**LAPORAN JOBSHEET 8**

**String**



Disusun Oleh :

DWI RINAYATUS SA’ADAH

3.34.18.1.10

IK- 1B

**PROGRAM STUDI TEKNIK INFORMATIKA**

**JURUSAN TEKNIK ELEKTRO**

**POLITEKNIK NEGERI SEMARANG**

**TAHUN 2018/2019**

**DAFTAR ISI**

Daftar isi……………………………………………………………………..2

1. Tujuan Instruksional…………………………………………………………2
2. Pendahuluan…………………………………………………………………3
3. Konstanta String……………………………………………………………..4
4. Variabel String……………………………………………………………....4
5. Inisialisasi String…………………………………………………………….4
6. Input Output Data String…………………………………………………….4
7. Memasukkan Data String……………………………………………..5
8. Menampilkan Data String…………………………………………….5
9. Mengakses Elemen String…………………………………………………...8
10. *Built-In Fungtion* untuk manipulasi String………………………………….9
11. Fungsi *strcpy()*………………………………………………………10
12. Fungsi *strlen()*……………………………………………………….11
13. Fungsi *strcmp()*……………………………………………………...12
14. Fungsi *strcmpi()*……………………………………………………..13
15. Fungsi yang terdapat dalam string.h………………………………...14
16. Fungsi yang belum dibahas………………………………………….15
17. Latihan……………………………………………………………………...15
18. Kesimpulan…………………………………………………………………32
19. Penutup……………………………………………………………………..32

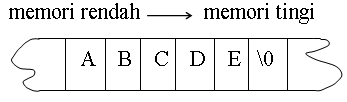
**STRING**

1. **Tujuan Instruksional**
2. Mengetahui dan memahami tentang perintah untuk membuat struktur dalam bahasa C.
3. Menjelaskan tentang bentuk umum atau struktur dari perintah-perintah yang terkait dengan struktur.
4. Membuat program yang melibatkan permasalahan-permasalahan yang membutuhkan struktur.
5. **Pendahuluan**
6. String merupakan bentuk data yang biasa dipakai dalam bahasa pemrograman untuk keperluan menampung dan memanipulasi data teks, misalnya untuk menampung (menyimpan) suatu kalimat.
7. Pada bahasa C, string bukanlah merupakan tipe data tersendiri, melainkan hanyalah kumpulan dari nilai-nilai karakter yang berurutan dalam bentuk ***array*** berdimensi satu 🡪 *array of char*.
8. **Konstanta String**

* Suatu konstanta string ditulis dengan diawali dan diakhiri tanda petik ganda, misalnya:

“ABCDE”

* Nilai string ini disimpan dalam memori secara berurutan dengan komposisi sebagai berikut:



* Setiap karakter akan menempati memori sebesar 1 byte.
* Byte terakhir otomatis akan berisi karakter NULL (\0), dengan demikian maka akhir dari nilai suatu string akan dapat dideteksi.
* Sebagai sebuah *array of char*, karakter pertama dari nilai string mempunyai indeks ke-0, karakter kedua mempunyai indeks ke-1, dan seterusnya.

1. **Variabel String**

* Variabel string adalah variabel yang dipakai utuk menyimpan nilai string. Misalnya :

char name[15];

merupakan instruksi untuk mendeklarasikan variabel string dengan panjang maksimal 15 karakter (termasuk karakter NULL).

* Deklarasi tersebut sebenarnya tidak lain merupakan deklarasi array bertipe *char*.

1. **Inisialisasi String**

* Suatu variabel string dapat diinisialisasi seperti halnya array yang lain (dalam kurung kurawal dipisahkan koma). Namun tentu saja elemen terakhirnya haruslah berupa karakter NULL. Sebagai contoh :

char name[] = {'R','I','N', 'I',’\0’};

yang menyatakan bahwa **name** adalah variabel string dengan nilai awal berupa string : “RINI” .

* Bentuk inisialisasi yang lebih singkat :

char name[] = “RINI”;

pada bentuk ini, karakter NULL tidak perlu ditulis. Secara IMPLISIT akan disisipkan oleh kompiler.

1. **Input Output Data String**

* Untuk memasukkan atau menampilkan data String digunakan bisa beberapa fungsi standar yang ada di stdio.h.
* Untuk operasi input :
  + scanf()
  + gets()
  + fgets()
* Untuk operasi output :
  + puts()
  + printf()

1. **Memasukkan Data String**

* Pemasukan data string ke dalam suatu variabel biasa dilakukan dengan fungsi gets() atau scanf().
* Bentuk umum pemakaiannya adalah sebagai berikut :

#include <stdio.h>

gets(nama\_array);

atau

#include <stdio.h>

scanf(“%s”, nama\_array);

1. **Menampilkan Data String**

Perhatikan :

* **nama\_array** adalah variabel bertipe *array of char* yang akan digunakan untuk menyimpan string masukan.
* Di depan **nama\_array** tidak perlu ada operator ***&*** (operator alamat), karena **nama\_array** tanpa kurung siku sudah menyatakan alamat yang ditempati oleh elemen pertama dari *array* tsb.
* Kalau memakai scanf(), data string masukan tidak boleh mengandung spasi

Contoh Program

main(){

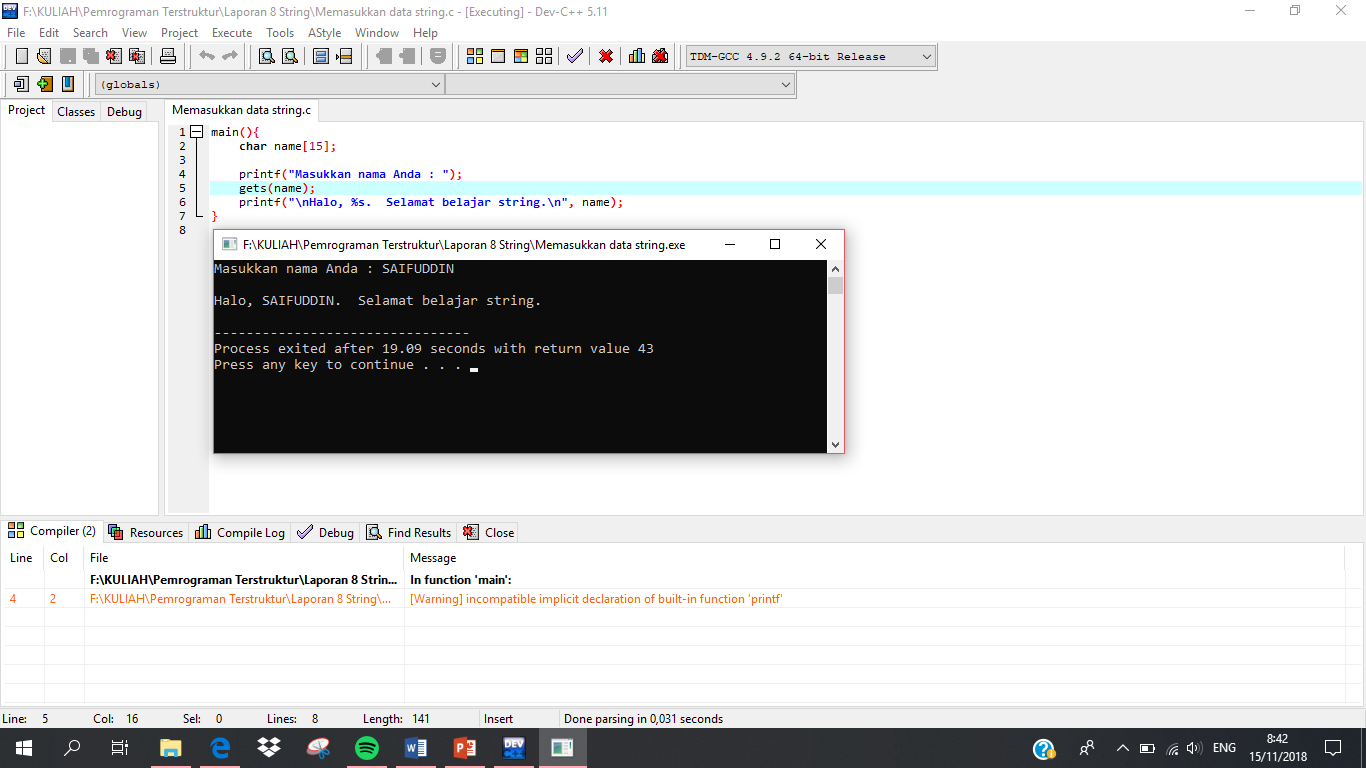
char name[15];

printf("Masukkan nama Anda : ");

gets(name);

printf("\nHalo, %s. Selamat belajar string.\n", name);

}



* Perlu diketahui, fungsi gets() akan membaca seluruh karakter yang diketik melalui keyboard sampai tombol ENTER ditekan dengan tanpa mengecek batasan panjang array yang merupakan argumennya.
* Jika string yang dimasukkan melebihi ukuran array, maka sisa string (panjang string masukan dikurangi ukuran array plus karakter NULL) akan ditempatkan di lokasi sesudah bagian akhir dari array tersebut. Tentu saja kejadian seperti ini bisa menimbulkan hal yang tidak diinginkan, misalnya berubahnya isi variabel yang dideklarasikan sesudah array tersebut karena tertumpuki oleh string yang dimasukkan *(overwrite)*, atau perilaku program yang sama sekali berbeda dengan kemauan user yang dalam hal ini pelacakan kesalahannya *(debugging)* sangat sulit dilakukan, atau bahkan terjadi penghentian program secara tidak normal
* Untuk mengatasi hal itu, disarankan untuk menggunakan fungsi fgets() untuk menggantikan fungsi gets() dalam memasukkan data string.
* Bentuk umum pemakaian fgets() adalah :

#include <stdio.h>

fgets(nama\_array, sizeof nama\_array, stdin);

**Uji Coba gets()**

#include <stdio.h>

#define MAKS 5

main(){

char kar = 'A';

char nama[MAKS];

printf("Karakternya = %c\n", kar);

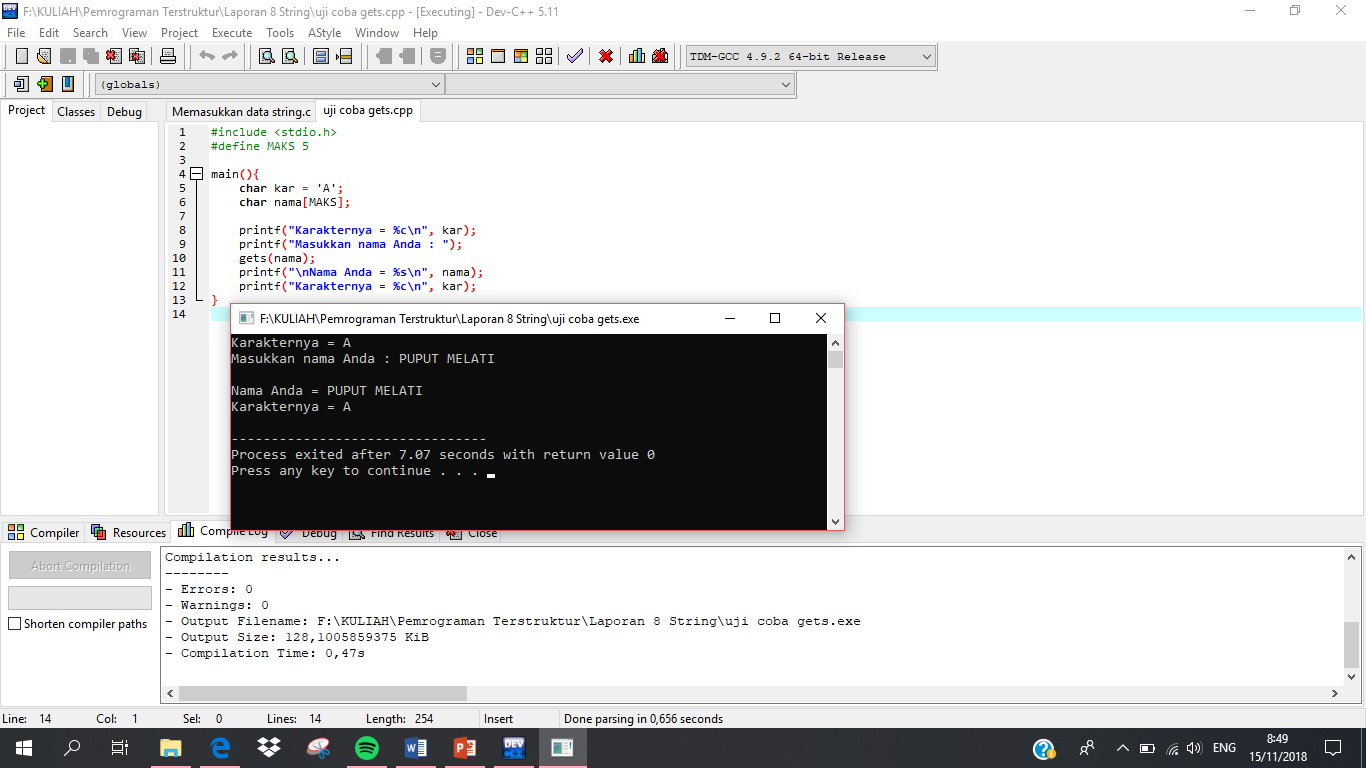
printf("Masukkan nama Anda : ");

gets(nama);

printf("\nNama Anda = %s\n", nama);

printf("Karakternya = %c\n", kar);

}



* Untuk menampilkan isi variabel string, fungsi yang digunakan adalah puts() atau printf().
* Bentuk umum pemakaiannya adalah sebagai berikut :

#include <stdio.h>

puts(var\_string);

atau

printf("%s",var\_string);

Dalam hal ini var\_string adalah sebuah variabel yang berupa sebuah *array of char*.

