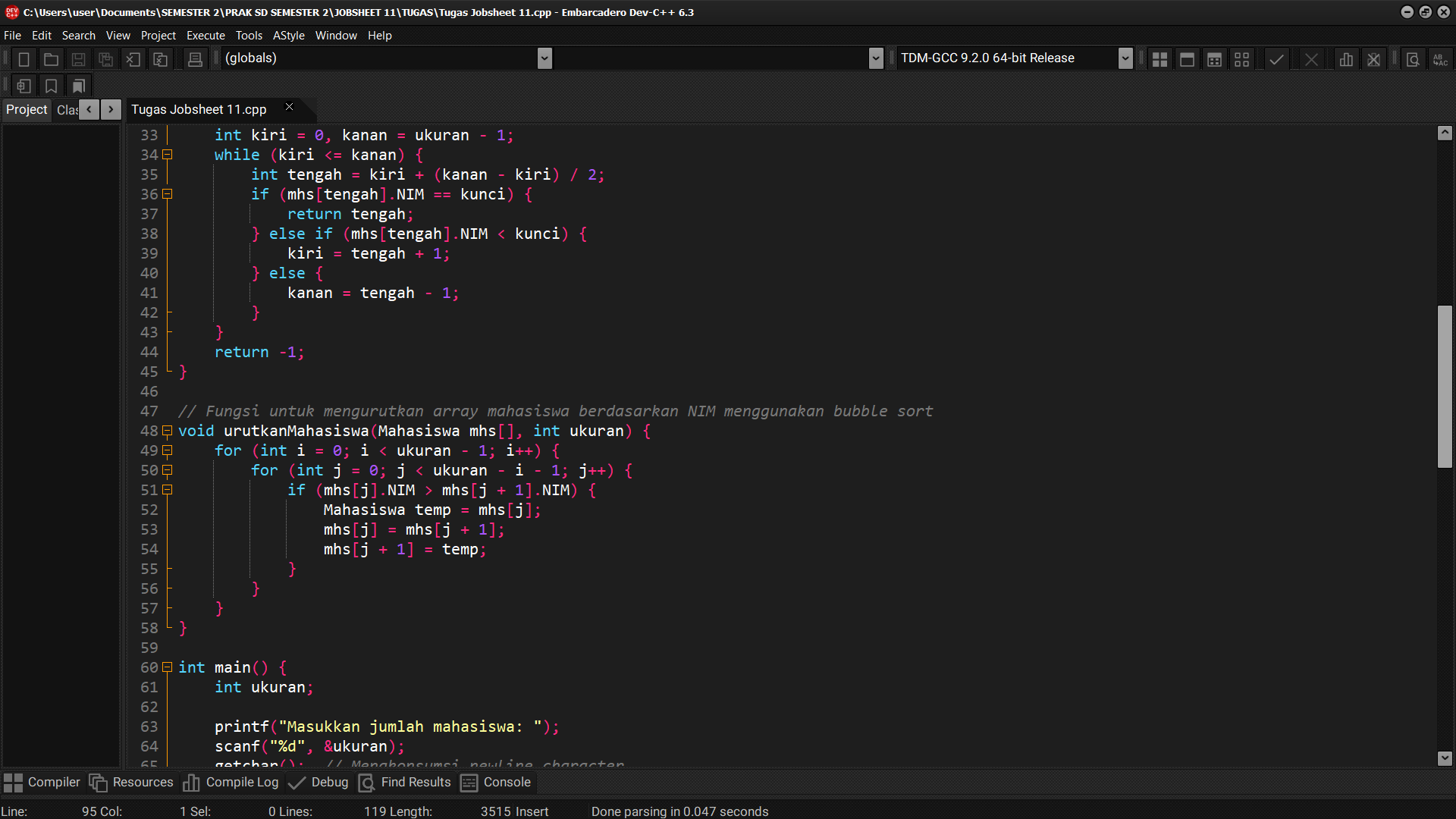
Mengurutkan Data dengan Bubble Sort: Program mengurutkan data mahasiswa berdasarkan NIM menggunakan bubble sort untuk persiapan pencarian dengan binary search.

