**USER MANUAL**

**PROGRAM *EDIT DISTANCE***



Oleh:

Dedy Van Hauten 1400910039

Farouq Fasial Anam 1400910045

Richard Nathaniel Chandra 1400910015

**PROGRAM STUDI HUMAN-COMPUTER INTERACTION**

**FAKULTAS ILMU HAYATI**

**UNIVERSITAS SURYA**

**SERPONG**

**2015**

**KATA PENGANTAR**

Puji syukur kami panjatkan ke hadirat Tuhan Yang Maha Esa atas kasih- Nya yang besar sehingga kami dapat menyelesaikan program *Edit distance* ini dengan baik dan tepat waktu. Pertama kami mengucapkan terima kasih kepada Ibu Jelita Asian, Ph.D, selaku dosen mata kuliah pemrograman lanjut, atas bimbingan yang diberikan selama kami mengerjakan program ini. Tidak lupa juga kami mengucapkan terima kasih kepada teman-teman dan keluarga yang mendukung kami selama kami menyelesaikan program ini.

Program ini kami buat untuk memenuhi tugas pertama dari mata kuliah Algoritma dan Struktur Data. Banyak hal positif yang kami dapatkan selama mengerjakan program ini. Melalui tugas ini, kami telah belajar lebih dalam lagi mengenai algoritma *Edit distance.*

Akhir kata, kami berharap program ini dapat berguna bagi banyak orang sebagai karya orisinil anak bangsa, demi membangun Indonesia jaya.

Tangerang, 2 Desember 2015

Penyusun

**DAFTAR ISI**

Cover Depan 1

Kata Pengantar 2

Daftar Isi 3

Bab I Pendahuluan 4

1. *Edit distance* 4
2. Program Secara Umum 5
3. Keistimewaan Program 5

Bab II Petunjuk Praktis Pemakaian 6

1. Cara *Compile* 6

A.1. Cara *Compile* Dengan Command Prompt 6

A.2. Cara *Compile* Dengan Dev C++ 6

1. Menu Utama 8
2. Program Berjalan 9
3. *Error Checking* 11

D.1. *Input* Angka Negatif 11

D.2. *Input* Karakter 11

D.3. *Input* Karakter dan Angka 12

D.4. *Input* Spesial Karakter 12

1. Request Program 13

D.1. Jika *Input* Program Salah 13

D.2. Jika *Input* Program Benar 13

Daftar Pustaka 14

**BAB I**

**PENDAHULUAN**

1. *Edit distance*

*Edit distance* adalah sebuah algoritma untuk mengukur jarak perbedaan antara dua tipe data string dengan cara menghitung angka operasi minimum yang dibutuhkan untuk mengubah string yang menjadi string lainnya yang dibandingkan. *Edit distance* biasa lebih dikenal dengan nama Levenshtein Distance yang diambil dari nama penemunya, yaitu Vladimir Levenshtein, yang menemukan *Edit distance* pada tahun 1965.

Kegiatan *Edit distance* antara lain mengganti karakter, menambah karakter, atau menghapus karakter.

1. Mengganti karakter:

String ‘commuter’ dan ‘computer’, dengan mengubah huruf m pada string 1 menjadi huruf p.

1. Menambah karakter

String ‘sort’ dan ‘sport’, dengan menambahkan huruf p pada string 1.

1. Menghapus karakter

String ‘sport’ dan ‘sort’, dengan menghilangkan huruf p pada string 1.

B. Program Secara Umum

Program *Edit distance* merupakan sebuah program yang bertujuan untuk mencari dan mengurutkan semua kata dalam file *input* yang memiliki *edit distance* tertentu dengan kata yang kita inginkan. Program akan meng-upload file *input* txt terlebih dahulu, kemudian string dalam kata tersebut diubah menjadi huruf kecil semua dan menghilangkan semua tanda baca, sehingga hanya meninggalkan angka 0-9, huruf a-z, dan tanda ‘. Selanjutnya, program akan menghitung frekuensi kata-kata dan disimpan di array. Setelah itu, program akan meminta user memasukkan kata yang diinginkan dan nilai *edit distance* maksimum. Program akan menampilkan dan mengurutkan kata pada file *input* yang memiliki *edit distance* tertentu dari kata yang dimasukkan oleh user.

C. Keistimewaan Program

Program ini tidak membutuhkan *memory* yang banyak. Selain itu, program ini juga memiliki code yang mudah dipahami sehingga mudah di-maintenance. Dengan demikian, *programmer* manapun dapat melakukan *maintenance* secara berkala pada program ini. Kompleksitas program ini pun hanya O(n^2) jauh lebih cepat dibanding program sejenisnya. Program ini juga memiliki *user interface* yang rapi dan menarik sehingga nyaman dan mudah digunakan.

**BAB II**

**PETUNJUK PRAKTIS PEMAKAIAN**

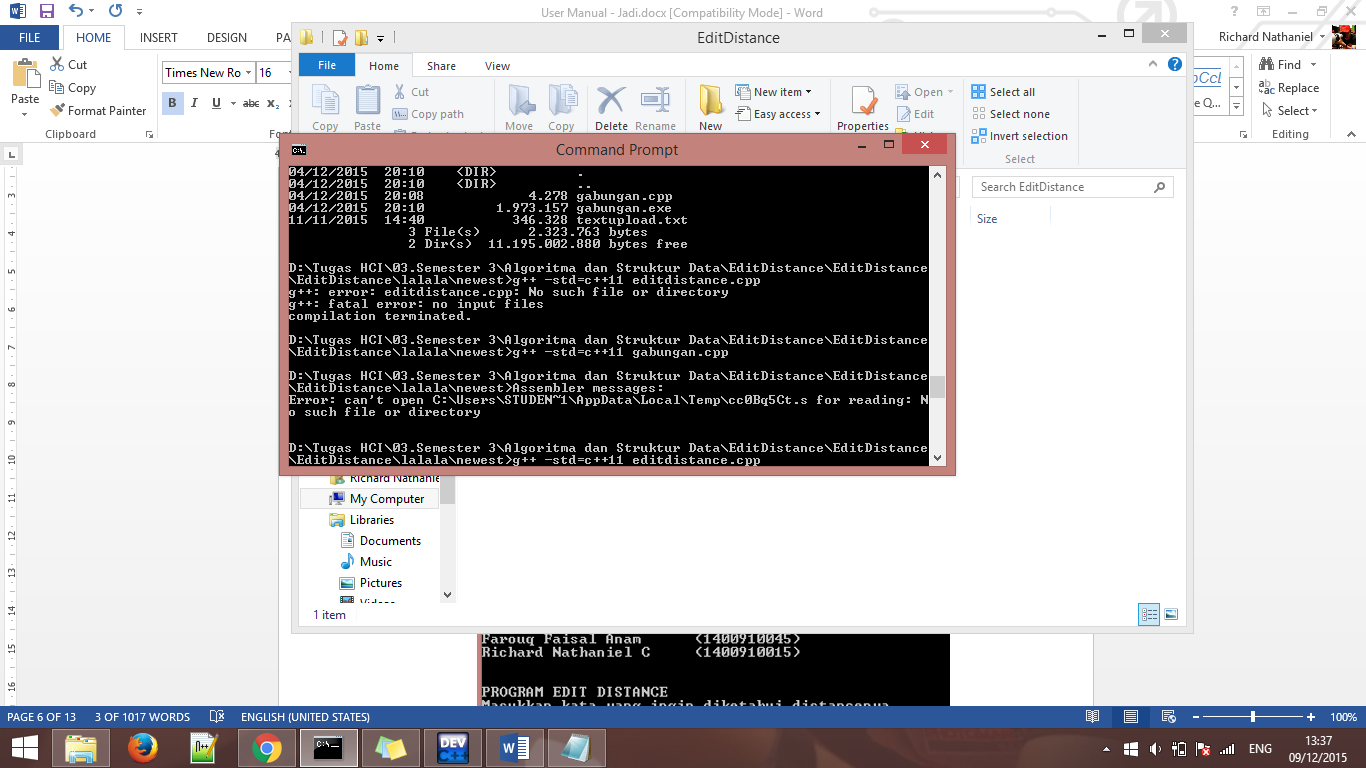
1. Cara *Compile*

Algoritma yang digunakan pada Program *Edit distance* memerlukan *compiler* GNU C++ 11. Oleh karena itu, ada beberapa cara untuk melakukan *compile* program ini, yaitu dengan Command Prompt atau Dev C++.

