**LAPORAN JOBSHEET 5**

**Fungsi**



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**

**BAB 5**

**FUNGSI**

### TUJUAN

* 1. Melakukan pemanggilan fungsi dengan nilai dan referensi.
  2. Menggolongkan variabel berdasarkan kelas penyimpanan.
  3. Menerapkan fungsi rekursi

### DASAR TEORI

Fungsi merupakan bagian (blok) dari kode program yang dirancang untuk melaksanakan tugas tertentu. Fungsi merupakan bagian penting dalam pemrograman C, dengan tujuan:

* 1. Program menjadi terstruktur, sehingga mudah dipahami dan mudah dikembangkan.
  2. Dapat mengurangi pengulangan kode (duplikasi kode)

Ada beberapa contoh fungsi standar yang sudah pernah kita gunakan dan memiliki tugas khusus, seperti: printf(),getch(), puts() dan masih banyak lagi fungsi lain yang bisa kita pakai sesuai tugasnya. Disamping itu kita juga dapat membuat fungsi sendiri sesuai dengan kebutuhan kita. Dalam bekerja dengan fungsi kita memerlukan masukan yang dinamakan sebagai argumen dan parameter. Masukan ini selanjutnya diolah oleh fungsi dan hasil keluarannya dapat berupa sebuah nilai (nilai balik fungsi). Bentuk umum sebuah fungsi adalah sebagai berikut:

Penentu\_tipe nama\_fungsi (daftar parameter) Deklarasi parameter

{

Tubuh fungsi

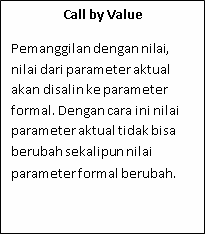
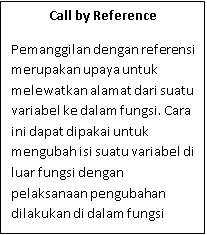
}

Penentu tipe digunakan untuk menentukan tipe keluaran fungsi, defaultnya int.

1. Melewatkan Parameter

Ada 2 cara melewatkan parameter ke dalam fungsi :

1. Pemanggilan dengan nilai *(call by value)*
2. Pemanggilan dengan referensi *(call by reference)*



1. Penggolongan Variabel Berdasarkan Kelas Penyimpanan

Suatu variabel disamping dapat digolongkan berdasarkan jenis/tipe data juga dapat diklasifikasikan berdasarkan kelas penyimpanan berupa :

* Variabel lokal
* Variabel eksternal
* Variabel statis
* Variabel register

Variabel lokal adalah variabel yang dideklarasikan dalam fungsi.

Contoh:

Void fung\_x(void)

{

x Int x;

adalah variabel lokal x

.......

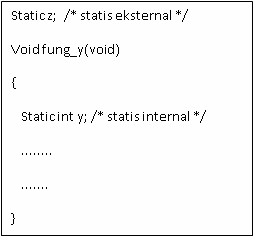
.......