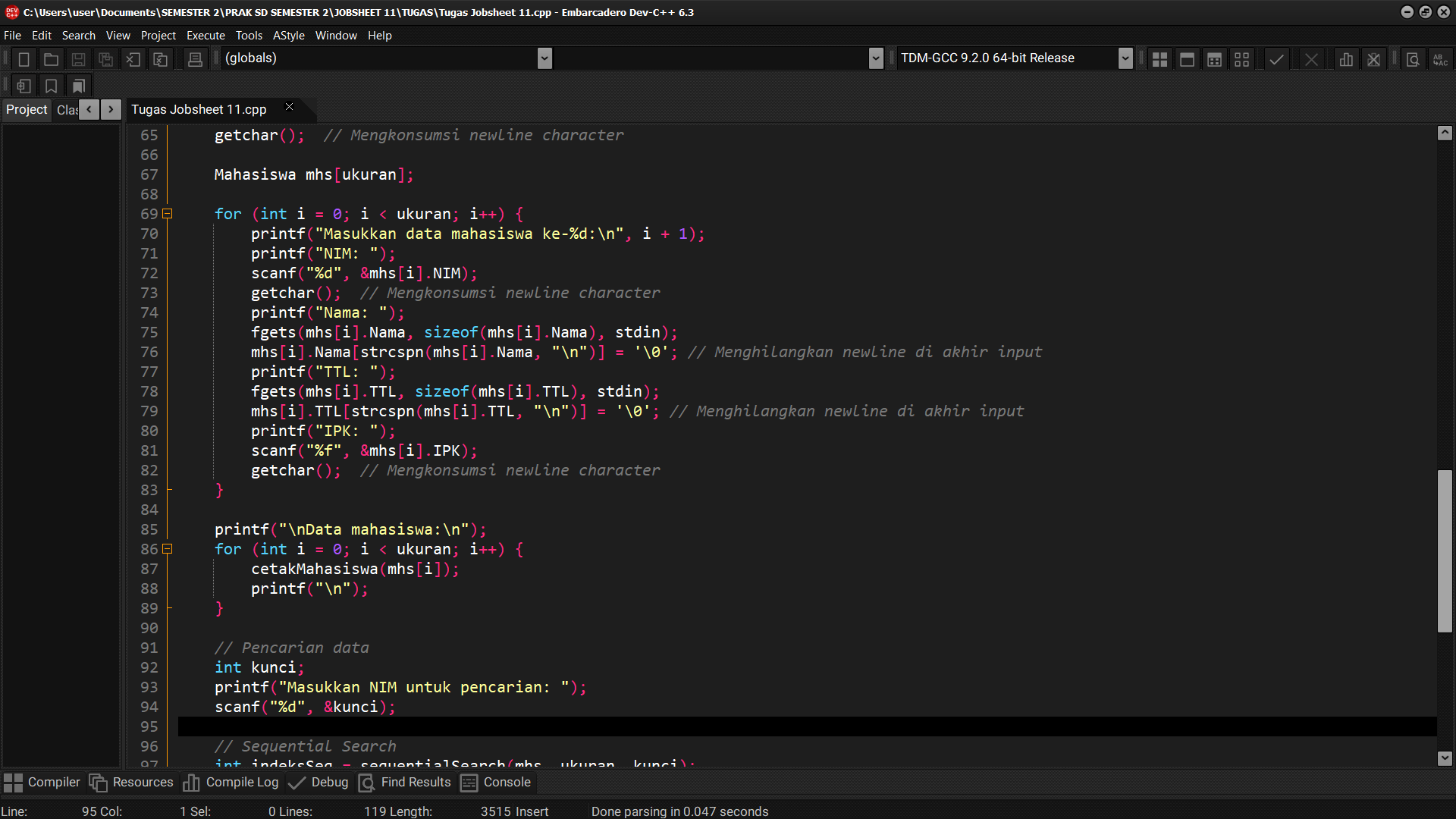
Pencarian Data dengan Binary Search: Setelah data terurut, program melakukan pencarian data mahasiswa berdasarkan NIM menggunakan binary search.