A.1. *Compile* Dengan Command Prompt

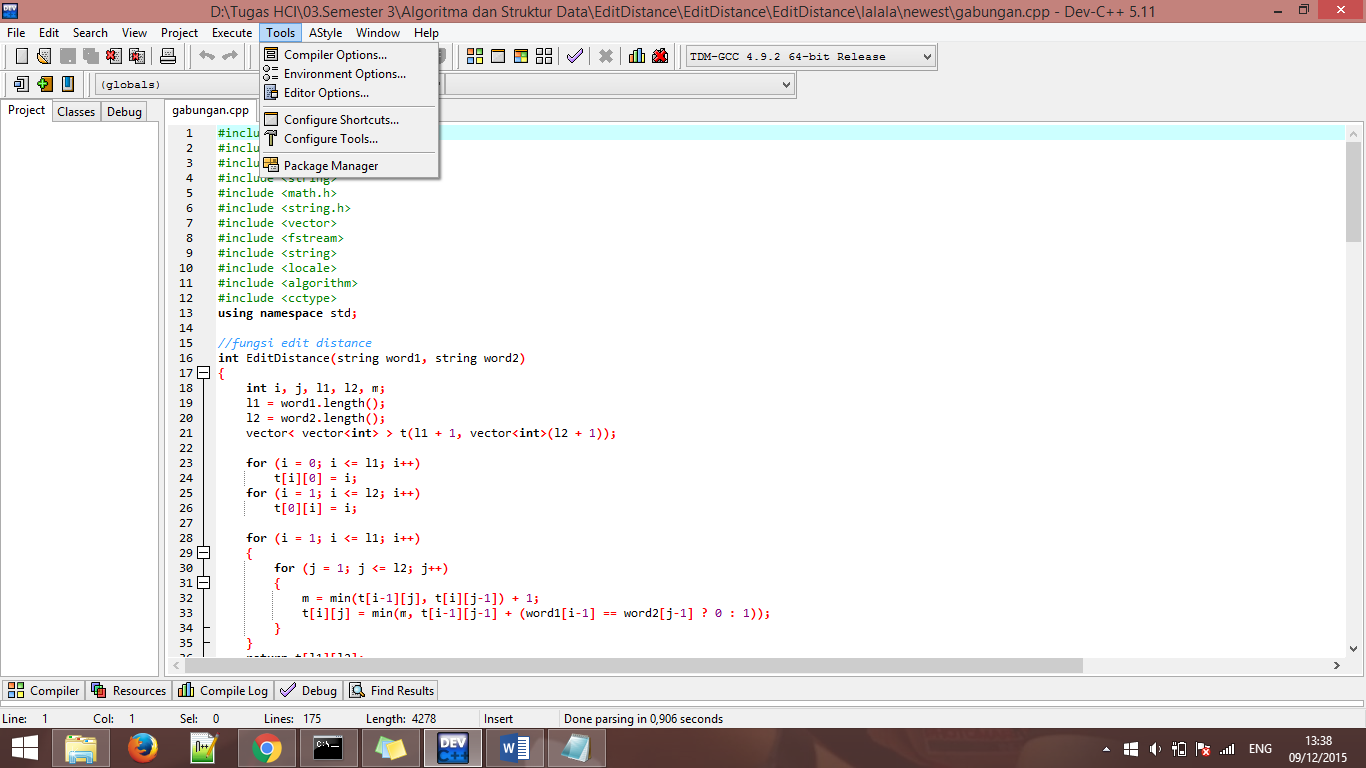
Panggil dengan cara:

**g++ -std=c++11** *<namaprogram>.***cpp**

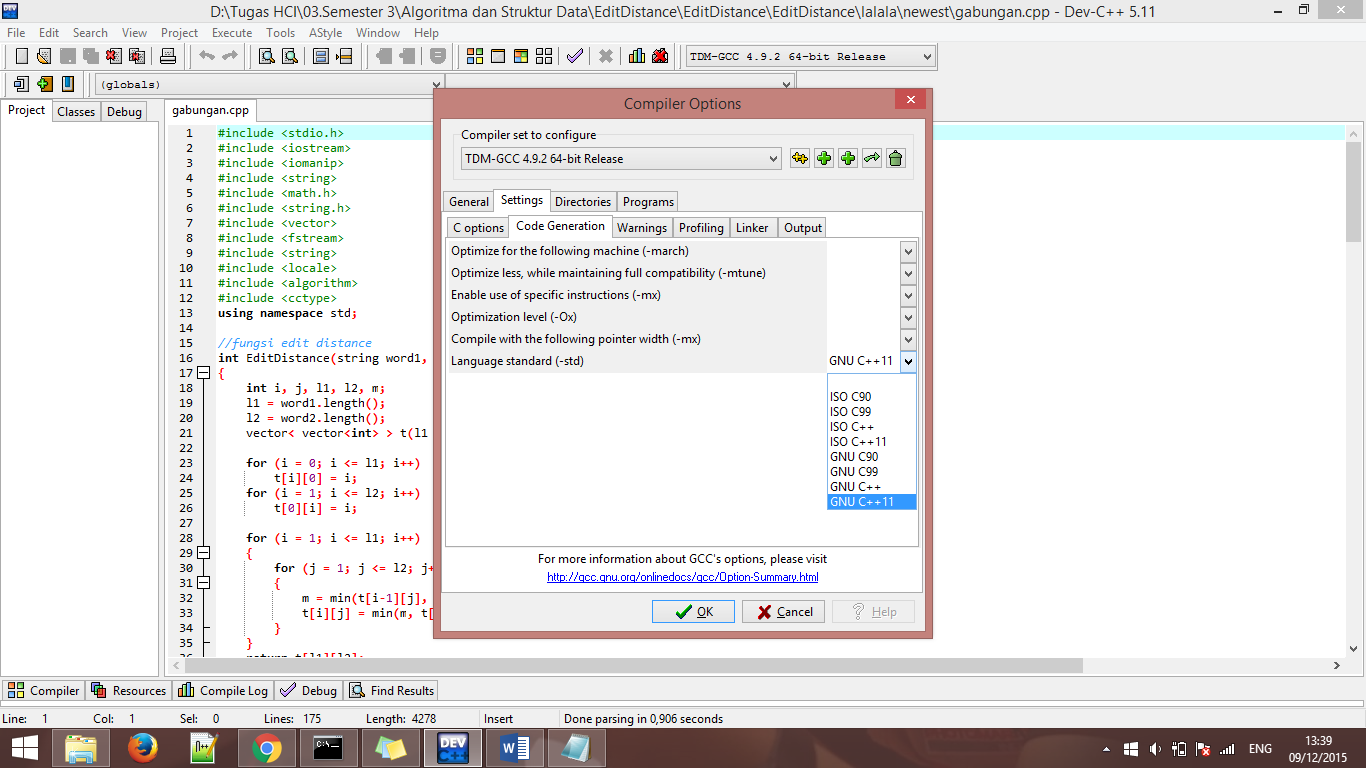


A.2. *Compile* Dengan Dev C++

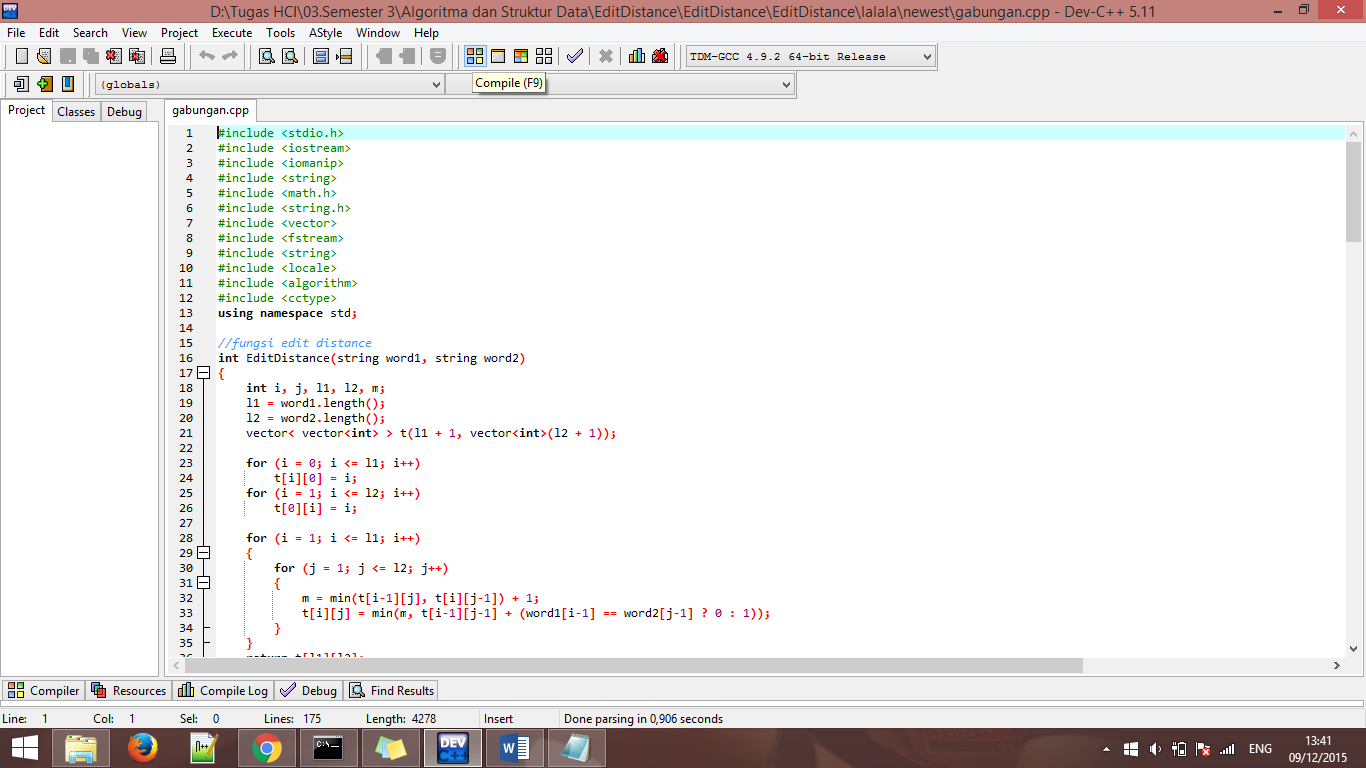
Pilih menu **Tools** > **Compiler Options**.