}

adalah variabel lokal bagi fung\_x

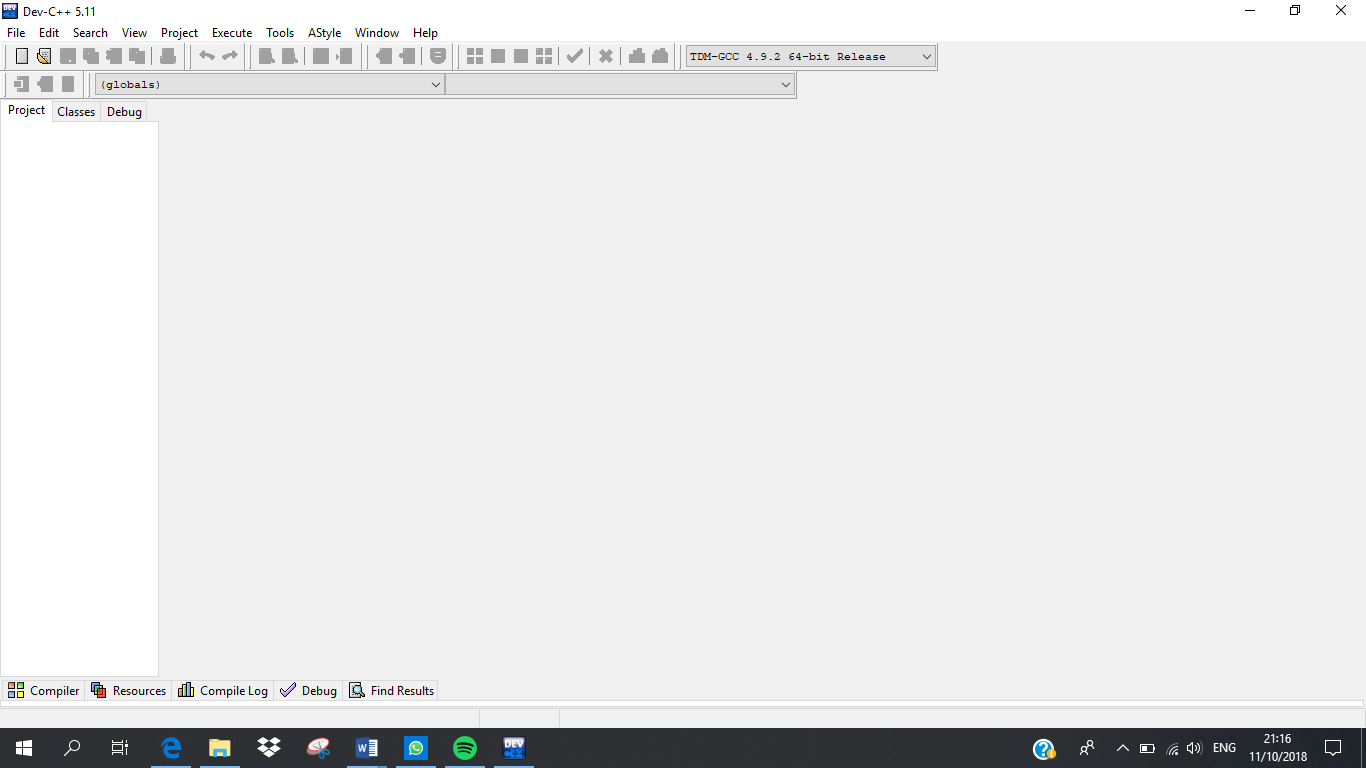
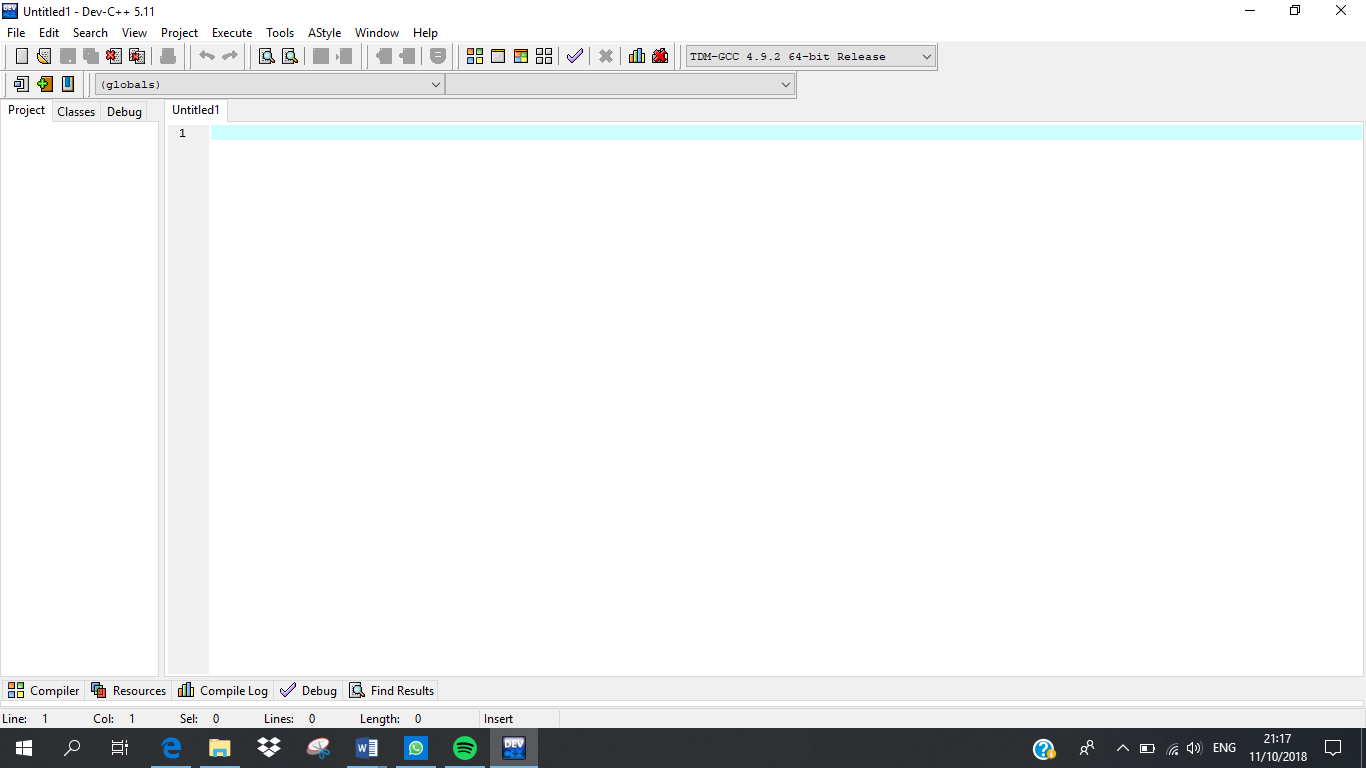
Variabel eksternal merupakan variabel yang dideklarasikan di luar fungsi. Variabel statis dapat berupa variabel internal (didefinisikan di dalam fungsi) maupun variabel eksternal.