* Fungsi puts() akan menampilkan isi dari var\_string dan secara otomatis menambahkan karakter '\n' di akhir string.
* Sedangkan fungsi printf() akan menampilkan isi variabel string tanpa memberikan tambahan '\n'. Sehingga, agar kedua pernyataan di atas memberikan keluaran yang sama, maka pada pernyataan printf() dirubah menjadi :

printf("%s\n", var\_string);

1. **Mengakses Elemen String**

* Variabel string merupakan bentuk khusus dari array bertipe *char*. Oleh karena itu, elemen dari variabel string dapat diakses seperti halnya pengaksesan elemen pada array.
* Perhitungan jumlah karakter dari string teks dapat dilakukan dengan memeriksa elemen dari string dimulai dari posisi yang pertama (indeks ke-0) sampai ditemukannya karakter NULL.
* Program berikut menunjukkan cara mengakses elemen array untuk menghitung total karakter dari string yang dimasukkan melalui keyboard

Contoh Program

#include <stdio.h>

#define MAKS 256

main() {

int i, jumkar = 0;

char teks[MAKS];

puts("Masukkan suatu kalimat (maks 255 karakter).");

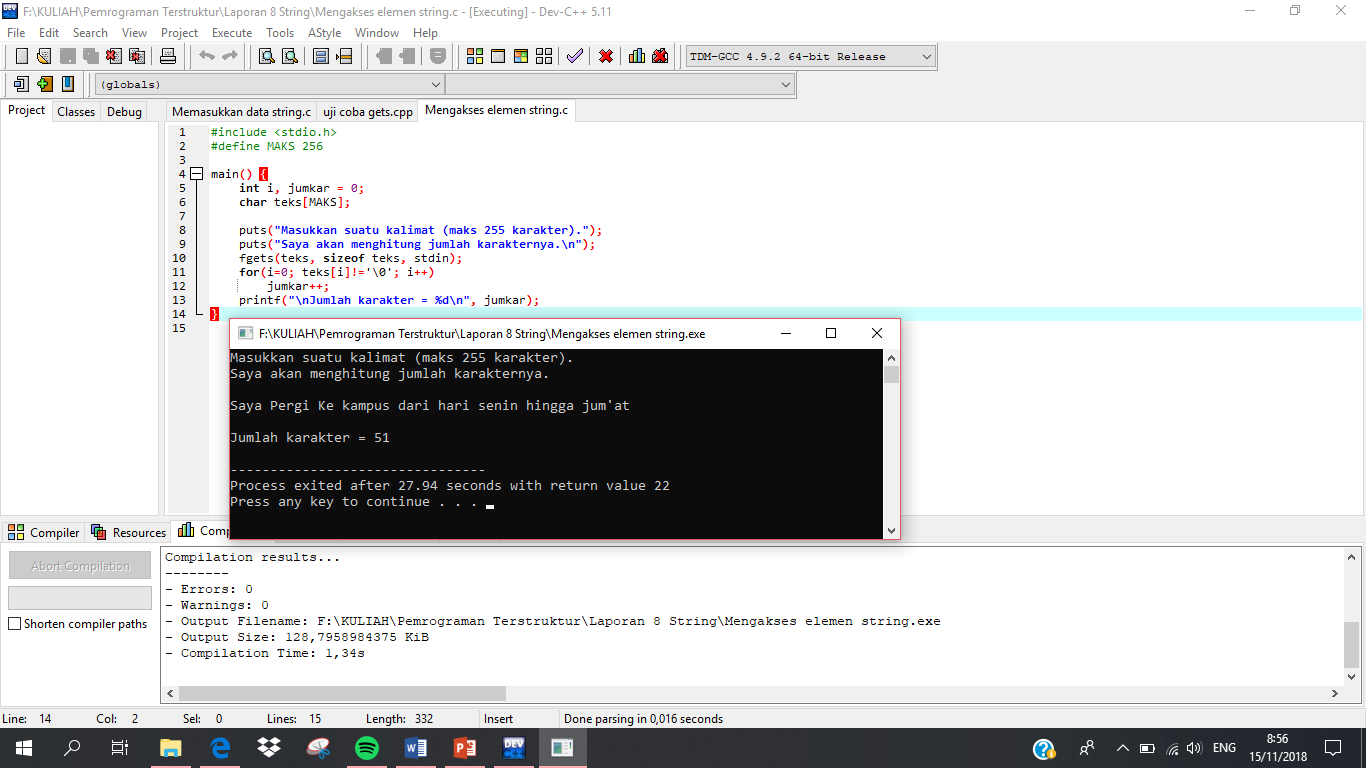
puts("Saya akan menghitung jumlah karakternya.\n");

fgets(teks, sizeof teks, stdin);

for(i=0; teks[i]!='\0'; i++)

jumkar++;

printf("\nJumlah karakter = %d\n", jumkar);

}

1. ***Built-In Fungtion* untuk manipulasi String**

* Untuk manipulasi string, C telah menyediakan beberapa fungsi standar yang ada pada string.h
* Beberapa yang akan dibahas kali ini adalah
  1. Fungsi strcpy()
  2. Fungsi strlen()
  3. Fungsi strrev()
  4. Fungsi strcmp()
  5. Fungsi strcmpi()

1. **Fungsi *strcpy()***

* Bentuk pemakaian :

#include <string.h>

strcpy(tujuan, asal);

* Fungsi ini dipakai untuk mengcopy string asal ke variabel string tujuantermasuk karakter '\0'.
* Dalam hal ini, variabel tujuan haruslah mempunyai ukuran yang dapat digunakan untuk menampung seluruh karakter dari string asal

Contoh Program

#include <stdio.h>

#include <string.h>

#define MAKS 80

main()

{

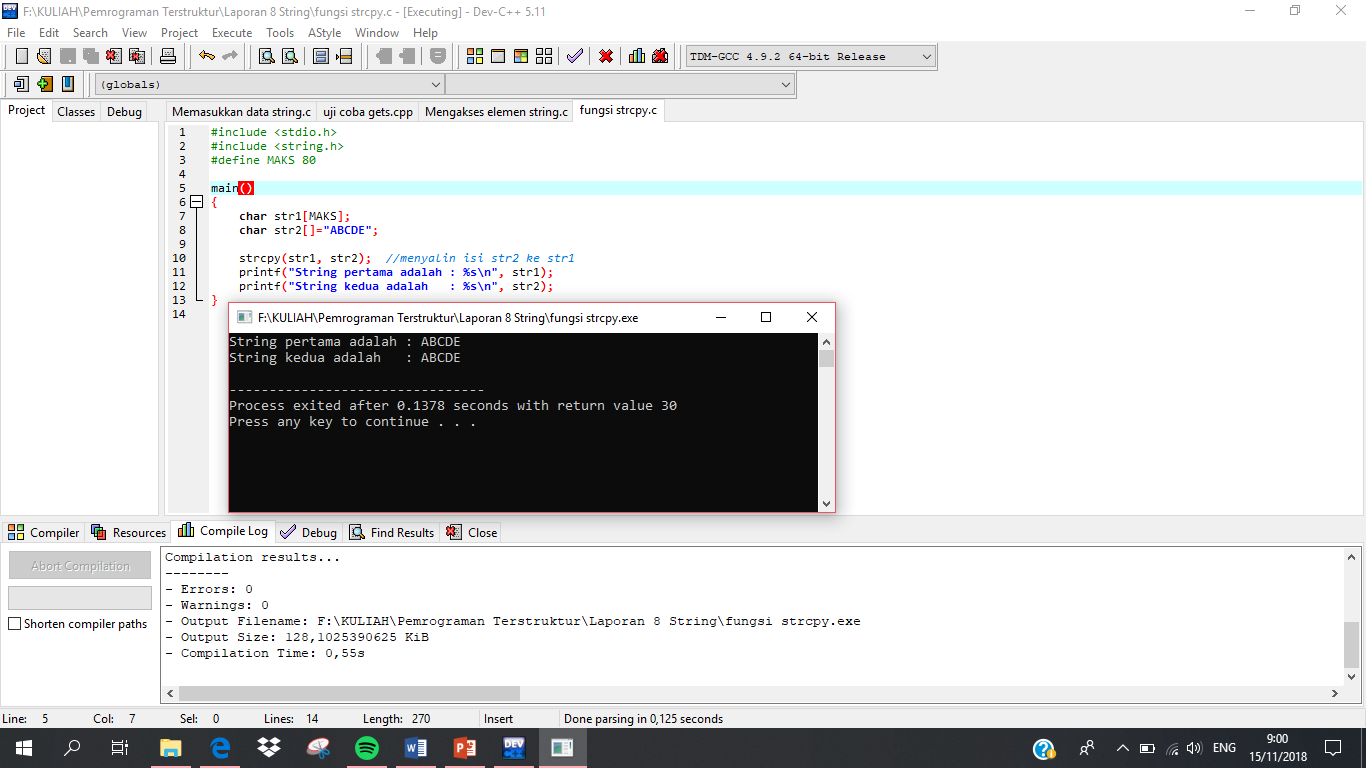
char str1[MAKS];

char str2[]="ABCDE";

strcpy(str1, str2); //menyalin isi str2 ke str1

printf("String pertama adalah : %s\n", str1);

printf("String kedua adalah : %s\n", str2);

}

1. **Fungsi *strlen()***

* Bentuk pemakaian :

#include <string.h>

strlen(var\_string);

* Fungsi ini digunakan untuk memperoleh banyaknya karakter di dalam string yang menjadi argumennya (var\_string).
* Keluaran dari fungsi ini adalah panjang dari var\_string(karakter NULL tidak ikut dihitung)

Contoh Program

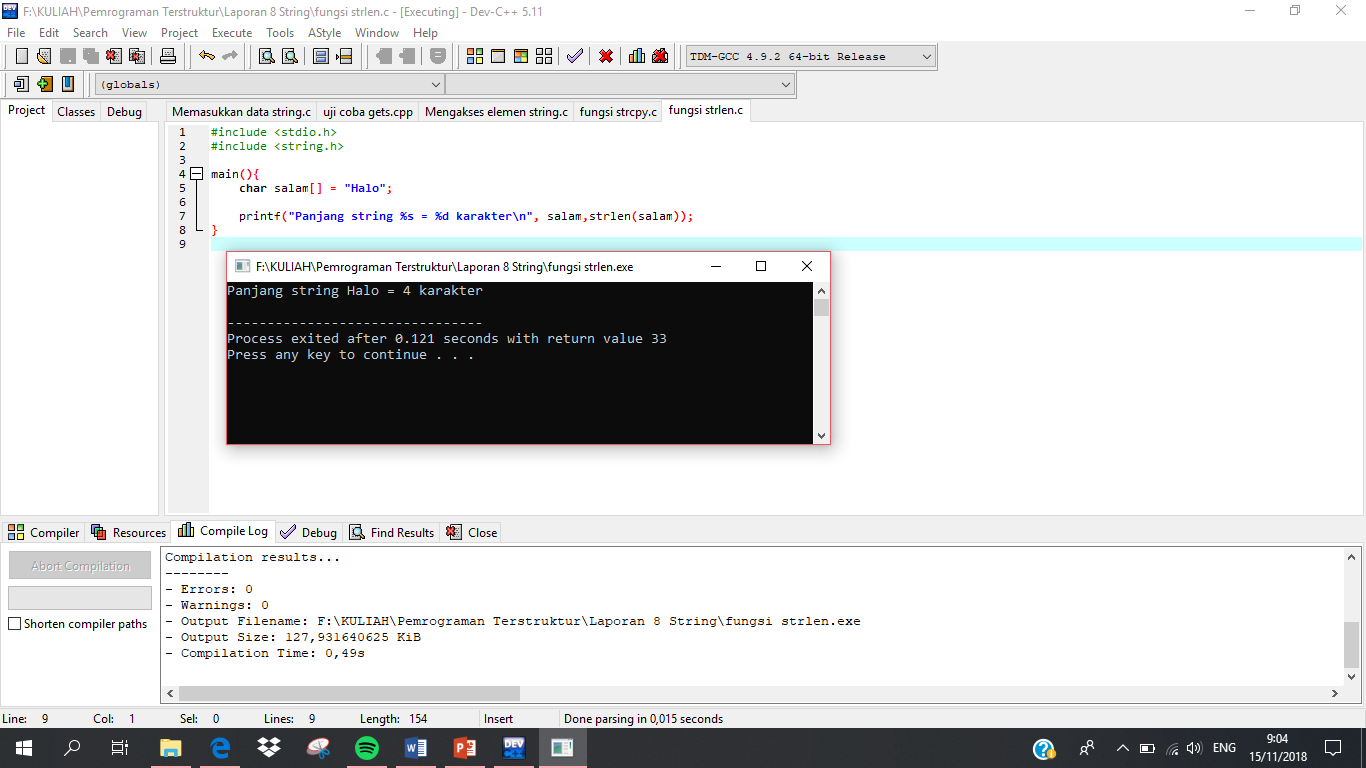
#include <stdio.h>

#include <string.h>

main(){

char salam[] = "Halo";

printf("Panjang string %s = %d karakter\n", salam,strlen(salam));

}

1. **Fungsi *strcmp()***

* Membandingkan dua nilai string juga tidak dapat digunakan dengan operator hubungan, karena operator tersebut tidak untuk operasi string.
* Membandingkan dua buah nilai string secara ***case sensitive*** dapat dilakukan dengan fungsi strcmp()*.*
* Contoh bentuk pemakaian fungsi :

#include <string.h>

strcmp(str1, str2);

* Fungsi ini dipakai untuk membandingkan string str1 dengan string str2. Keluaran dari fungsi ini bertipe *int* yang berupa nilai :
  + - **-1**, jika str1 kurang dari str2
    - **0**, jika str1sama dengan str2
    - **1**, jika str1 lebih dari str2
* Pembandingan dilakukan untuk karakter pada posisi yang sama dari str1 dan str2, dimulai dari karakter terkiri yang didasarkan oleh nilai ASCII-nya. Misal, karakter ‘A’ lebih kecil daripada ‘B’ dan karakter ‘B lebih kecil daripada ‘C’.