Pilih **Settings** > **Code Generation** **> Language standard (-std)** > **GNU C++ 11**

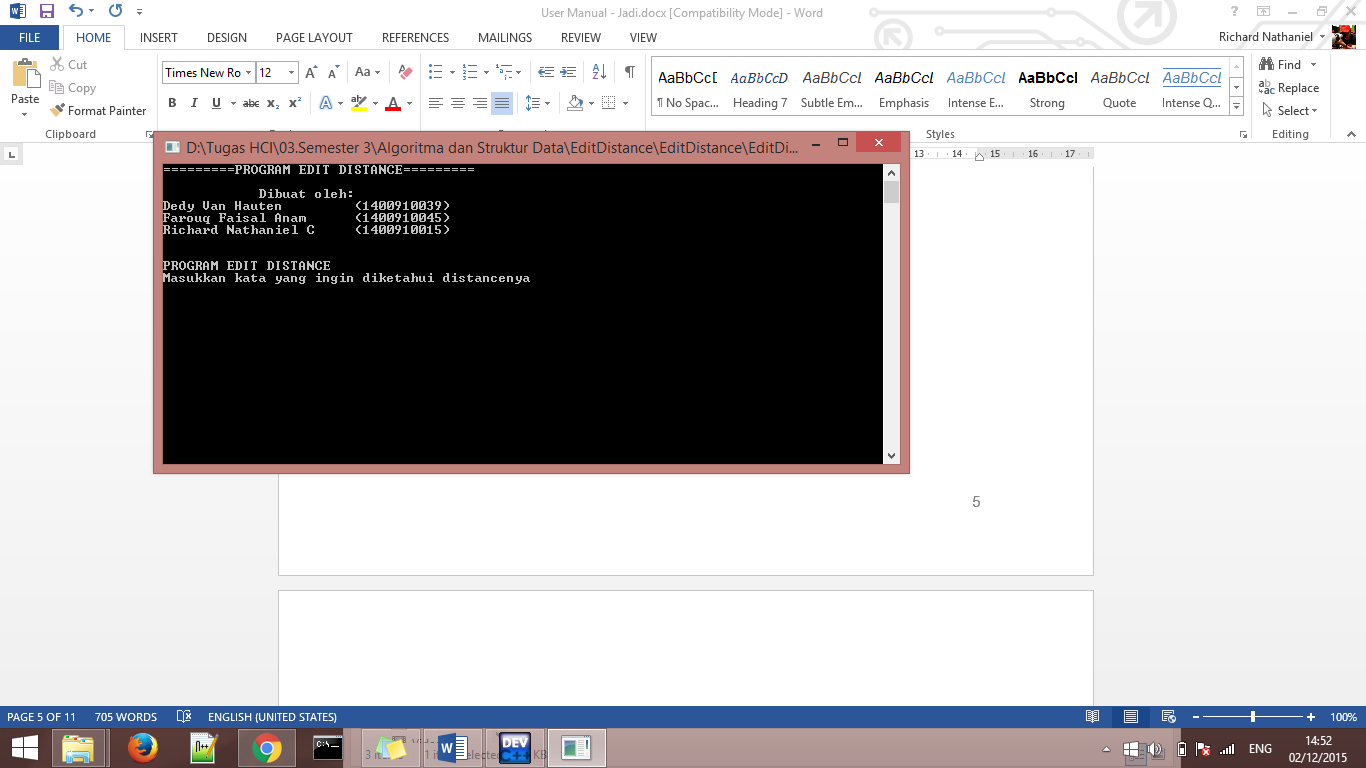


Kemudian klik OK.

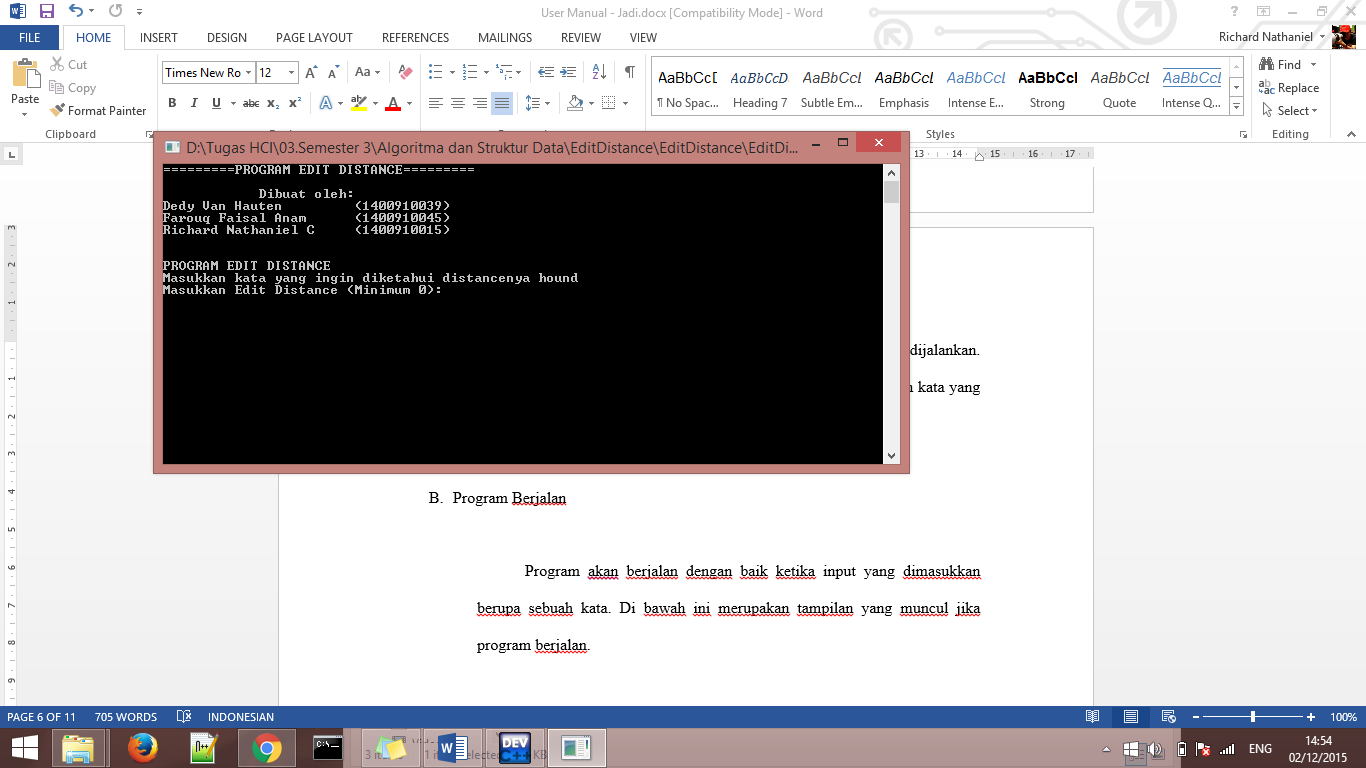


Klik menu **Compile** untuk meng-*compile* program, atau tekan F9.

1. Menu Utama



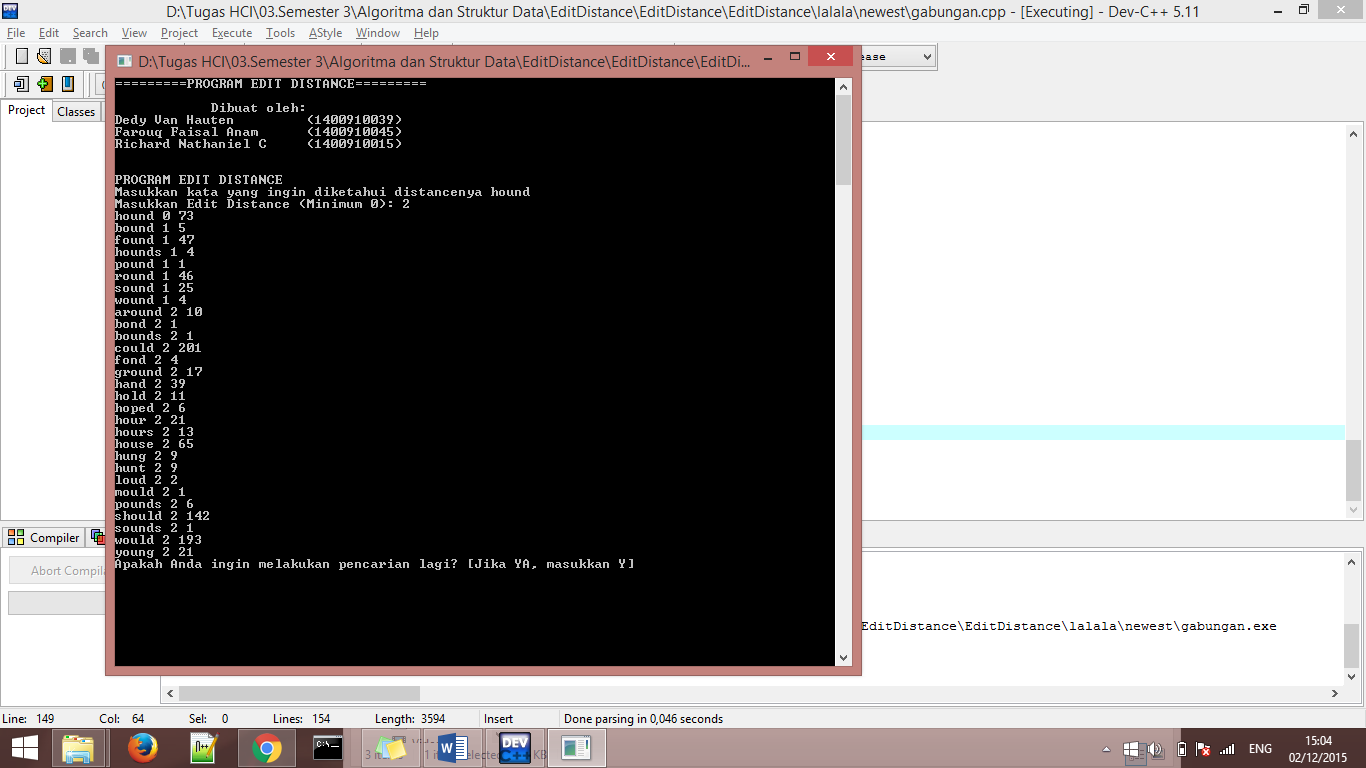
Ketika program dijalankan, program akan terlebih dahulu membaca file *input* dalam *folder* yang sama dengan program. Setelah itu, program akan menampilkan menu utama. Menu Utama merupakan tampilan awal ketika program dijalankan. Dalam Menu Utama ini, program meminta kita untuk memasukkan kata yang ingin diketahui *distance*-nya.



Setelah itu, program akan meminta memasukkan jarak *Edit distance* maksimum yang diinginkan.