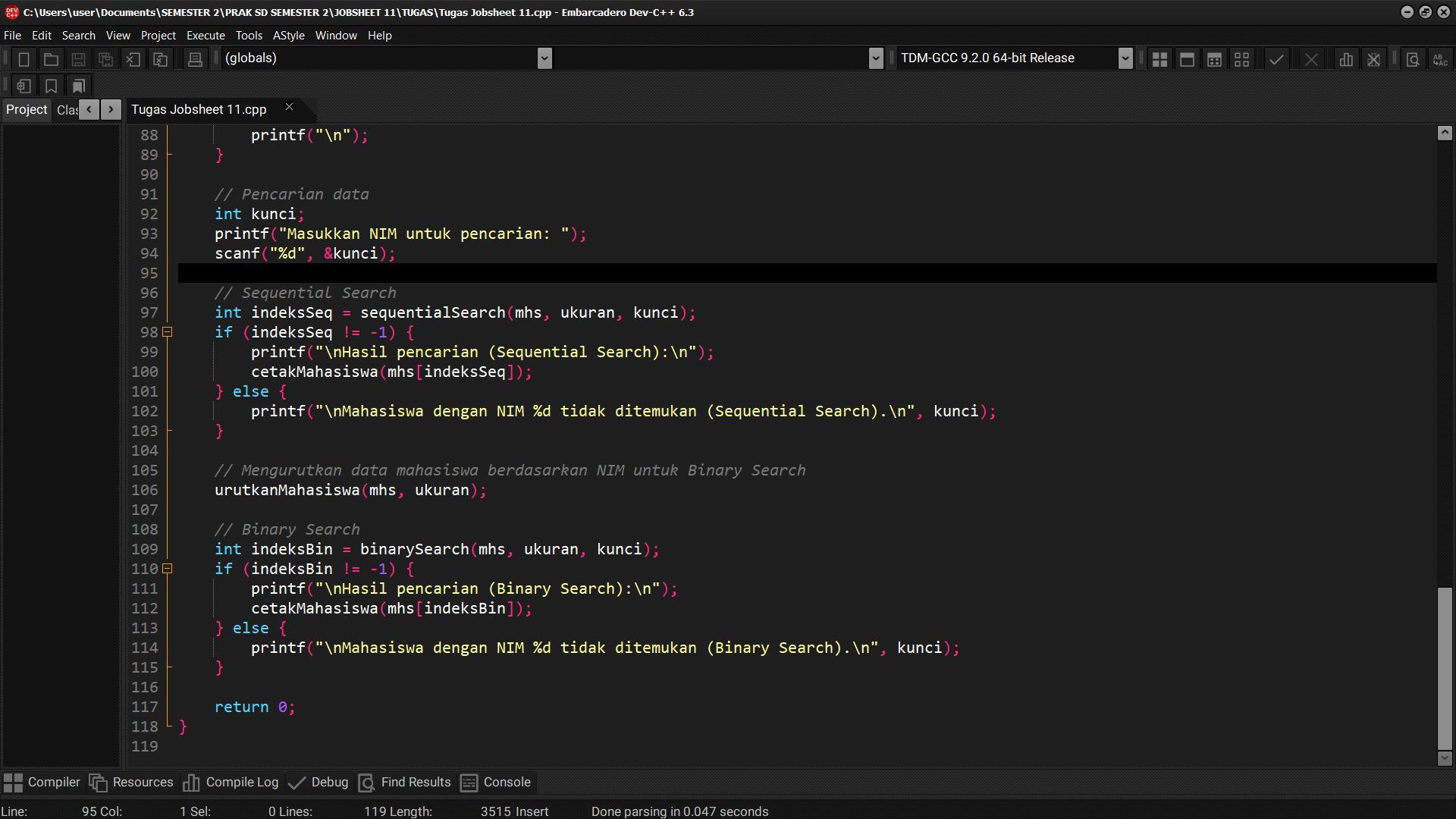
Program ini memanfaatkan algoritma dasar pencarian dan pengurutan untuk memanipulasi dan mencari data mahasiswa. Algoritma sequential search digunakan untuk pencarian tanpa pengurutan, sementara bubble sort digunakan untuk mengurutkan data sebelum melakukan pencarian lebih efisien dengan binary search.

1. **Screenshot program**









1. **Output**