Contoh Program

#include <stdio.h>

#include <string.h>

main(){

char str1[]="HALO";

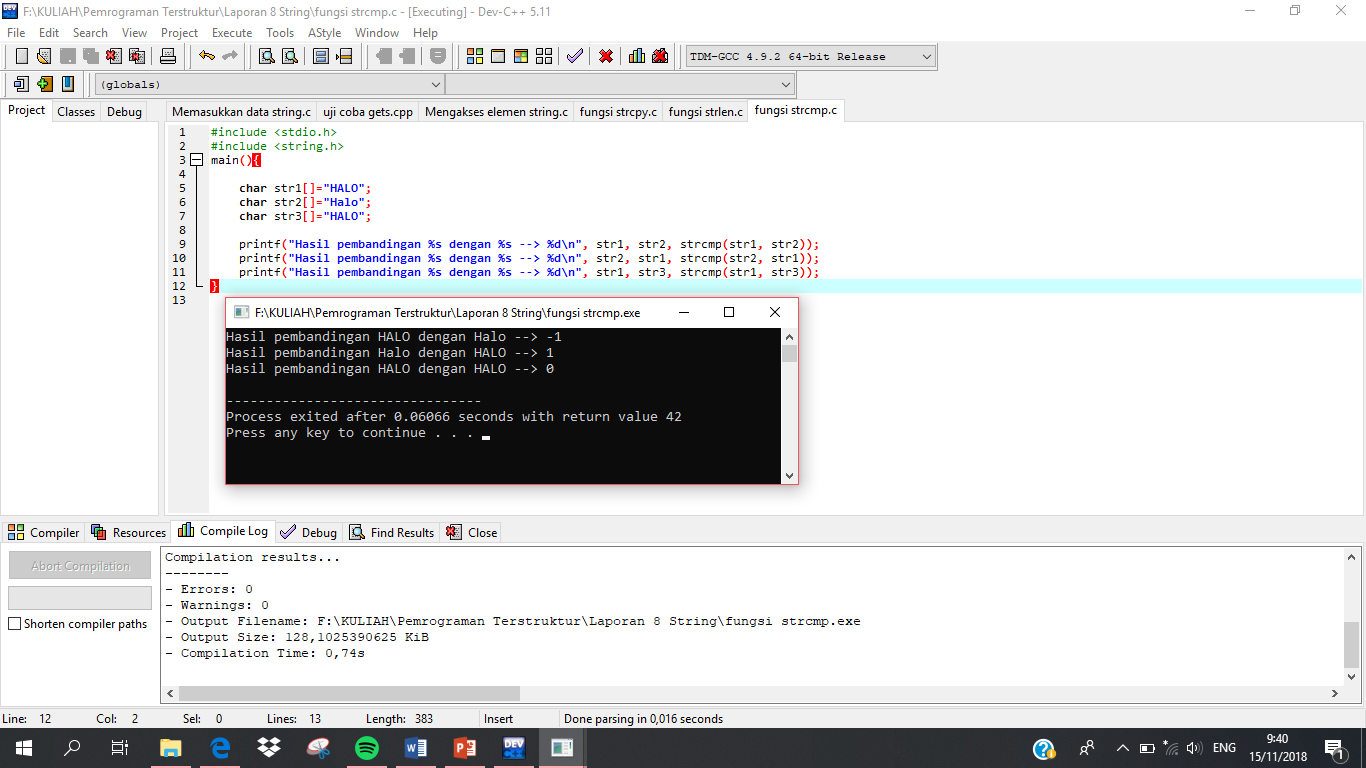
char str2[]="Halo";

char str3[]="HALO";

printf("Hasil pembandingan %s dengan %s --> %d\n", str1, str2, strcmp(str1, str2));

printf("Hasil pembandingan %s dengan %s --> %d\n", str2, str1, strcmp(str2, str1));

printf("Hasil pembandingan %s dengan %s --> %d\n", str1, str3, strcmp(str1, str3));

}

1. **Fungsi *strcmpi()***

* Membandingkan dua buah nilai string secara ***non case sensitive*** dapat dilakukan dengan fungsi strcmpi()*.*
* Contoh bentuk pemakaian fungsi :

#include <string.h>

strcmpi(str1, str2);

* Fungsi ini dipakai untuk membandingkan string str1 dengan string str2. Keluaran dari fungsi ini bertipe *int* yang berupa nilai :
  + - **-1**, jika str1 kurang dari str2
    - **0**, jika str1sama dengan str2
    - **1**, jika str1 lebih dari str2

Contoh Program

#include <stdio.h>

#include <string.h>

main(){

char str1[]="HALO";

char str2[]="harimau";

char str3[]="halo";

printf("Hasil pembandingan %s dengan %s --> %d\n",

str1, str2, strcmpi(str1, str2));

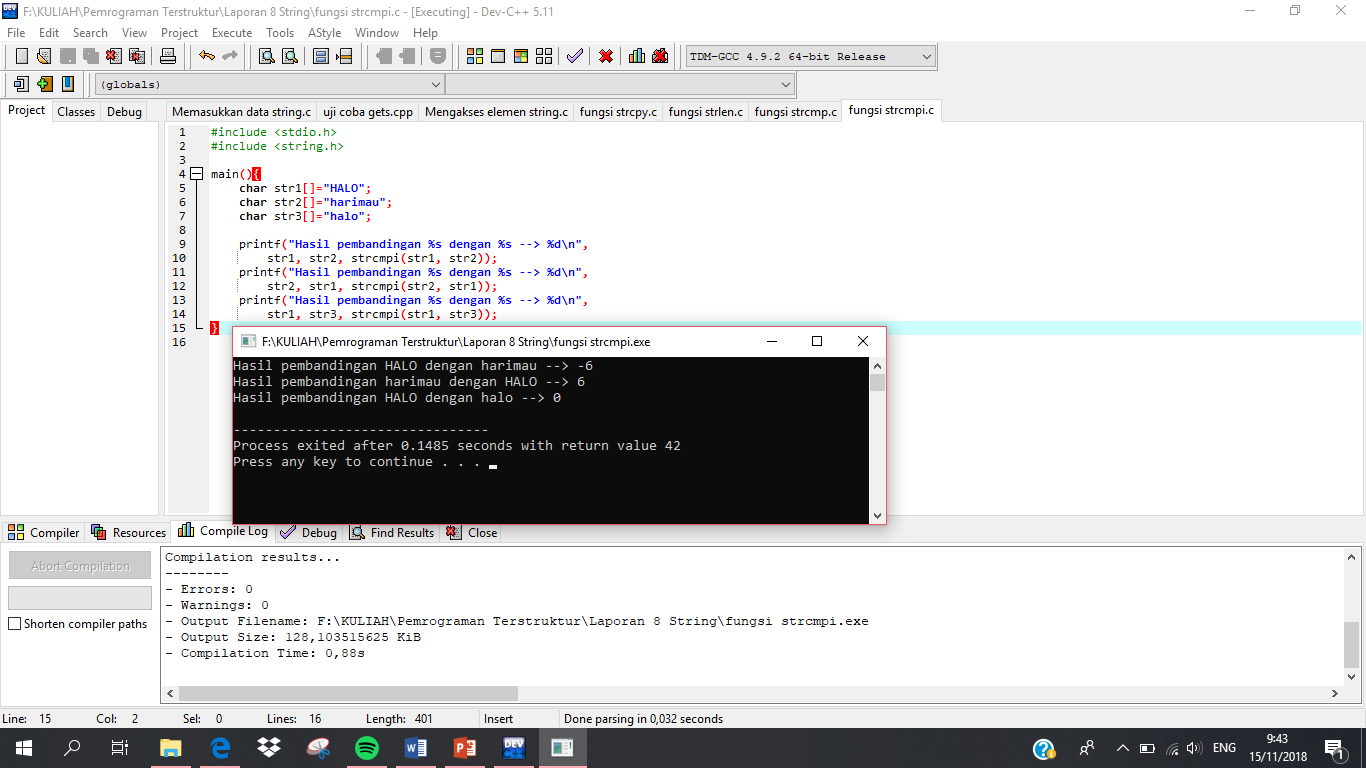
printf("Hasil pembandingan %s dengan %s --> %d\n",

str2, str1, strcmpi(str2, str1));

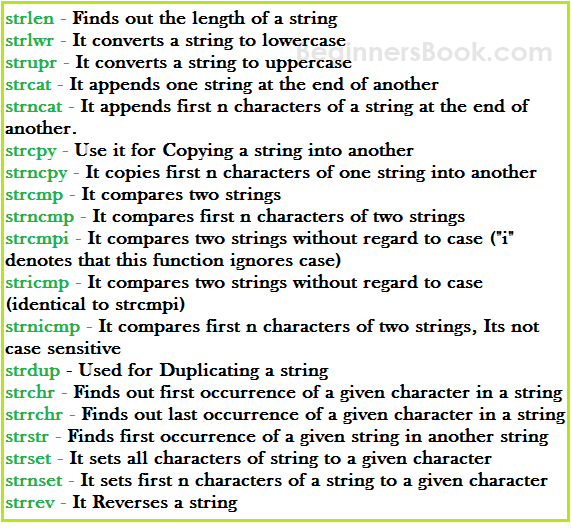
printf("Hasil pembandingan %s dengan %s --> %d\n",

str1, str3, strcmpi(str1, str3));

}



1. **Fungsi yang terdapat dalam string.h**

****

1. **Fungsi yang belum dibahas**

* Strlwr
* Strnset
* strrev
* Strupr
* Strcat
* Strncat
* Strncpy
* Strncmp
* Stricmp
* Strnicmp
* Strdup
* Strchr
* Strrchr
* Strstr

1. **Latihan**

* Ketikkan semua contoh program yang ada pada modul teori (10.String.ppt)
* Running setiap program dan amatilah outputnya
* Berikan analisis dan kesimpulan pada setiap contoh program tsb
* Program menampilkan data string