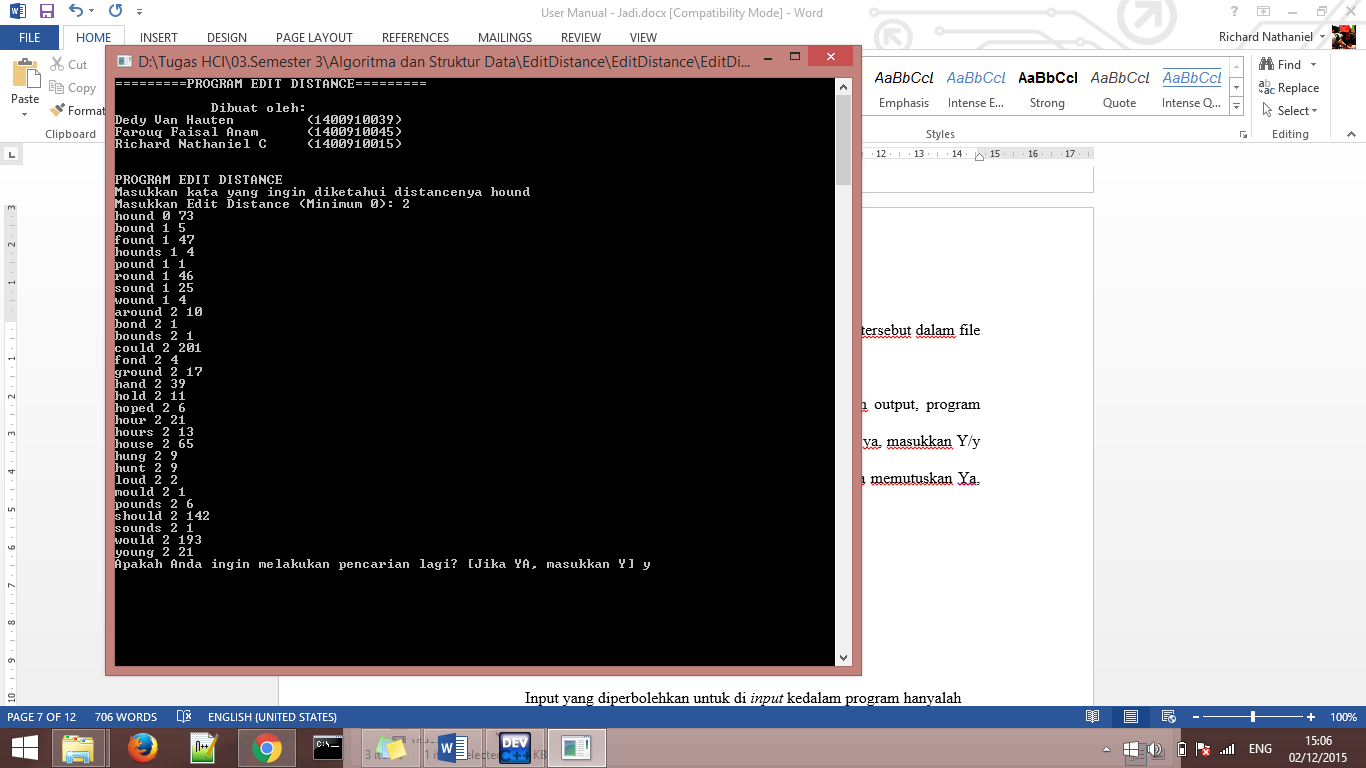
1. Program Berjalan

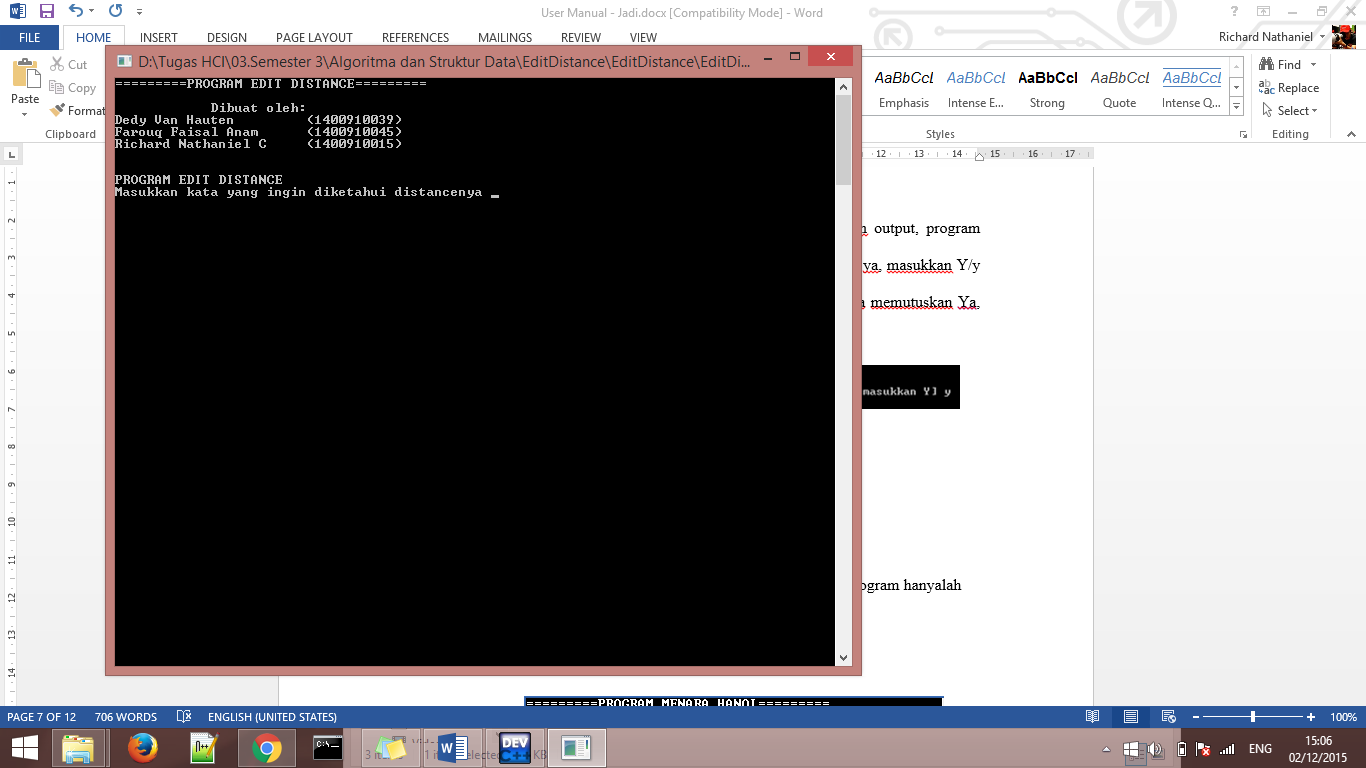
Program akan berjalan dengan baik ketika *input* kata yang dimasukkan berupa string dan *input* jarak *edit distance* yang dimasukkan berupa angka. Di bawah ini merupakan tampilan yang muncul jika program berjalan.



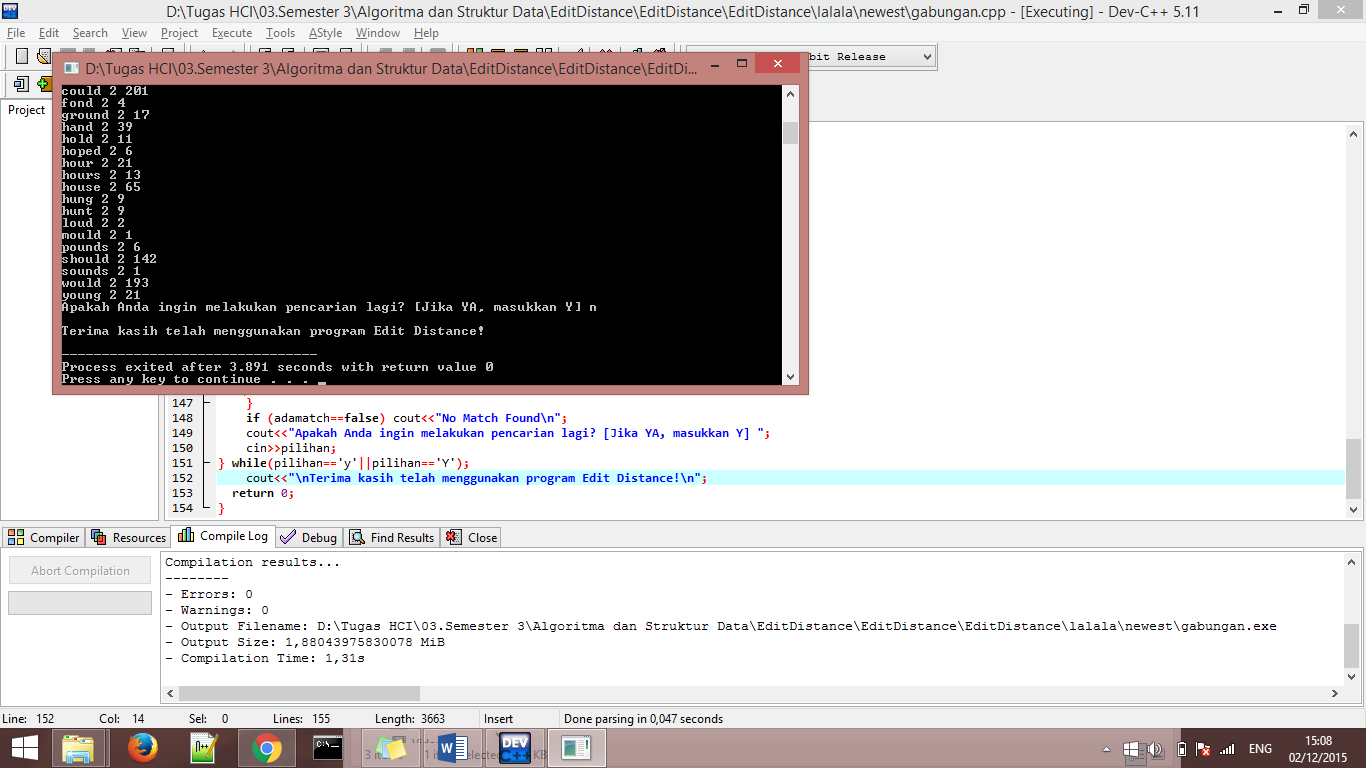
Format bentuk *output* adalah kolom pertama menunjukkan kata yang memiliki jarak *edit distance* tertentu dengan kata yang dimasukkan ke *input*, kolom kedua adalah jarak *edit distance* kata tersebut dengan kata yang dimasukkan ke *input*, dan kolom ketiga adalah jumlah kata tersebut / frekuensi dalam file *input*.