Contoh :



Variabel register adalah variabel yang nilainya disimpan dalam register bukan pada memori RAM. Diterapkan pada variabel yang lokal atau parameter formal, yang bertipe char atau int.

### LANGKAH KERJA

1. **Langkah pemanggilan dengan nilai**
   1. Jalankan Dev C++
   2. Alt+F New
   3. Ketik :

#include<stdio.h>

#include<conio.h>

void tukar (int, int);

main()

{

int a, b;

a = 88;

b = 77;

printf("nilai sebelum pemanggilan fungsi\n");

printf("a = %d b = %d\n\n", a, b);

tukar(a, b);

printf("nilai sesudah pemanggilan fungsi\n");

printf("a = %d b = %d\n\n", a, b);

getch();

}

void tukar (int x, int y)

{

int z;

/\* variabel sementara \*/

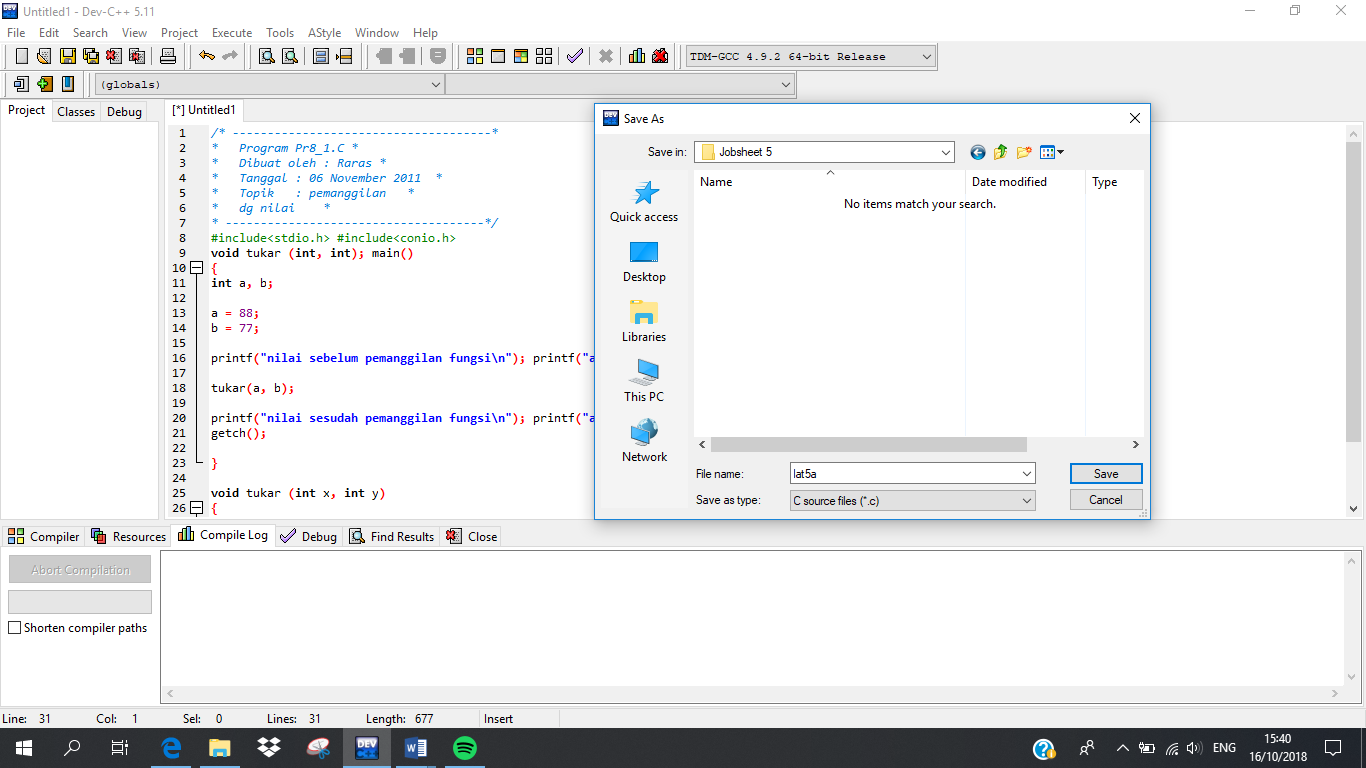
z = x;

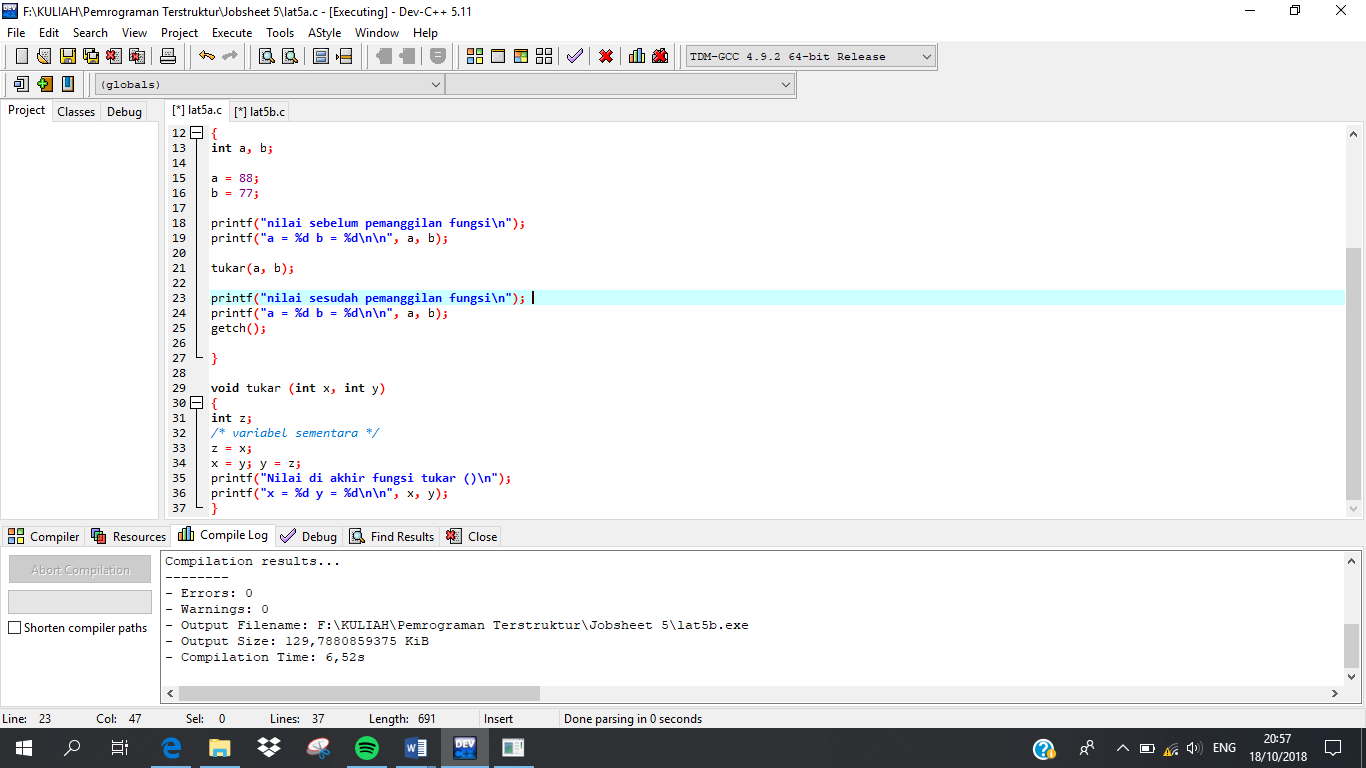
x = y; y = z;