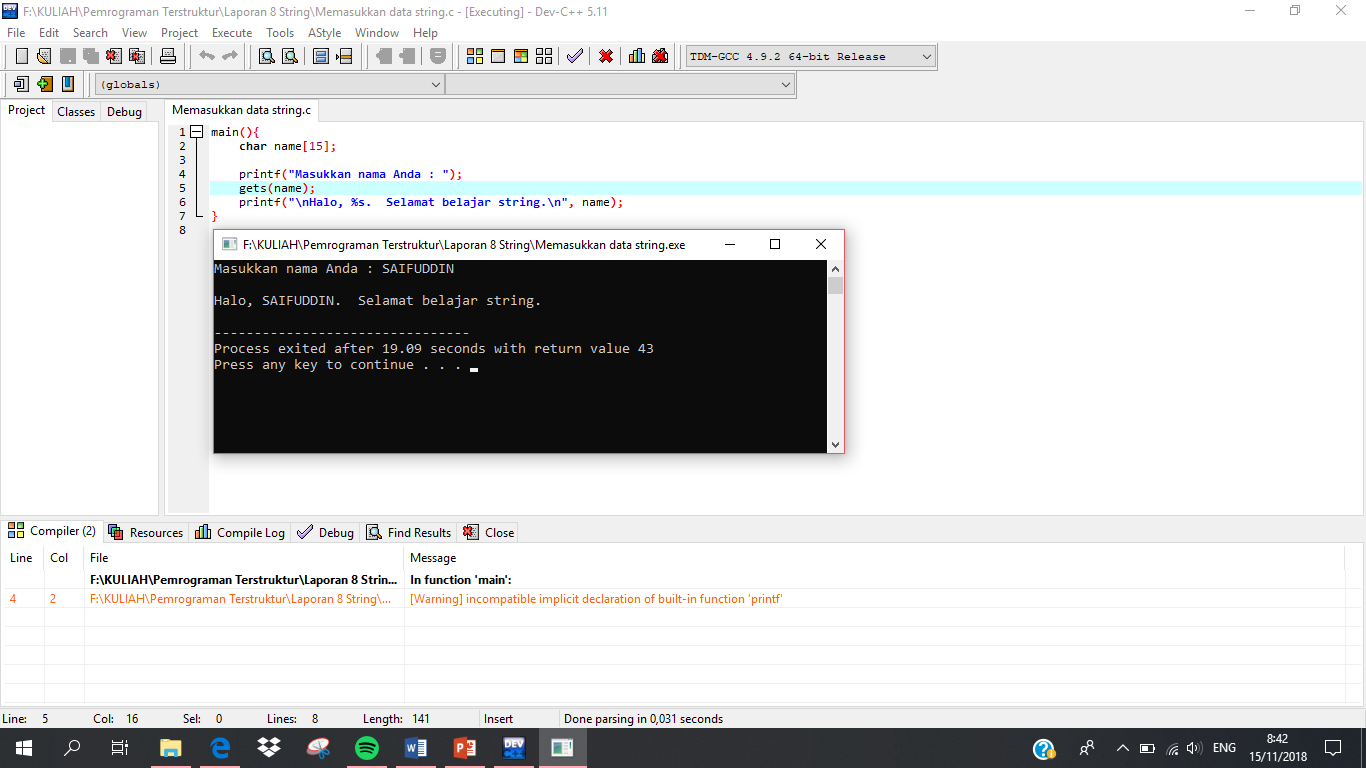
main(){

char name[15];

printf("Masukkan nama Anda : ");

gets(name);

printf("\nHalo, %s. Selamat belajar string.\n", name);

}

Analisis dan Kesimpulan

Ketika menggunakan string maka akan dideklarasikan berapa jumlah kotak yang dapat diisi nama termasuk \0 di akhir. Apabila di pendeklarasian 15 maka yang dapat digunakan hanya 14 kotak pendeklarasian.

* Uji Coba gets

#include <stdio.h>

#define MAKS 5

main(){

char kar = 'A';

char nama[MAKS];

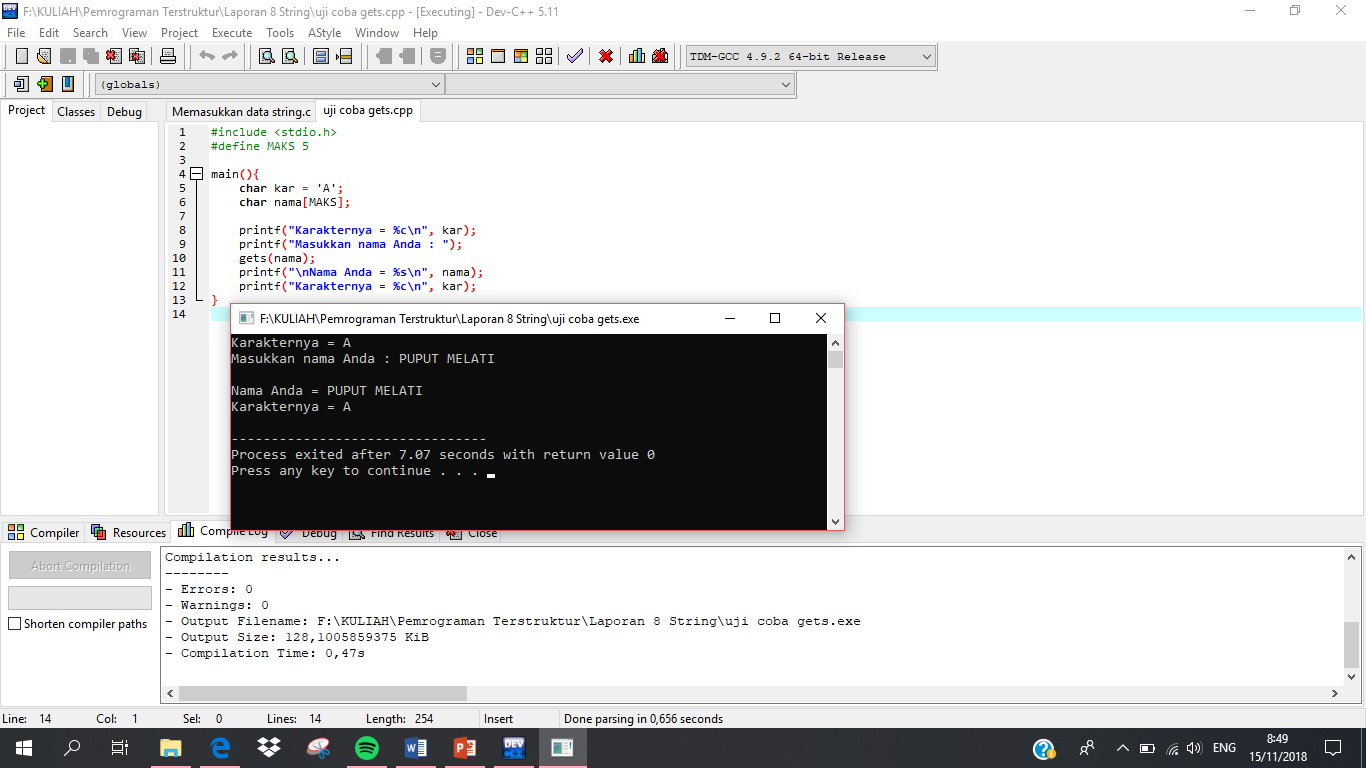
printf("Karakternya = %c\n", kar);

printf("Masukkan nama Anda : ");

gets(nama);

printf("\nNama Anda = %s\n", nama);

printf("Karakternya = %c\n", kar);

}

Analisis dan Kesimpulan

Pada program uji coba gets kita dapat menggunakan gets sebagai pengganti scanf karena pada dasarnya fungsinya sama.

* Mengakses elemen string

#include <stdio.h>

#define MAKS 256

main() {

int i, jumkar = 0;

char teks[MAKS];

puts("Masukkan suatu kalimat (maks 255 karakter).");

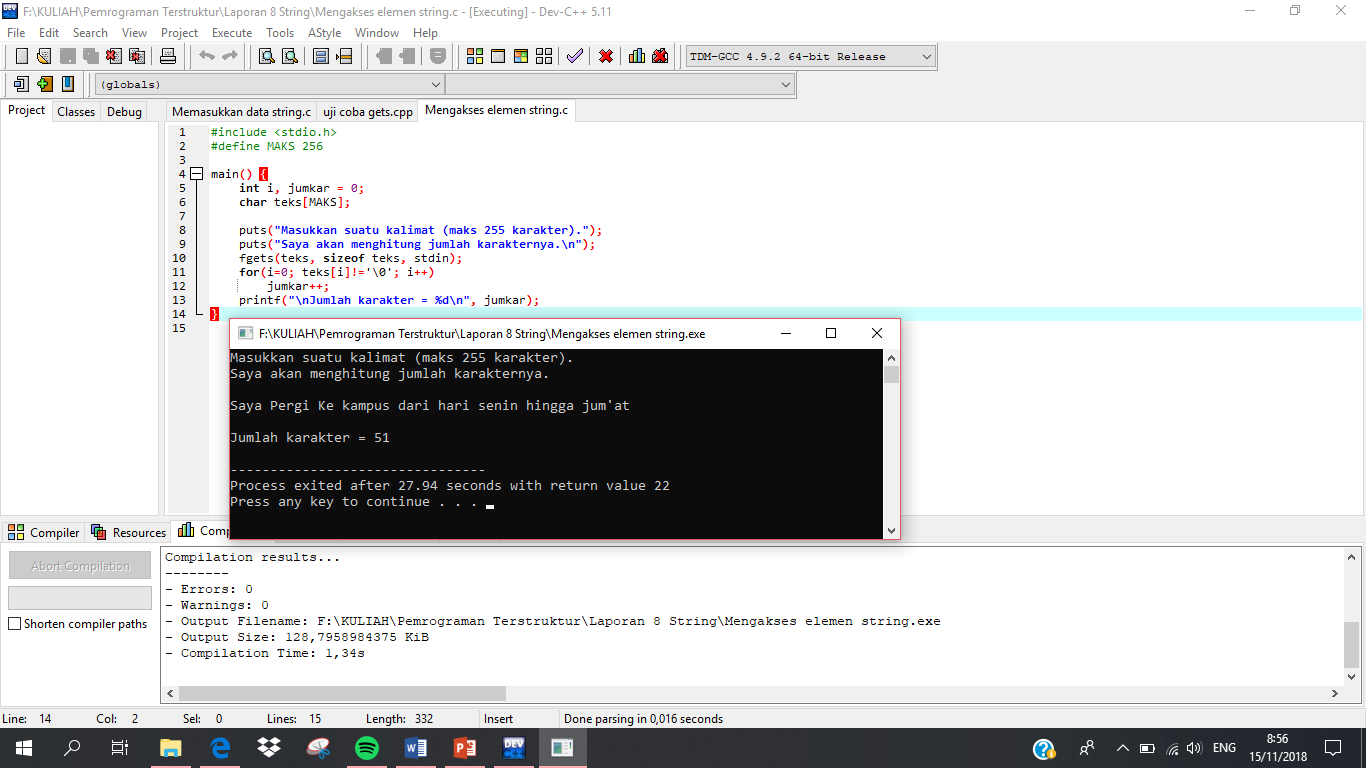
puts("Saya akan menghitung jumlah karakternya.\n");

fgets(teks, sizeof teks, stdin);

for(i=0; teks[i]!='\0'; i++)

jumkar++;

printf("\nJumlah karakter = %d\n", jumkar);

}

Analisis dan Kesimpulan

Program tersebut menunjukkan cara mengakses elemen array untuk menghitung total karakter dari string yang dimasukkan melalui keyboard

* Fungsi strcpy()

#include <stdio.h>

#include <string.h>

#define MAKS 80

main()

{

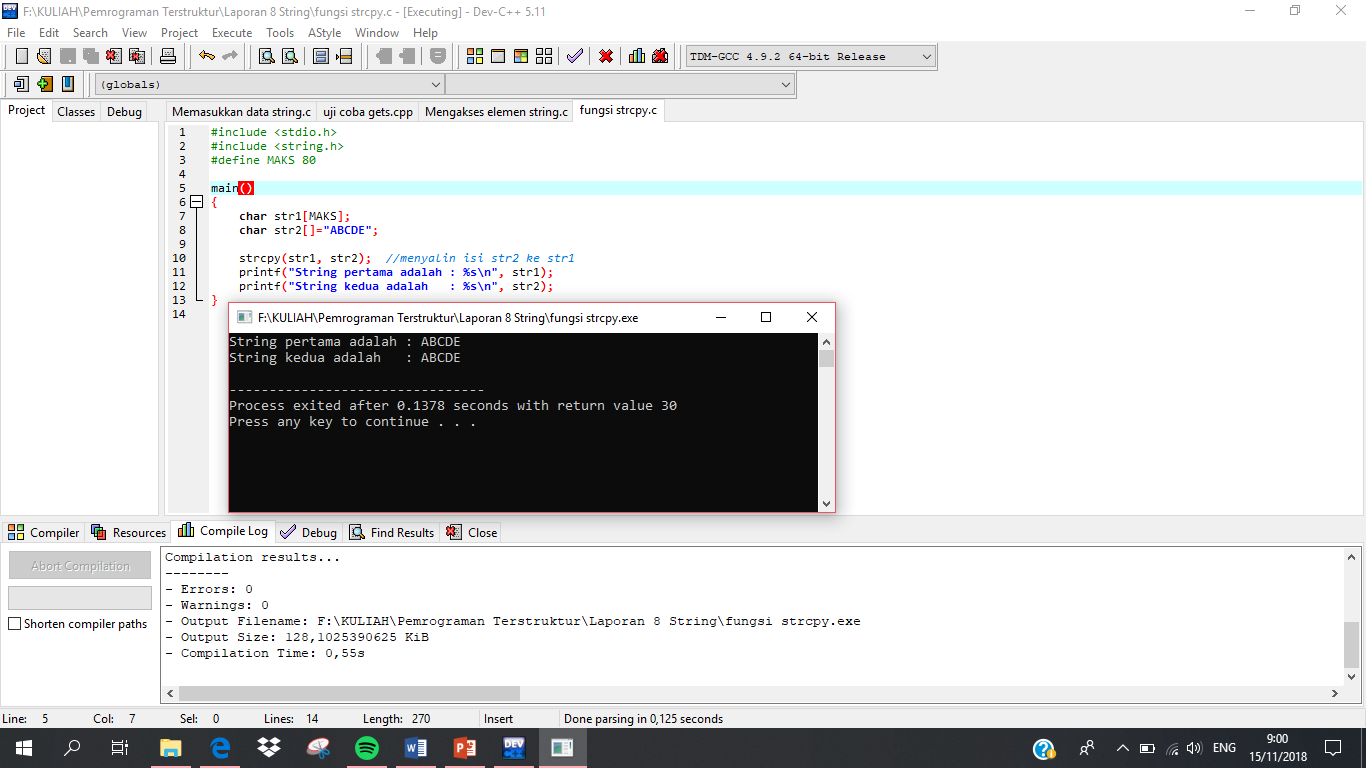
char str1[MAKS];

char str2[]="ABCDE";

strcpy(str1, str2); //menyalin isi str2 ke str1

printf("String pertama adalah : %s\n", str1);

printf("String kedua adalah : %s\n", str2);

}

Analisis dan Kesimpulan

Program tersebut menunjukkan penggunaan fungsi strcpy(). Dengan menggunakan fungsi strcpy maka semua yang berada di string asal akan dicopy ke variabel string tujuantermasuk karakter '\0'.