Setelah menyelesaikan pencarian dan menampilkan *output*, program akan bertanya apakah ingin melakukan pencarian lagi. Jika ya, masukkan Y/y lalu enter. Jika tidak, tekan tombol lain lalu enter. Jika kita memutuskan Ya, maka program akan kembali lagi ke awal.

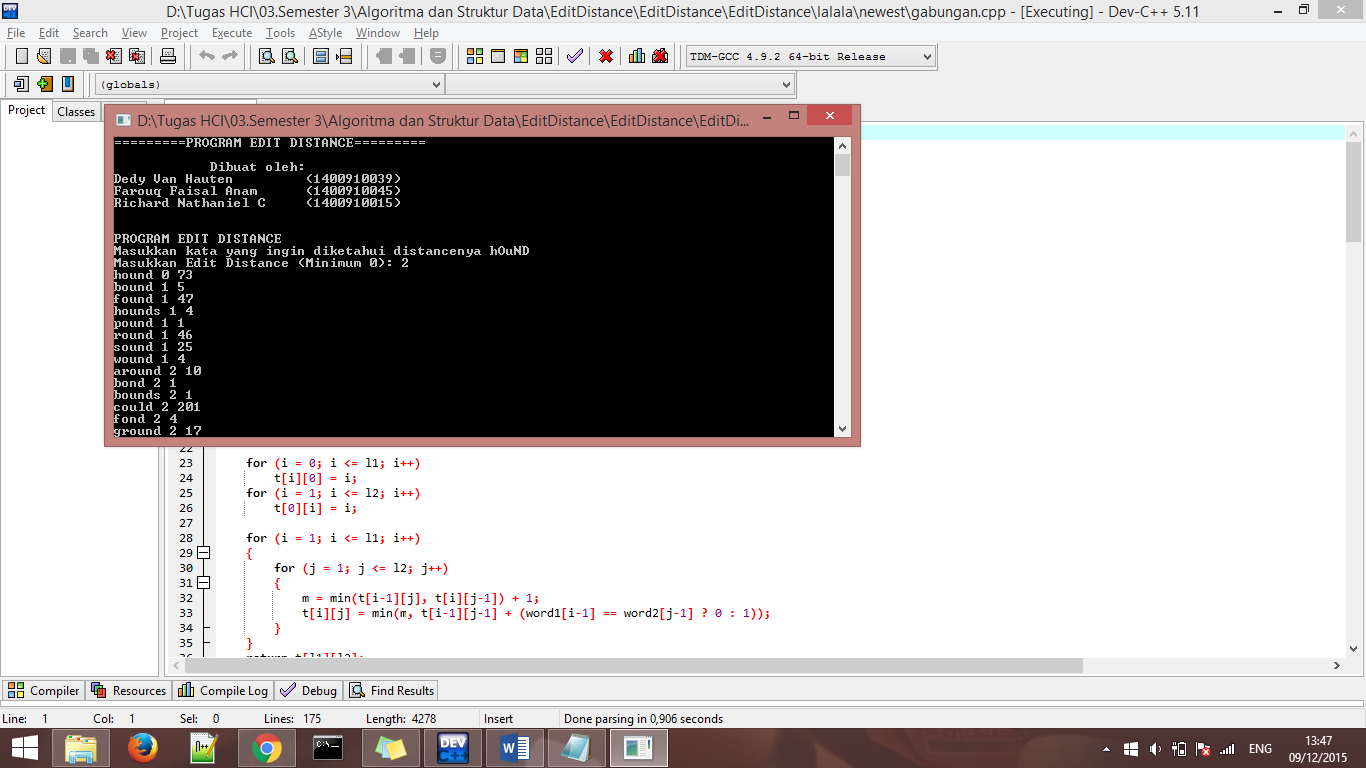




Jika kita memutuskan tidak, program akan keluar.



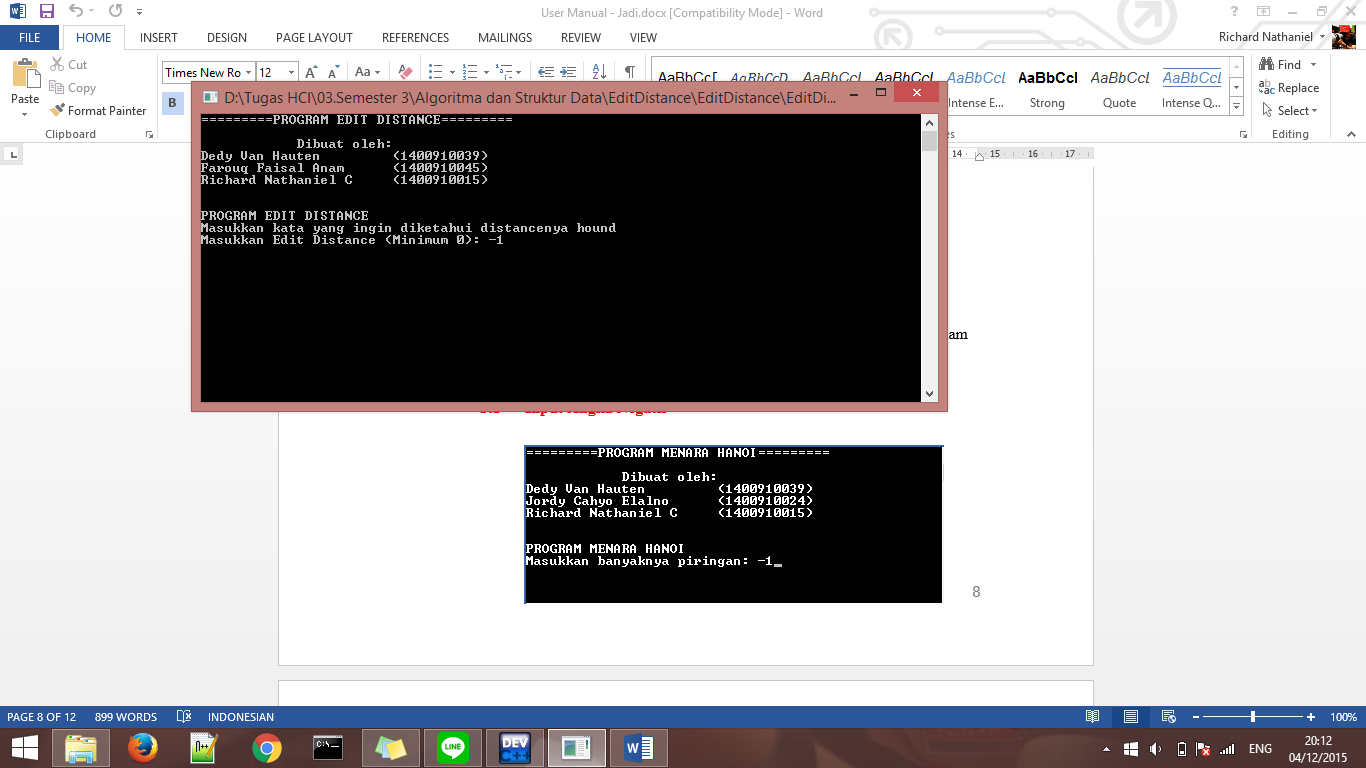
Kata yang dimasukkan dalam huruf besar secara otomatis akan diubah menjadi huruf kecil untuk menyesuaikan dengan program.



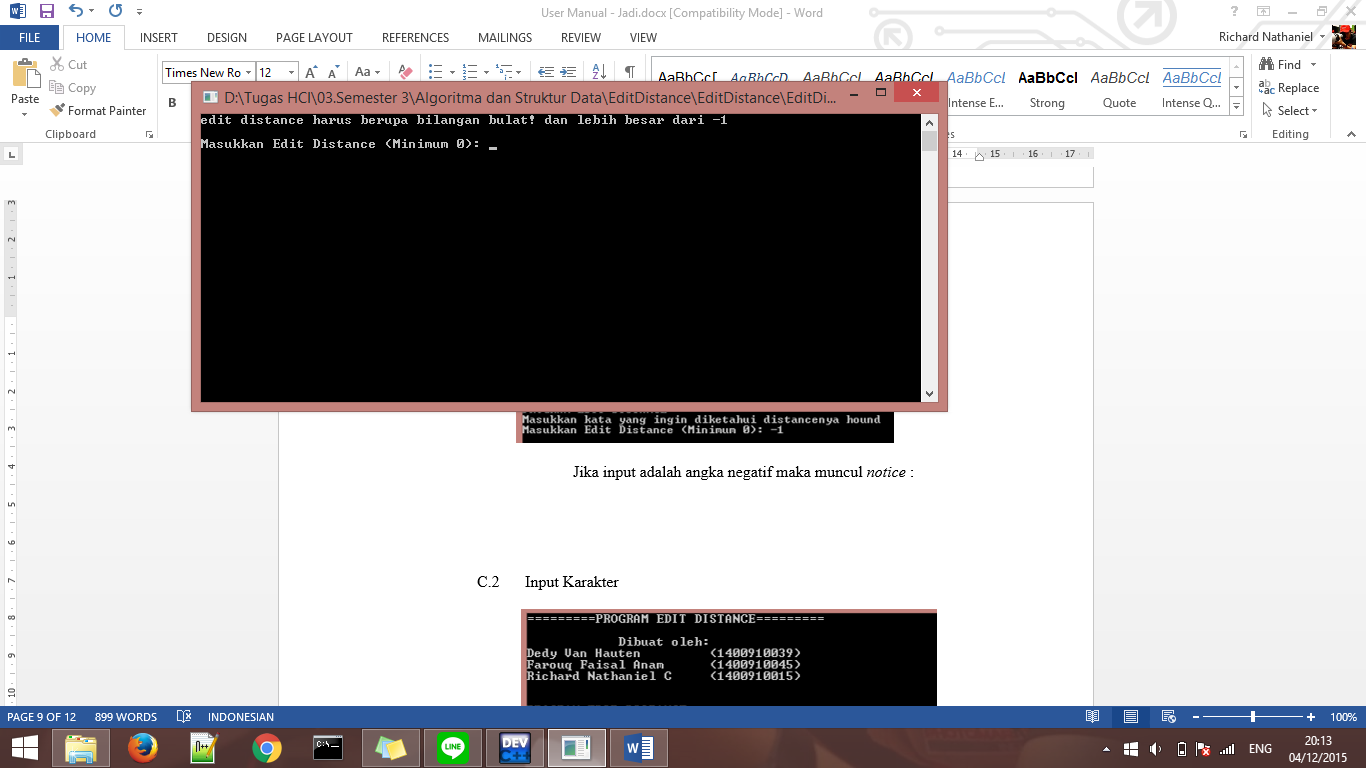
1. *Error Checking*

*Input* jarak *Edit distance* yang diperbolehkan untuk di *input* kedalam program hanyalah angka dan angka tersebut tidak boleh negatif.