printf("Nilai di akhir fungsi tukar ()\n");

printf("x = %d y = %d\n\n", x, y);

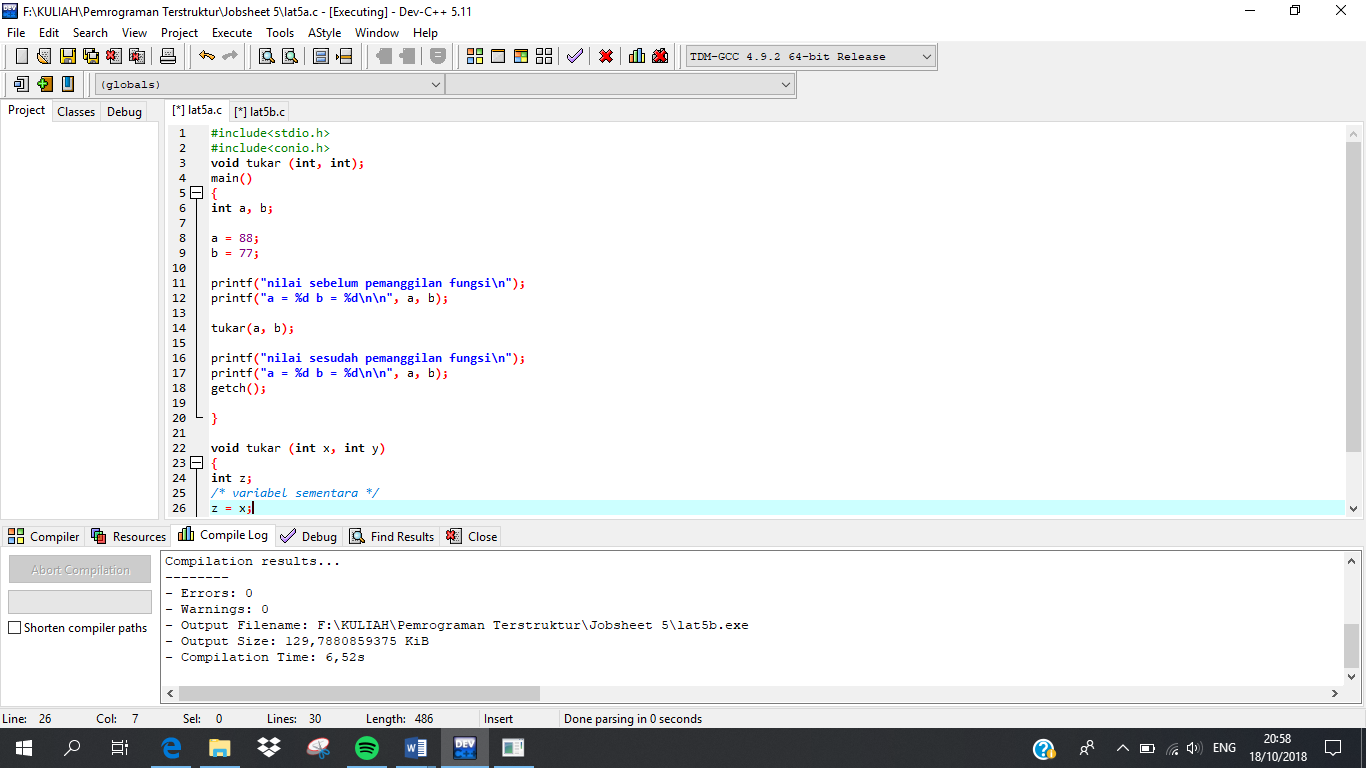
}