* Fungsi strlen()

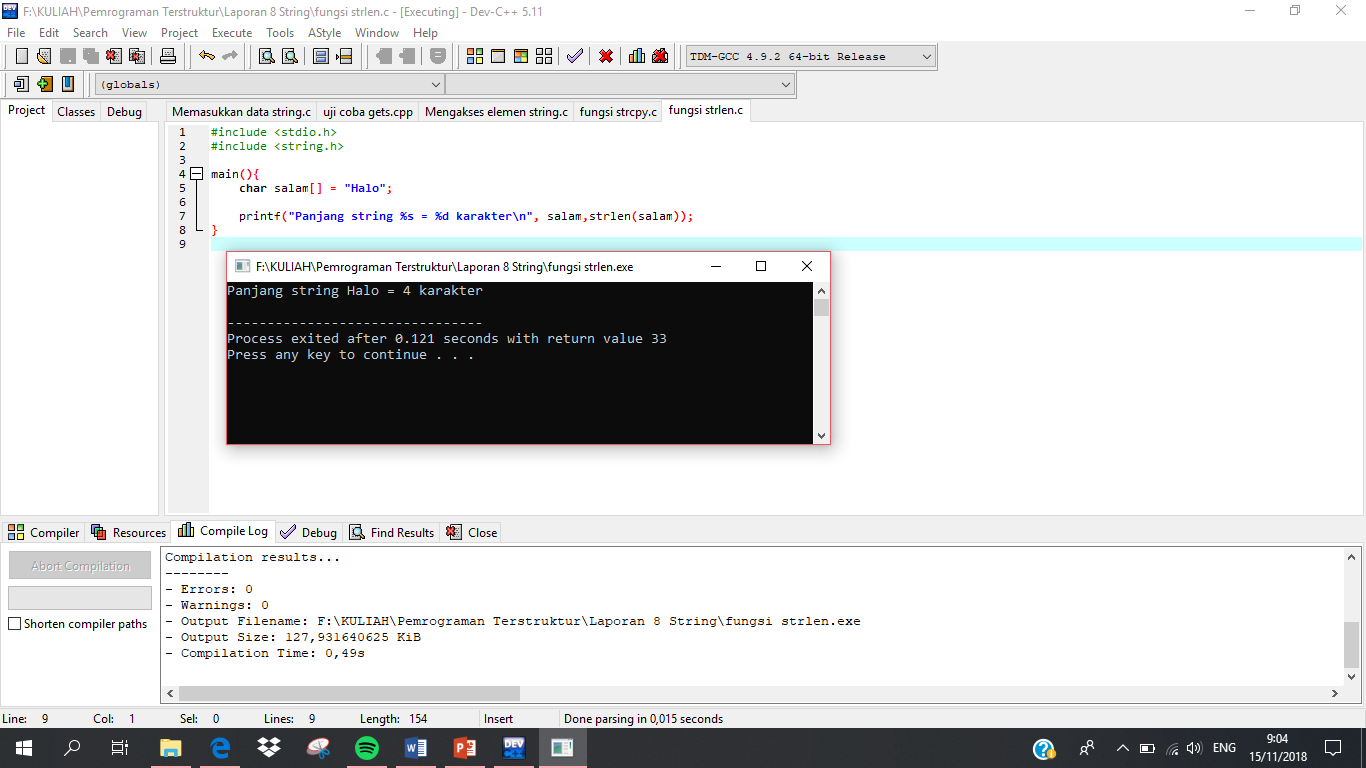
#include <stdio.h>

#include <string.h>

main(){

char salam[] = "Halo";

printf("Panjang string %s = %d karakter\n", salam,strlen(salam));

}

Analisis dan Kesimpulan

Program tersebut menunjukkan penggunaan fungsi strlen(). Dengan menggunakan fungsi strlen() berguna untuk memperoleh banyaknya karakter di dalam string yang menjadi argumennya (var\_string).

* Fungsi strcmp()

#include <stdio.h>

#include <string.h>

main(){

char str1[]="HALO";

char str2[]="Halo";

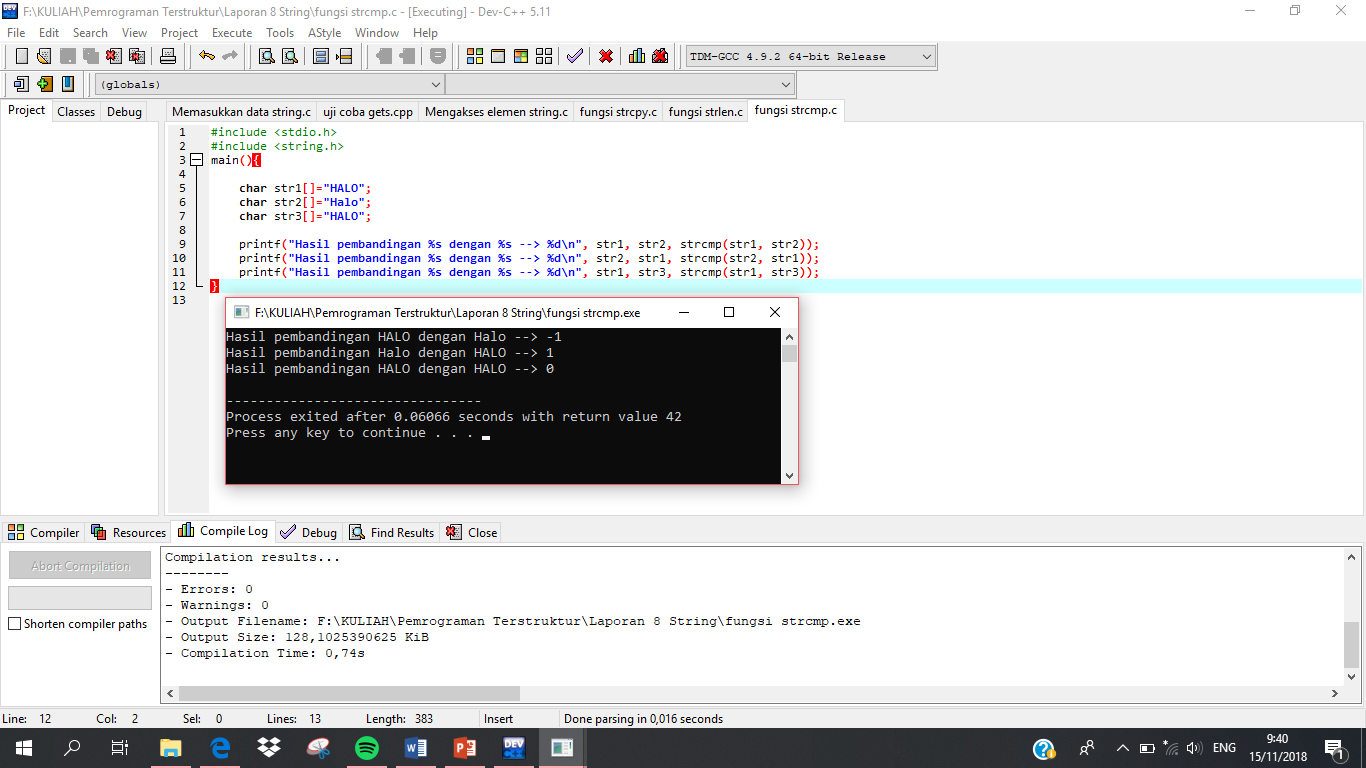
char str3[]="HALO";

printf("Hasil pembandingan %s dengan %s --> %d\n", str1, str2, strcmp(str1, str2));

printf("Hasil pembandingan %s dengan %s --> %d\n", str2, str1, strcmp(str2, str1));

printf("Hasil pembandingan %s dengan %s --> %d\n", str1, str3, strcmp(str1, str3));

}

Analisis dan Kesimpulan

Program tersebut menunjukkan penggunaan fungsi strcmp(). Dengan menggunakan fungsi strcmp() kita dapat membandingkan str 1 dengan str 2 secara **case sensitive** yaitu membandingkan dengan nilai karaker di ASCII-nya missal A dengan a berbeda nilainya di dalam ASCII.

Untuk hasil apabila -1 berarti str 1 lebih kecil daripada str 2, apabila 1 artinya str 1 lebih besar daripada str 2, apabila hasil 0 berarti str 1 dengan str 2 bernilai sama.

* Fungsi strcmpi()

#include <stdio.h>

#include <string.h>

main(){

char str1[]="HALO";

char str2[]="harimau";

char str3[]="halo";

printf("Hasil pembandingan %s dengan %s --> %d\n",

str1, str2, strcmpi(str1, str2));

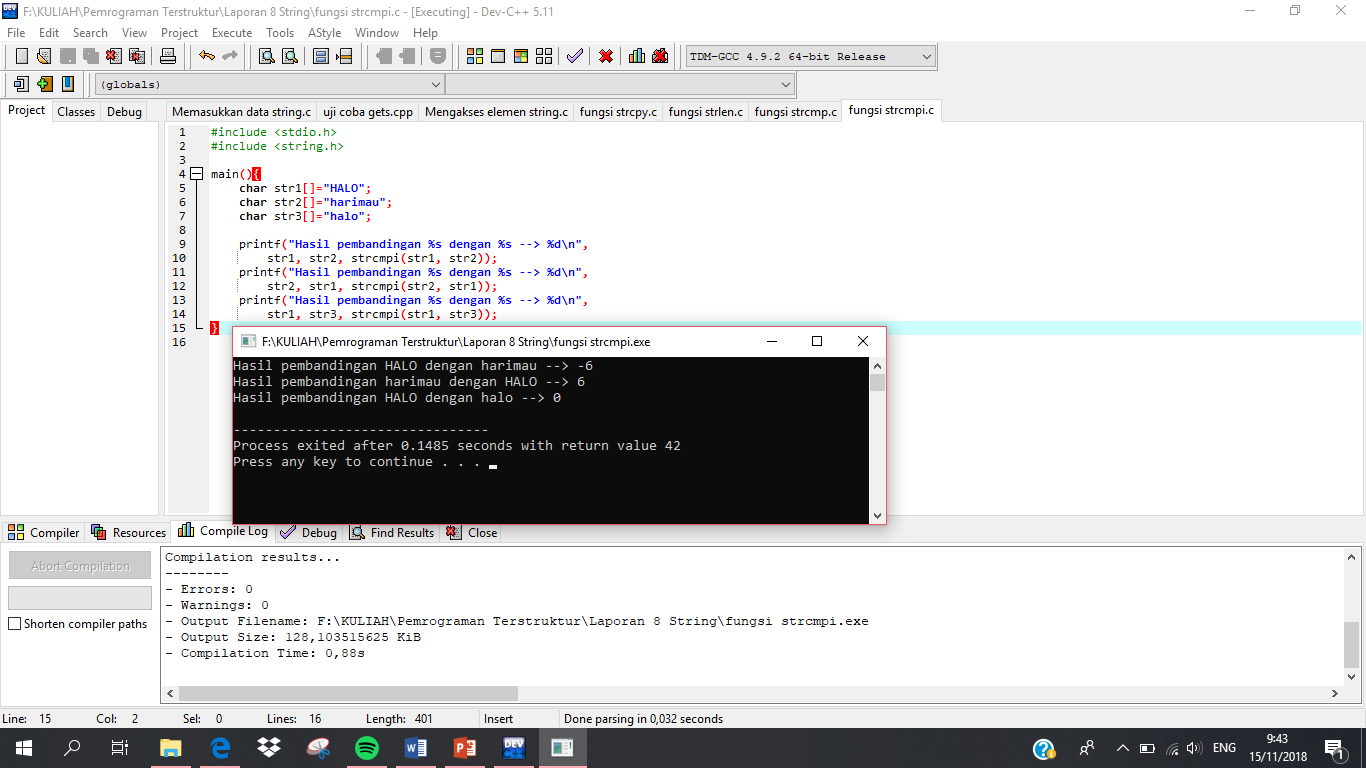
printf("Hasil pembandingan %s dengan %s --> %d\n",

str2, str1, strcmpi(str2, str1));

printf("Hasil pembandingan %s dengan %s --> %d\n",

str1, str3, strcmpi(str1, str3));

}



Analisis dan Kesimpulan

Program tersebut menunjukkan penggunaan fungsi strcmpi(). Dengan menggunakan fungsi strcmpi() () kita dapat membandingkan str 1 dengan str 2 secara **non** **case sensitive**. Untuk hasil apabila -1 berarti str 1 lebih kecil daripada str 2, apabila 1 artinya str 1 lebih besar daripada str 2, apabila hasil 0 berarti str 1 dengan str 2 bernilai sama.