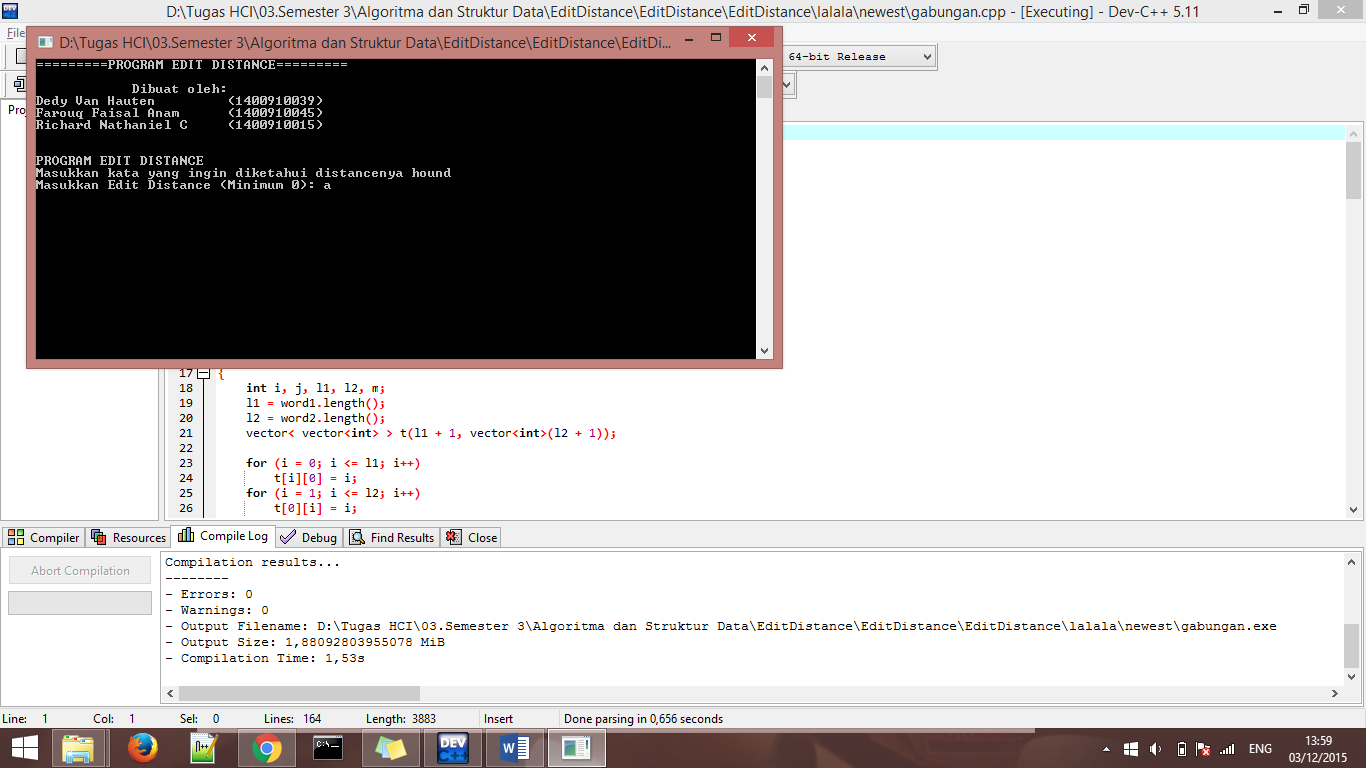
D.1. *Input* Angka Negatif



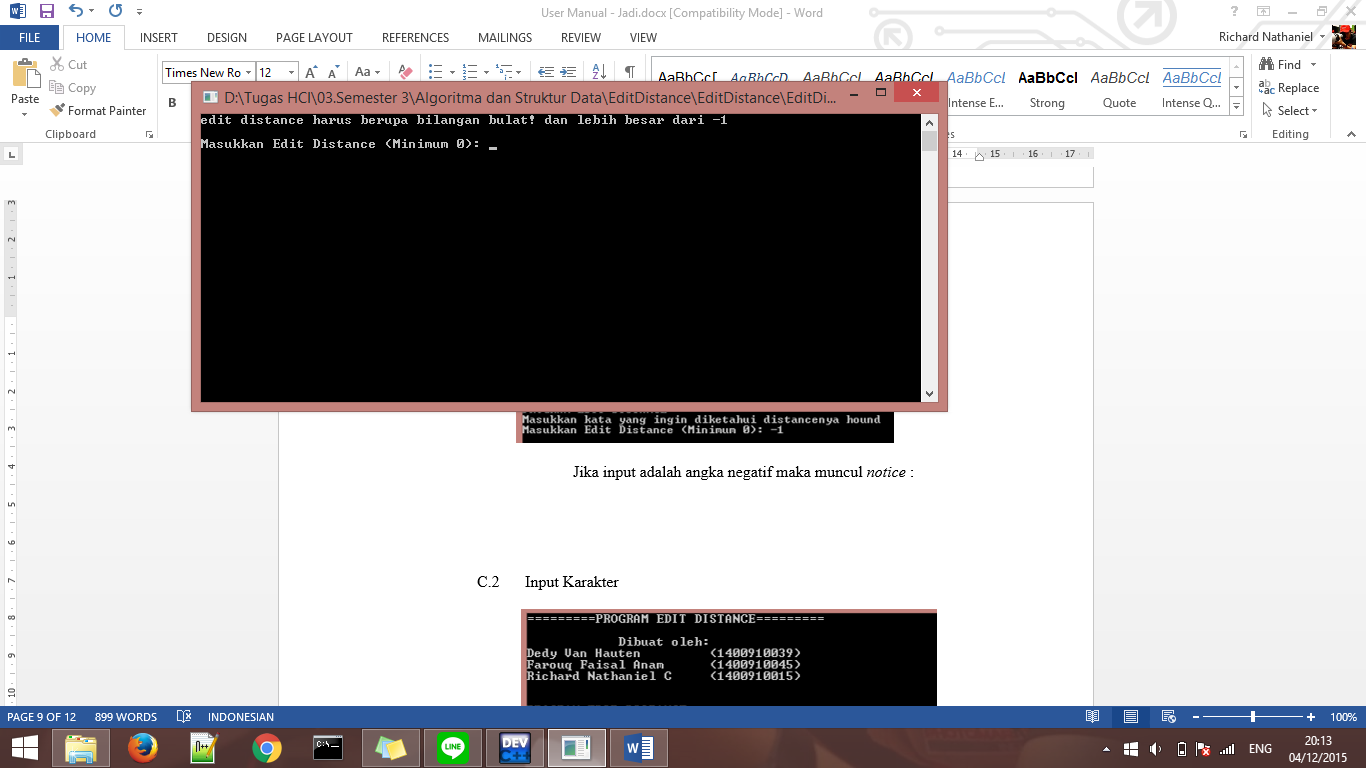
Jika *input* adalah angka negatif maka muncul *notice* :