String handling🡪 *User defined function*

1. Lakukan percobaan untuk menginputkan string dari keyboard dengan menggunakan : scanf(), gets() dan fgets(). Analisislah dan berikan kesimpulan untuk setiap fungsi tsb.

1. Scanf()

#include <stdio.h>

main()

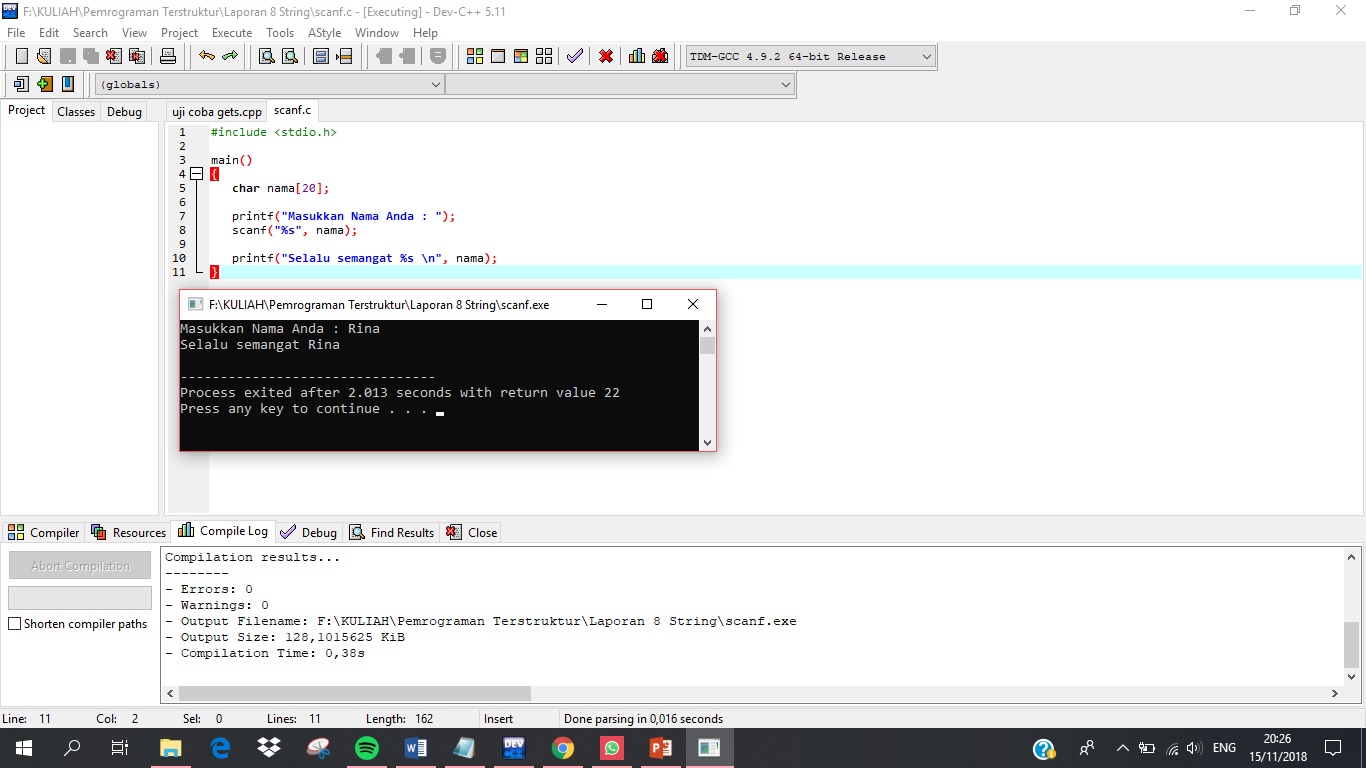
{

char nama[20];

printf("Masukkan Nama Anda : ");

scanf("%s", nama);

printf("Selalu semangat %s \n", nama);

}

1. gets()

#include <stdio.h>

main()

{

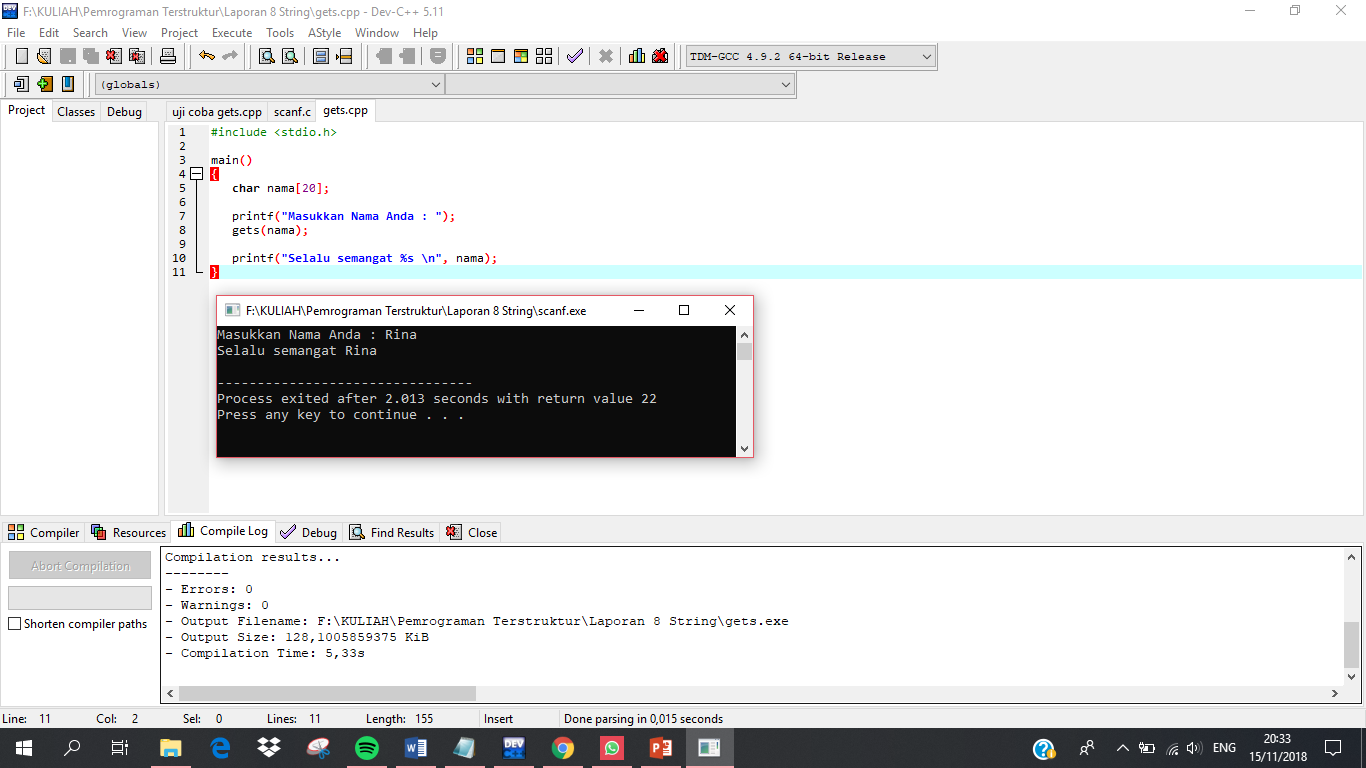
char nama[20];

printf("Masukkan Nama Anda : ");

gets(nama);

printf("Selalu semangat %s \n", nama);

}



1. fgets()

#include <stdio.h>

#include<string.h>

main()

{

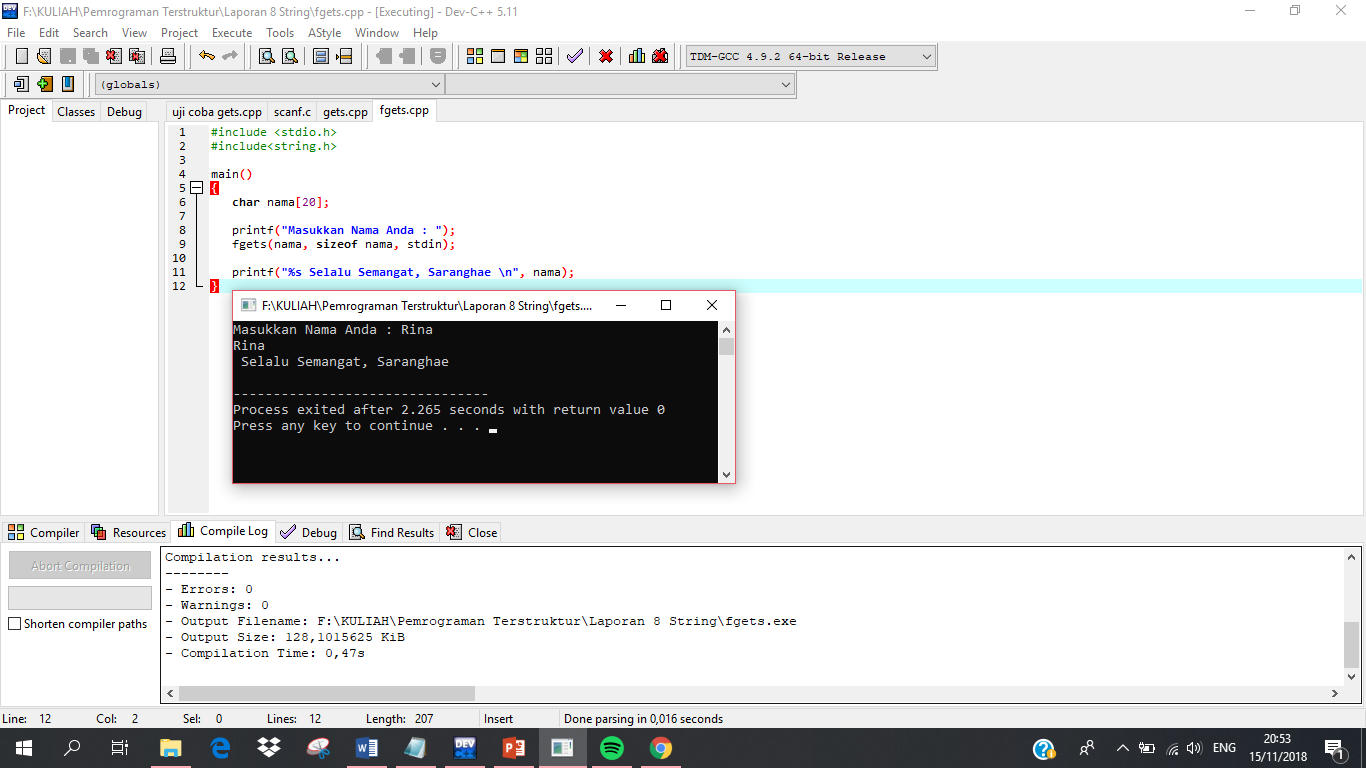
char nama[20];

printf("Masukkan Nama Anda : ");

fgets(nama, sizeof nama, stdin);

printf("%s Selalu Semangat, Saranghae \n", nama);

}



2. Buatlah program untuk menerima input string dari keyboard kemudian hitunglah panjang dari string tsb dan tampilkan hasilnya

#include <stdio.h>

#include <string.h>

char htng\_pnjg(char x[]);

main(){

char kalimat[256];

char hsl[256];

int jumlah,i;

puts("======================================");

puts("PROGRAM MENGHITUNG PANJANG DARI STRING");

puts("======================================");

puts("");

puts("Masukan kalimat (maksimal 255 karakter): ");

gets(kalimat);

jumlah = htng\_pnjg(kalimat);

printf("\nPanjang stringnya adalah %d karakter", jumlah);

}

char htng\_pnjg(char x[]){

int jumkar=0;