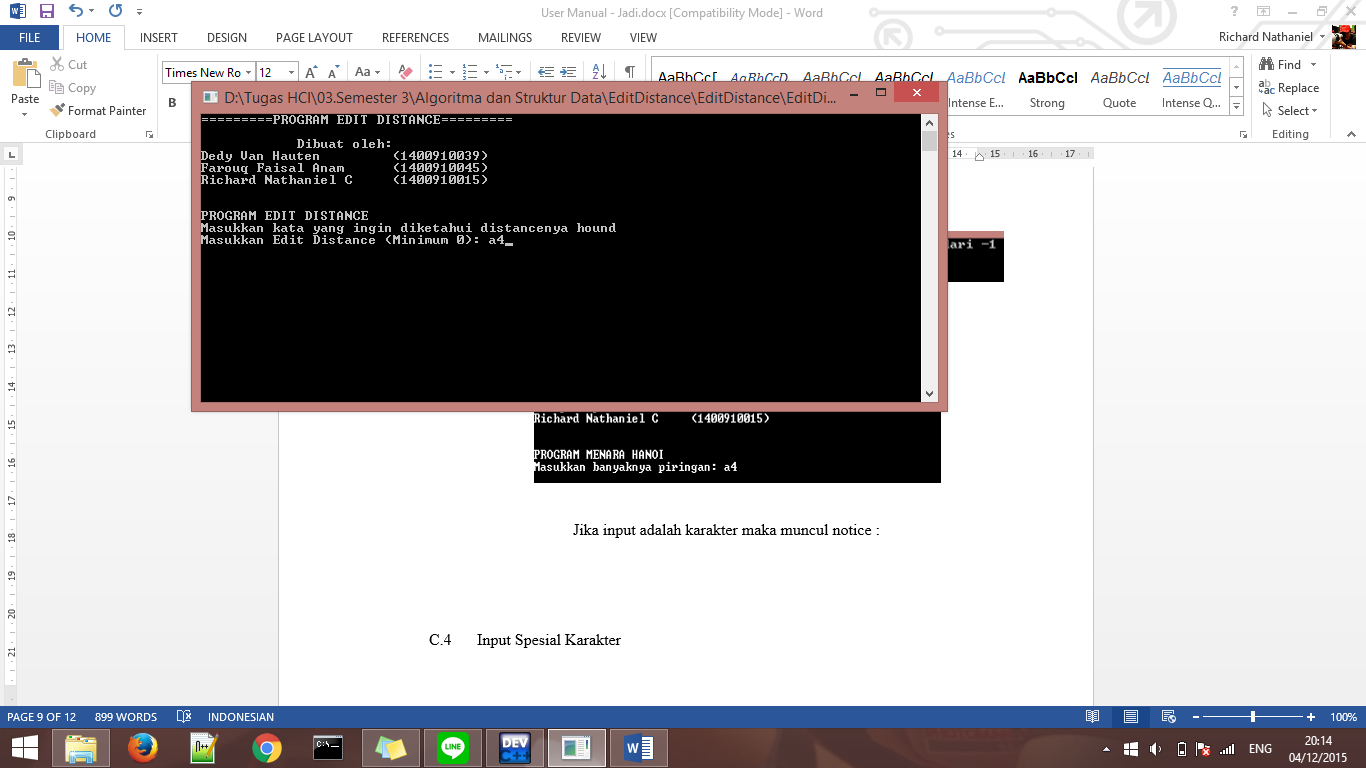
D.2 *Input* Karakter



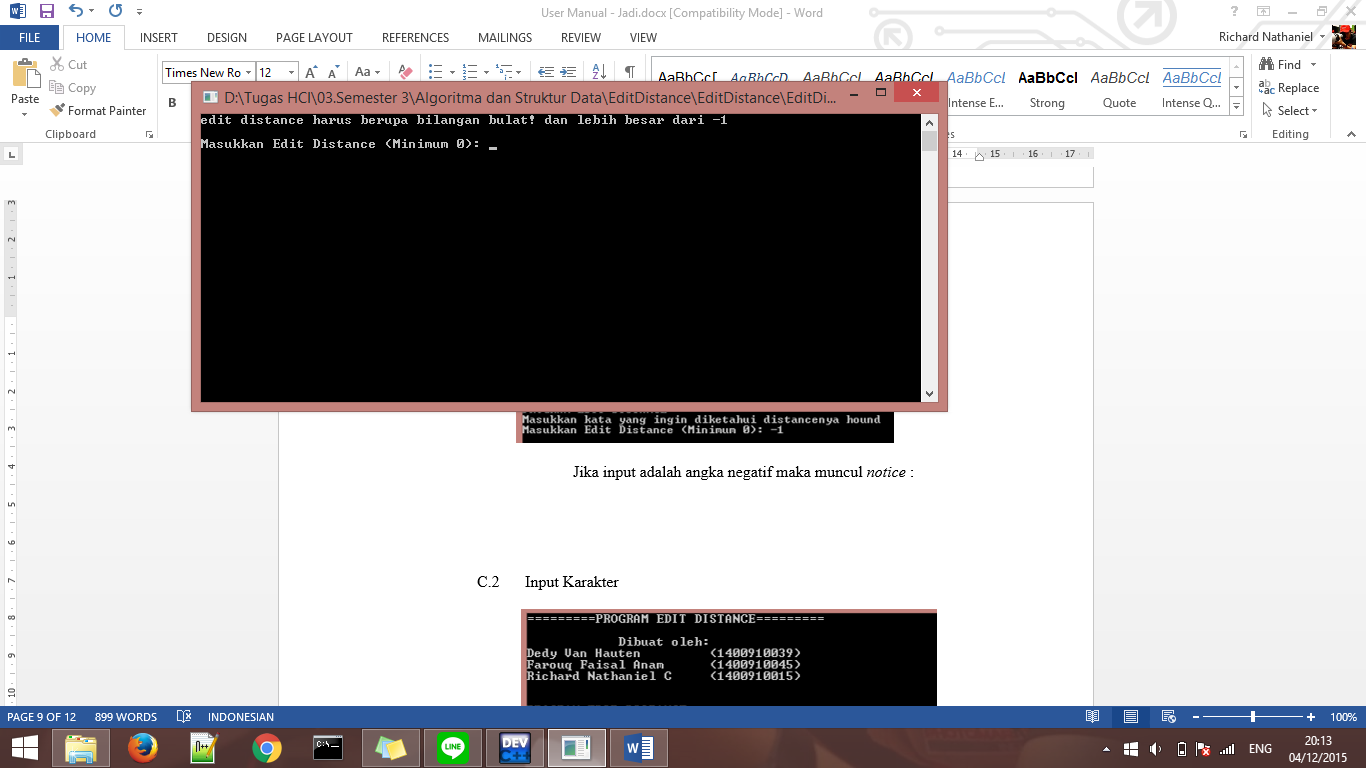
Jika *input* adalah karakter maka muncul *notice* :



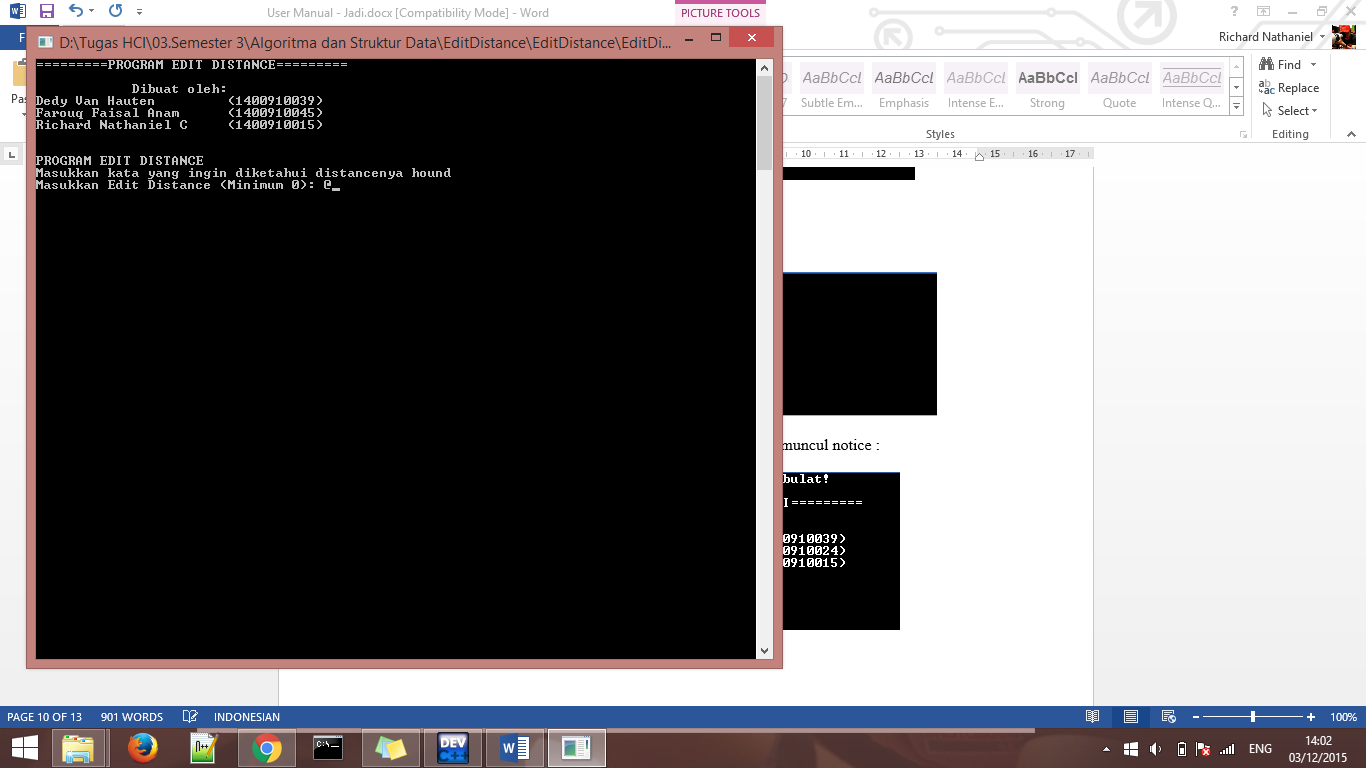
D.3. *Input* Karakter dan Angka



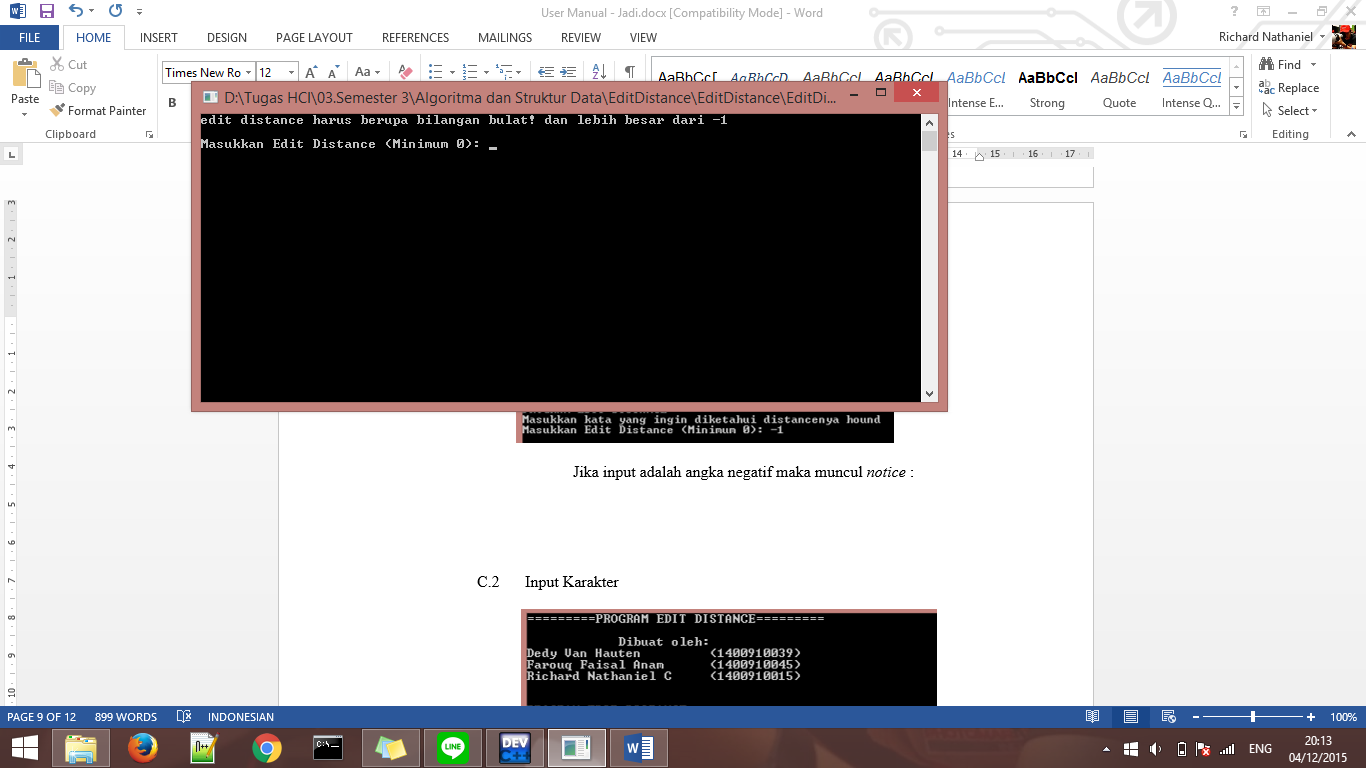
Jika *input* adalah karakter dan angka maka muncul *notice* :