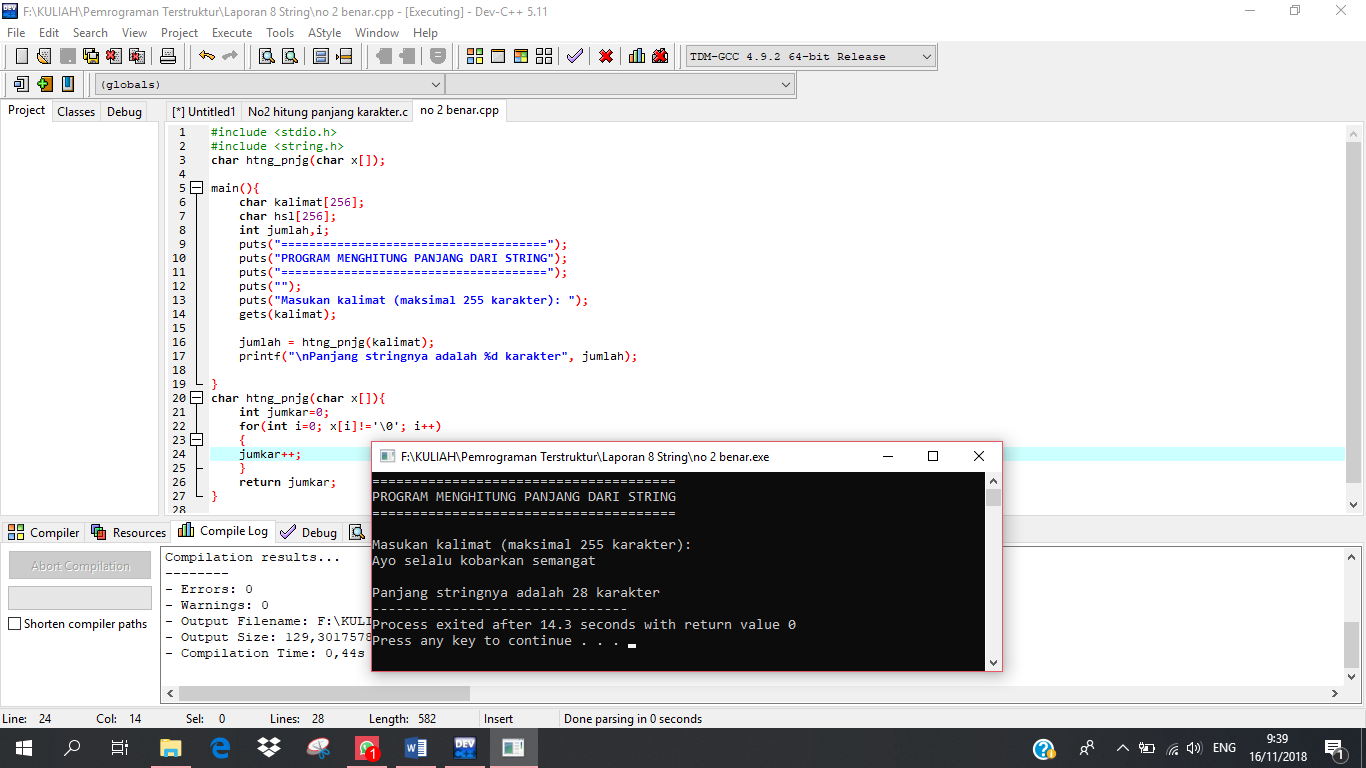
for(int i=0; x[i]!='\0'; i++)

{

jumkar++;

}

return jumkar;

}

3. Lanjutkan program nomor 2 untuk membalik string tsb, misalnya : budi 🡪 ibud

#include <stdio.h>

#include <string.h>

char htng\_pnjg(char x[]);

void balikkata(char y[], char z[]);

main(){

char kalimat[256];

char hsl[256];

int jumlah,i;

puts("======================================");

puts("PROGRAM MENGHITUNG PANJANG DARI STRING");

puts("======================================");

puts("");

puts("Masukan kalimat (maksimal 255 karakter): ");

gets(kalimat);

jumlah = htng\_pnjg(kalimat);

printf("\nPanjang stringnya adalah %d karakter", jumlah);

printf("\nBalik Kalimat = %s ---> ", kalimat);

balikkata(kalimat, hsl);

}

char htng\_pnjg(char x[]){

int jumkar=0;

for(int i=0; x[i]!='\0'; i++)

{

jumkar++;

}

return jumkar;

}

void balikkata(char x[], char y[]){

int pnjg, akhir;

int awal;

pnjg = htng\_pnjg( x);

akhir = pnjg - 1;

for(awal = 0 ; awal < pnjg; awal++) {

y[awal]=x[akhir];

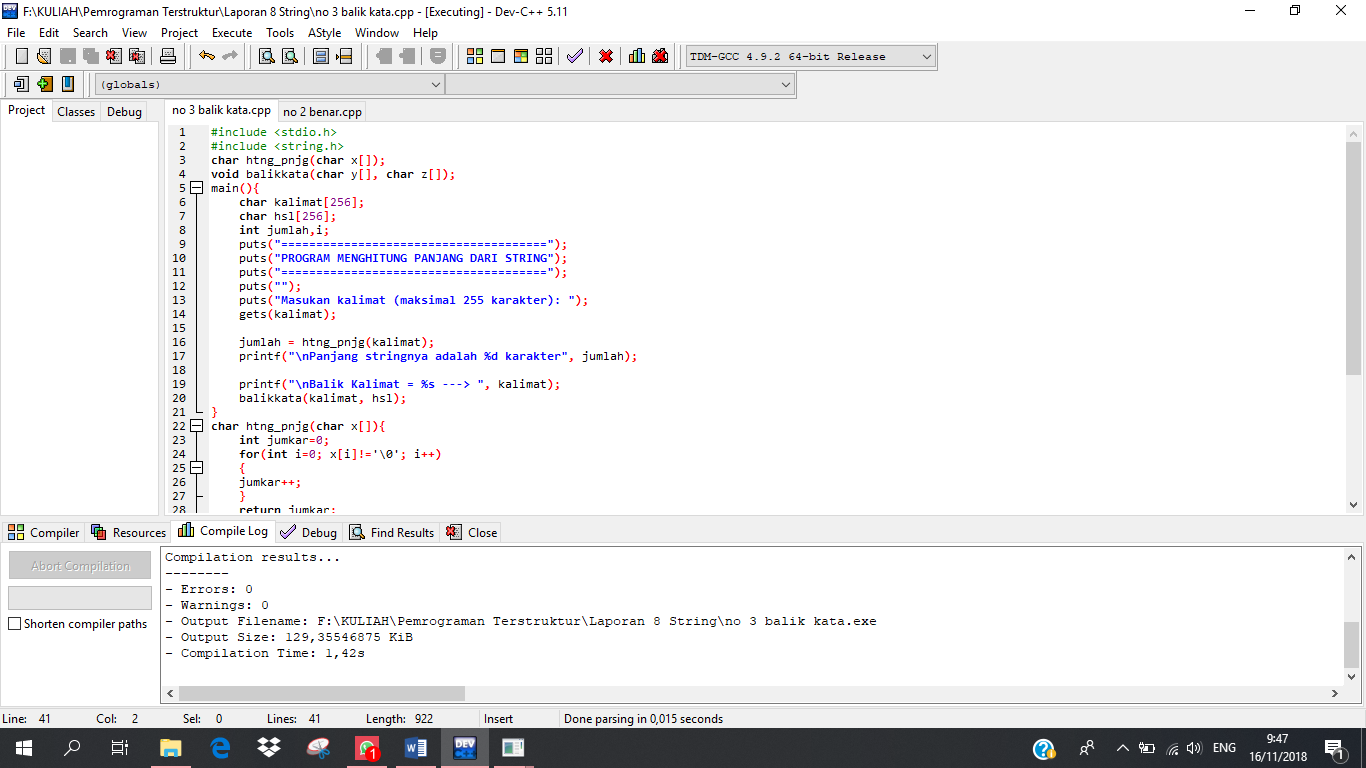
akhir--;

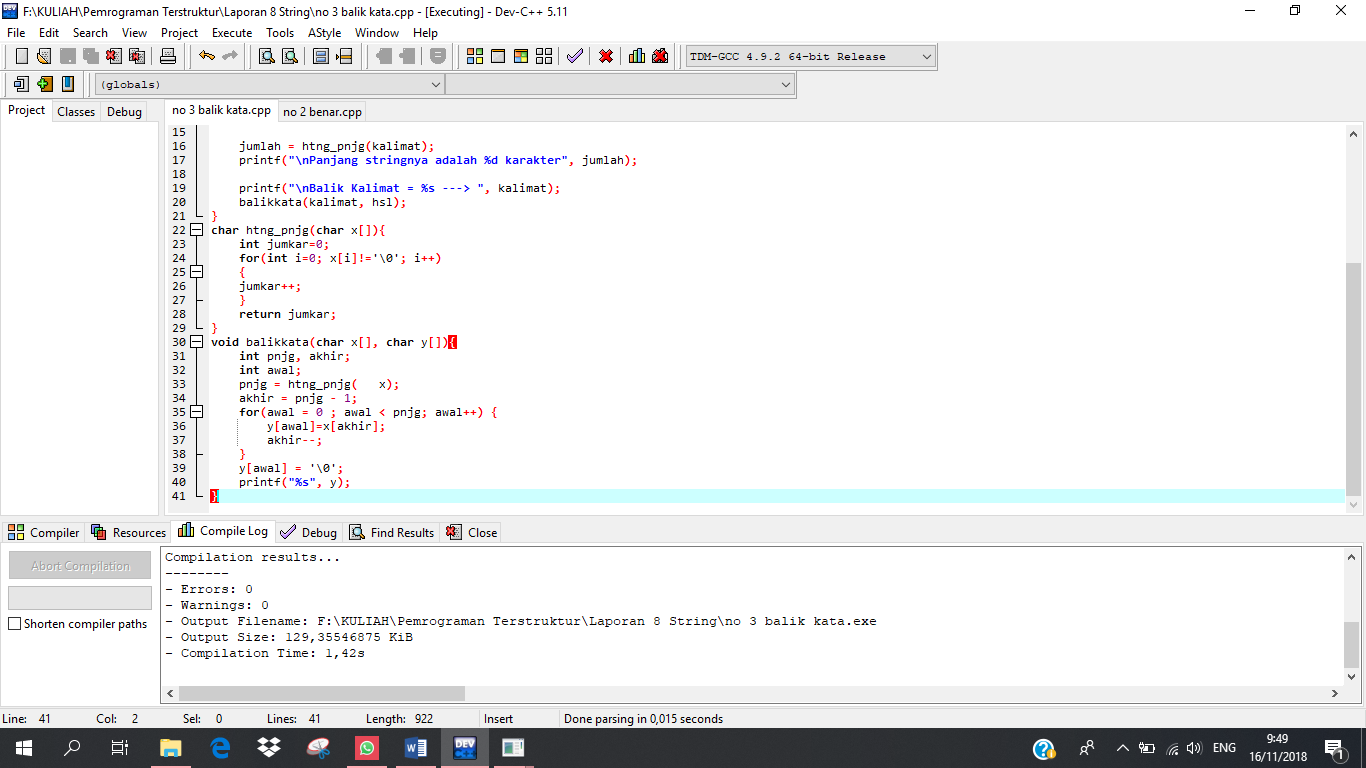
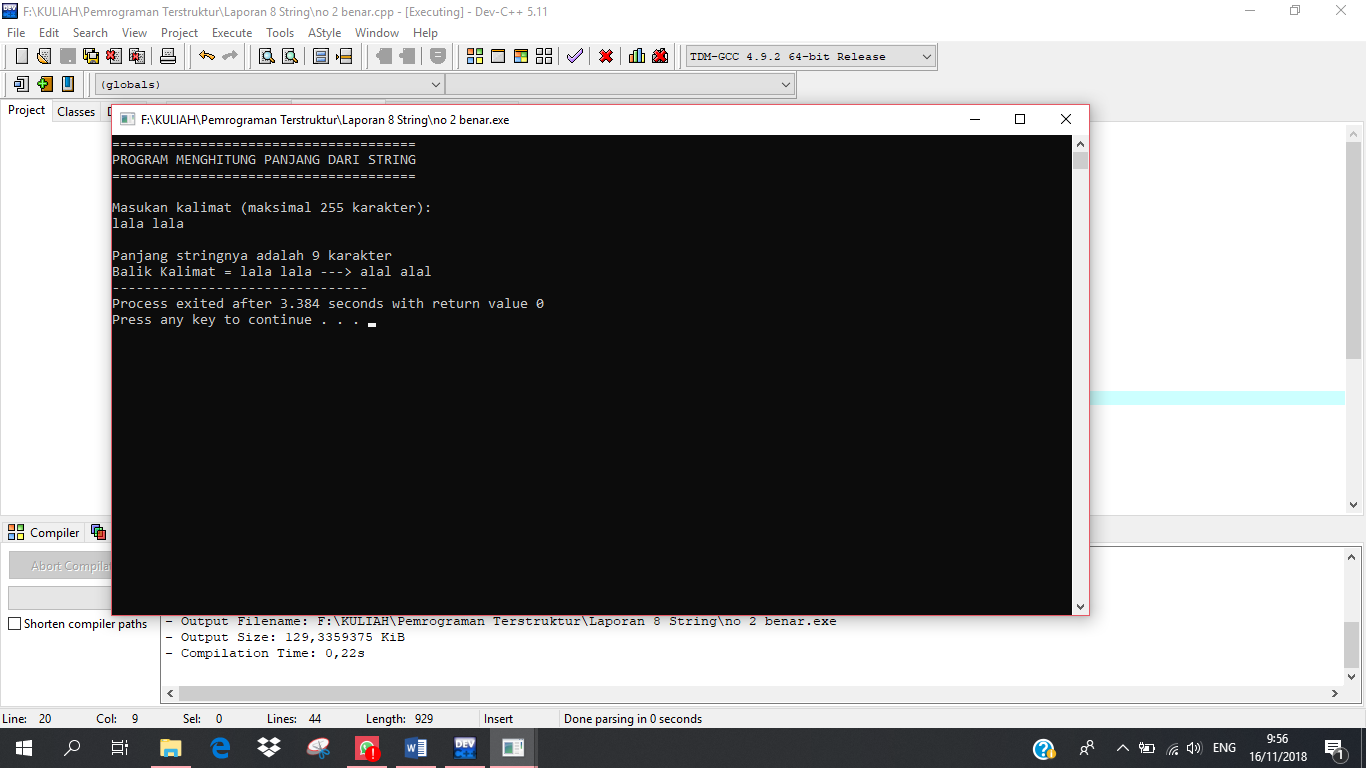
}

y[awal] = '\0';

printf("%s", y);