D.4. *Input* Spesial Karakter



Jika *input* adalah karakter maka muncul *notice* :

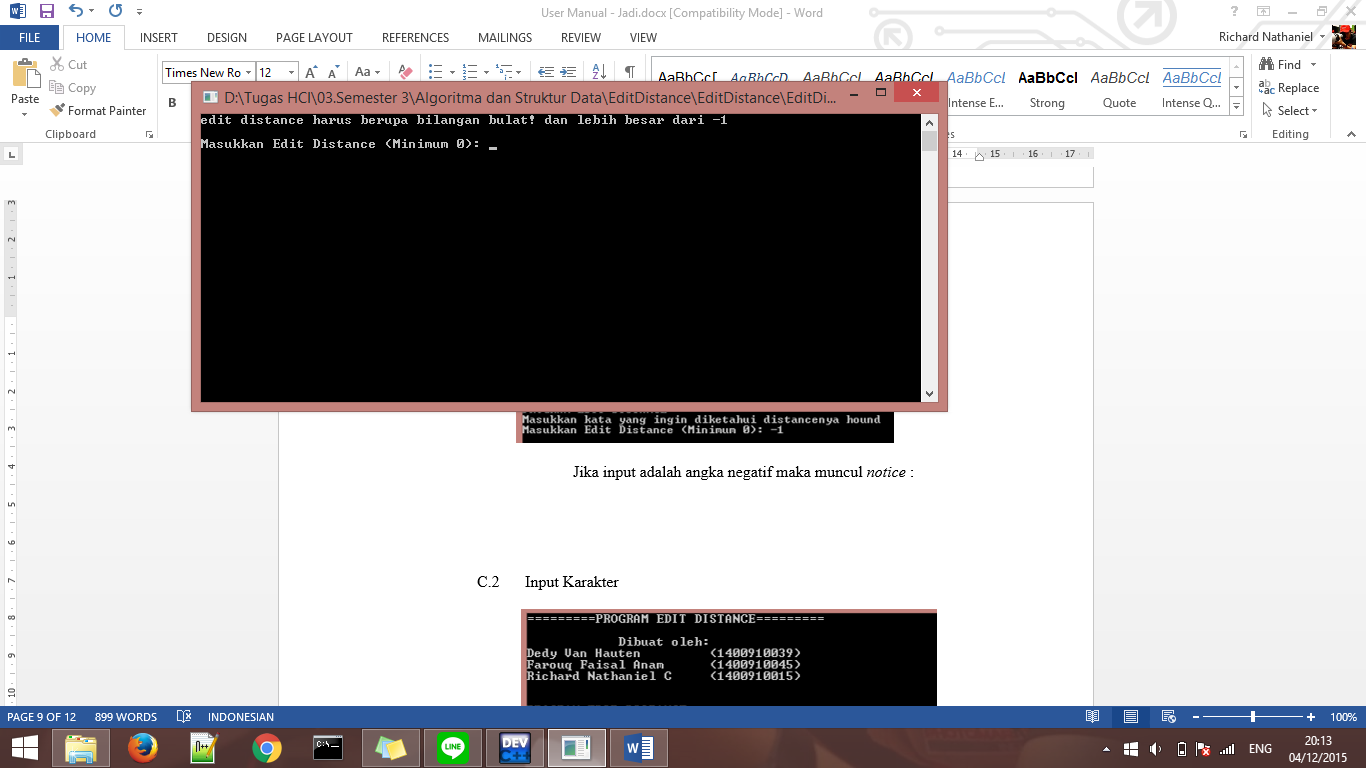


1. *Request Program*

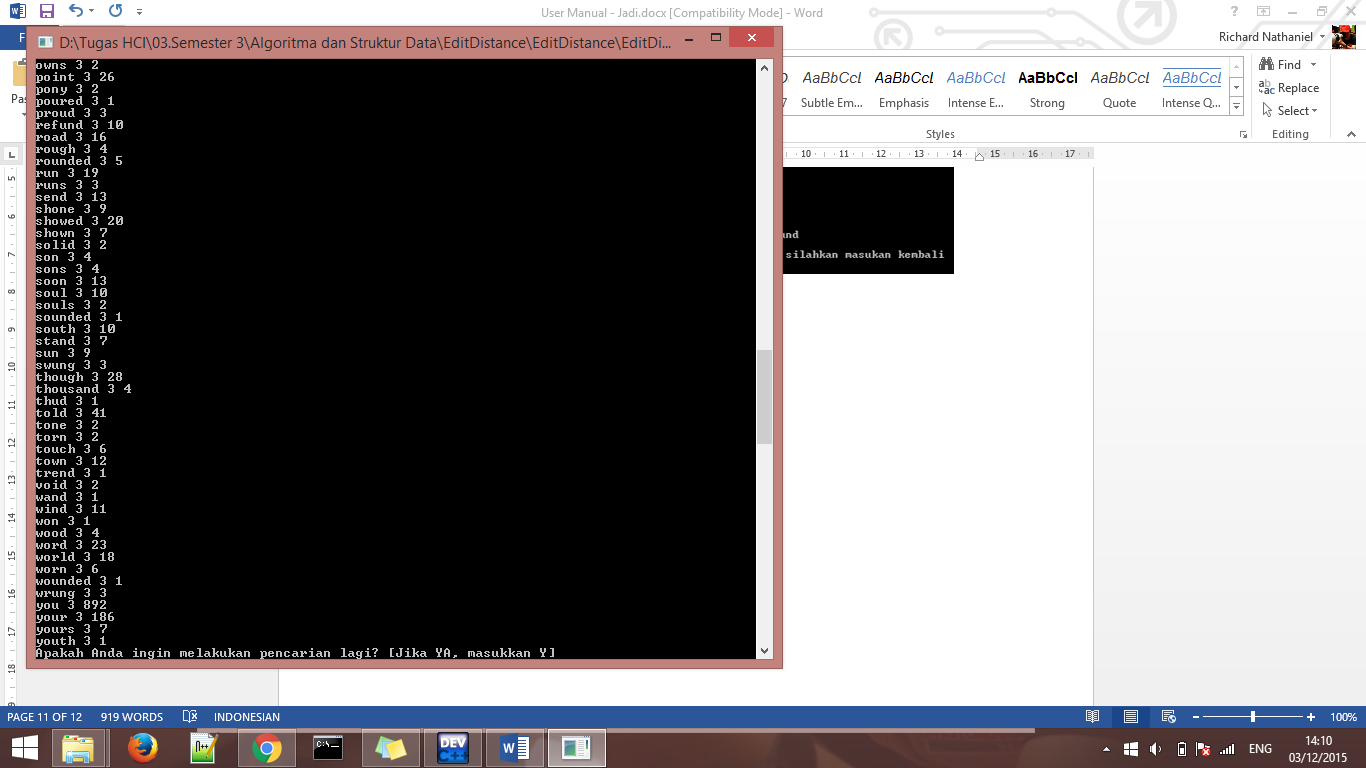
Ketika suatu *input* kata dan *edit distance* dimasukkan, program akan berjalan. Jika *input* *edit distance* salah, program secara otomatis akan meminta user untuk memasukkan *input* *edit distance* yang benar. Setelah *edit distance* dimasukkan dengan benar, program akan berjalan dan di akhir program akan diberikan pilihan apakah user ingin melakukan pencarian lagi atau tidak. Jika iya, maka user akan langsung diarahkan ke menu utama. Jika tidak, program akan berakhir.

D.1 Jika *Input* Program Salah

Jika *input* *edit distance* selain angka positif maka akan muncul *notice* :



D.2 Jika *Input* Program Benar



**DAFTAR PUSTAKA**

CPP Reference. “std::remove, std::remove\_if”. From: http://en.cppreference.com/w/cpp/algorithm/remove accessed 19 November 2015 at 23:45 WIB.

Mulyanto, Arip. 2010. “Analisis *Edit distance* Menggunakan Algoritma *Dynamic Programming*”*.* Universitas Negeri Gorontalo. Volume 5, No. 7.

Sanfoundry. “C++ Program to Implement Levenshtein Distance Computing Algorithm”. From: http://www.sanfoundry.com/cpp-program-implement-levenshtein-distance-computing-algorithm/ accessed 11 November 2015 at 21:16 WIB.

Stack Exchange. “Quick Sort – Implementation”. From: http://codereview.stackexchange.com/questions/77782/quick-sort-implementation accessed 11 November 2015 at 21:16 WIB.

Stack Overflow. “Tolower function for C++ strings“. From: http://stackoverflow.com/questions/3403844/tolower-function-for-c-strings accessed 13 November 2015 at 20:12 WIB.