}





4. Buatlah program yang mendeklarasikan sekaligus menginisialisasi sebuah array kata1[], kemudian copy-lah isi array kata1[] tsb ke dalam array kata2[], selanjutnya tampilkan isi kedua array tsb ke layer

#include<stdio.h>

#include<string.h>

void copystr(char\*,char\*);

int main(){

char asal[50]=""; char tujuan[50]="";

printf("Masukkan kata 1 : "); fgets(asal,sizeof asal, stdin);

copystr(tujuan,asal);

printf("kata 2 : %s\n", tujuan);

printf("kata 1 : %s\n", asal);

}

int pjgstr(char\*string){

int i=0;

while(string[i]!='\0'){

i++;

}

return i;

}

void copystr(char\*tujuan, char\*asal){

int i=0, m=pjgstr(asal)-1;

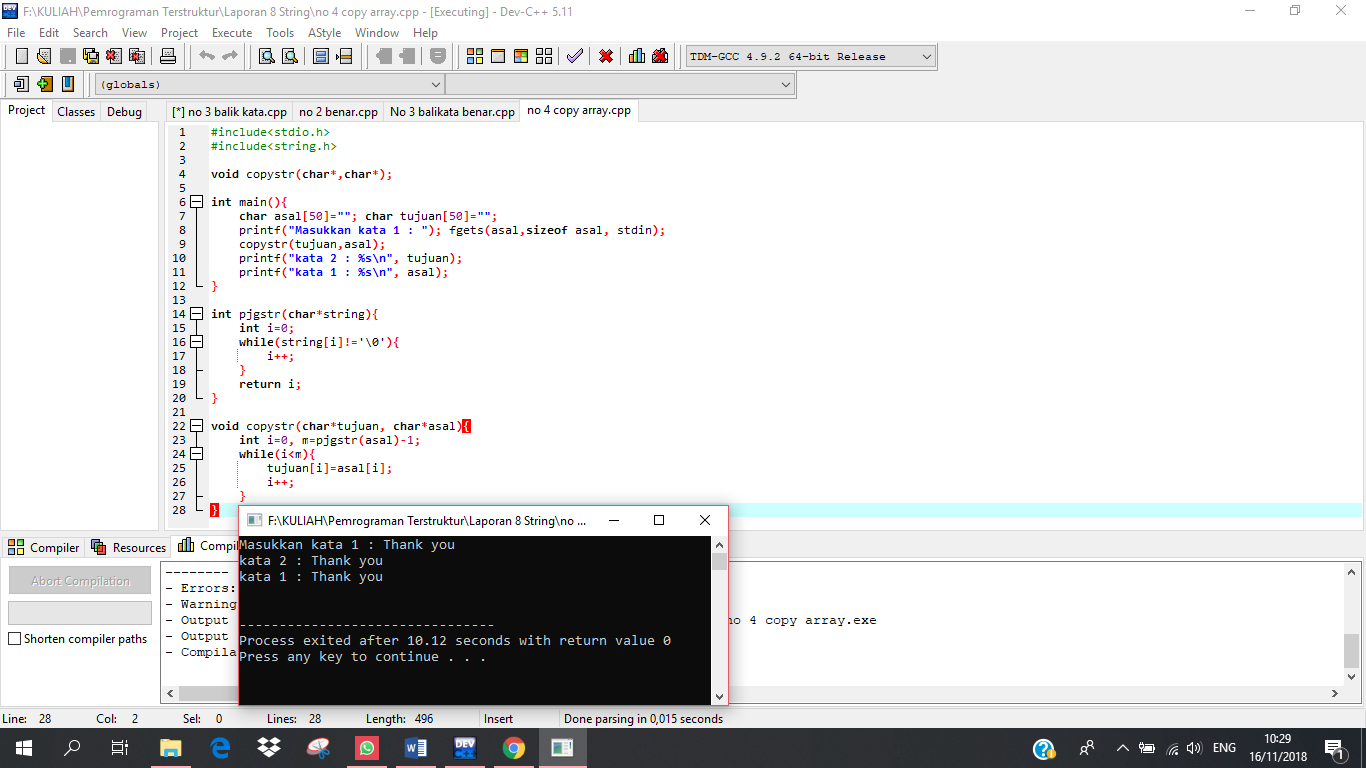
while(i<m){

tujuan[i]=asal[i];

i++;

}

}



5. Ulangilah soal nomor 2, 3 & 4 di atas dengan menggunakan fungsi-fungsi standard

#include<stdio.h>

#include<string.h>

char kar1[200];

char kar2[200];

main()

{

printf("Masukkan kar 1=");

gets(kar1);

printf("\nkarakter %s memiliki panjang %d",kar1,strlen(kar1));

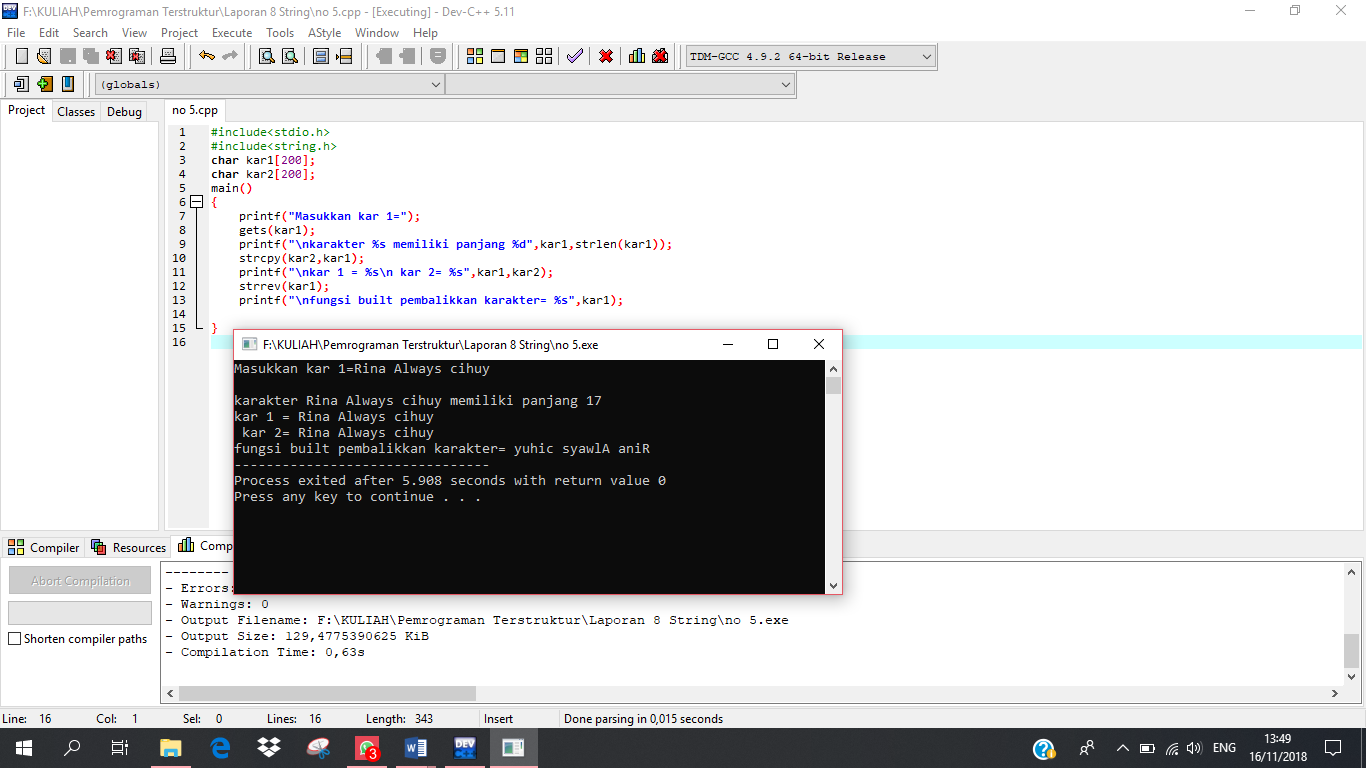
strcpy(kar2,kar1);

printf("\nkar 1 = %s\n kar 2= %s",kar1,kar2);

strrev(kar1);

printf("\nfungsi built pembalikkan karakter= %s",kar1);

}



6. Lakukan percobaan untuk membandingkan 2 buah string dengan menggunakan fungsi strcmp() dan strcmpi(). Analisislah dan berikan kesimpulan tentang perbedaan dan contoh aplikasi untuk keduanya.

#include <stdio.h>

#include <string.h>

main(){

char str1[]="slanK";

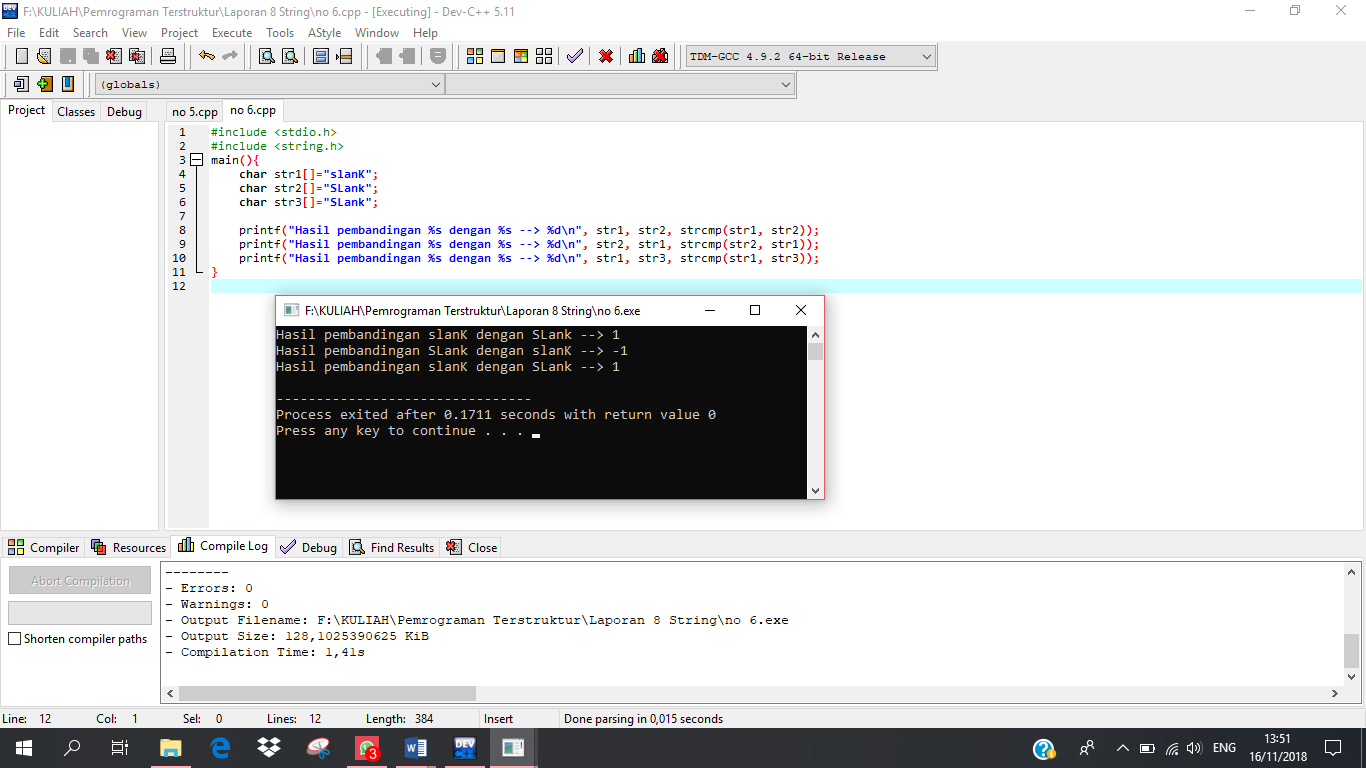
char str2[]="SLank";

char str3[]="SLank";

printf("Hasil pembandingan %s dengan %s --> %d\n", str1, str2, strcmp(str1, str2));

printf("Hasil pembandingan %s dengan %s --> %d\n", str2, str1, strcmp(str2, str1));

printf("Hasil pembandingan %s dengan %s --> %d\n", str1, str3, strcmp(str1, str3));

}

1. **Kesimpulan**
2. Pada hakikatnya, STRING merupakan sekumpulan karakter, sehingga sebuah STRING dapat pila kita panggil sebagai array of char.
3. Untuk membaca data string dari user, kita dapat menggunakan fungsi scanf,gets,dan fgets
4. Setiap string memiliki elemen tambahan ‘\0’ pada karakter terakhir sebagai penanda akhir dari string tersebut.
5. **Penutup**

Demikian laporan ini saya buat, saya sadar laporan ini masih jauh dari sempurna. Untuk itu saya mohon maaf apabila ada kekurangan dalam laporan yang saya buat.