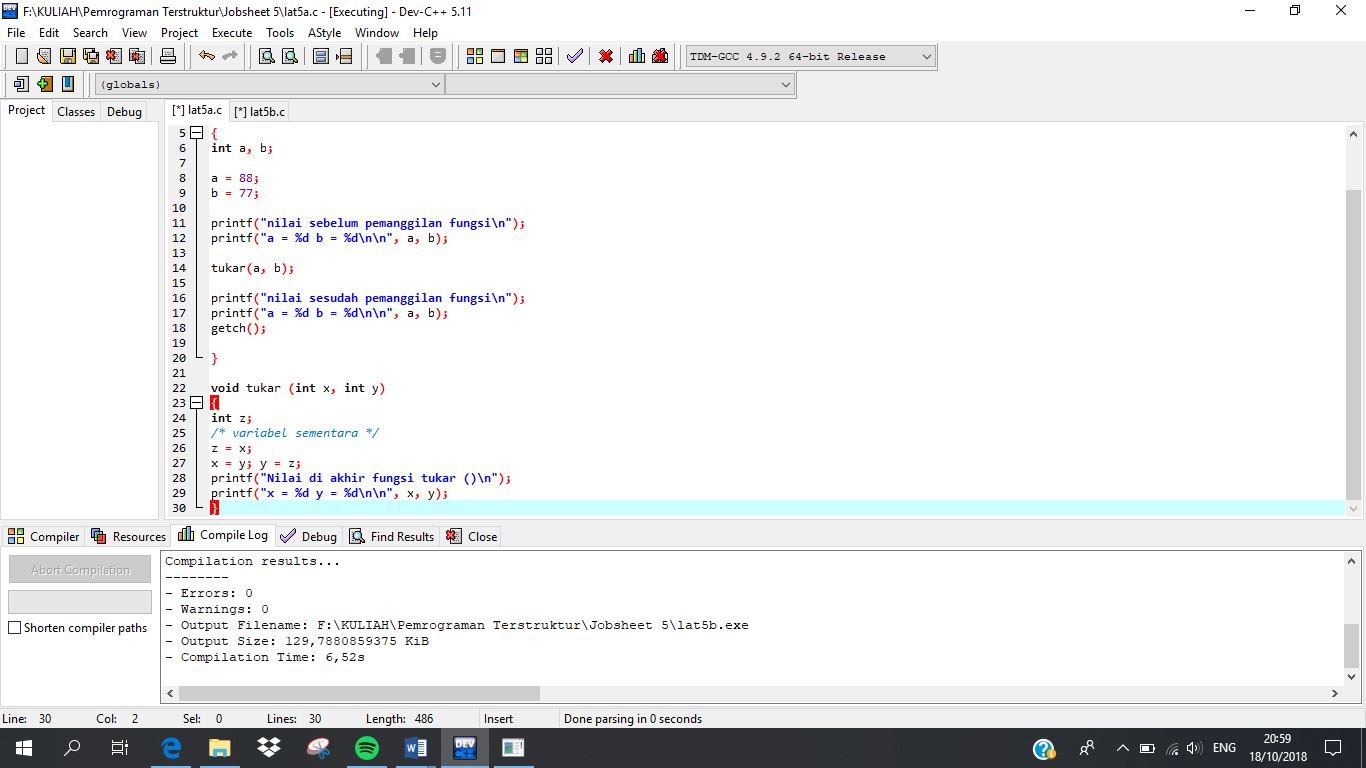
* 1. Alt+F, save dengan nama Lat5a.c
  2. Lalu Compile hingga tidak terdapat error dan dapat di run.

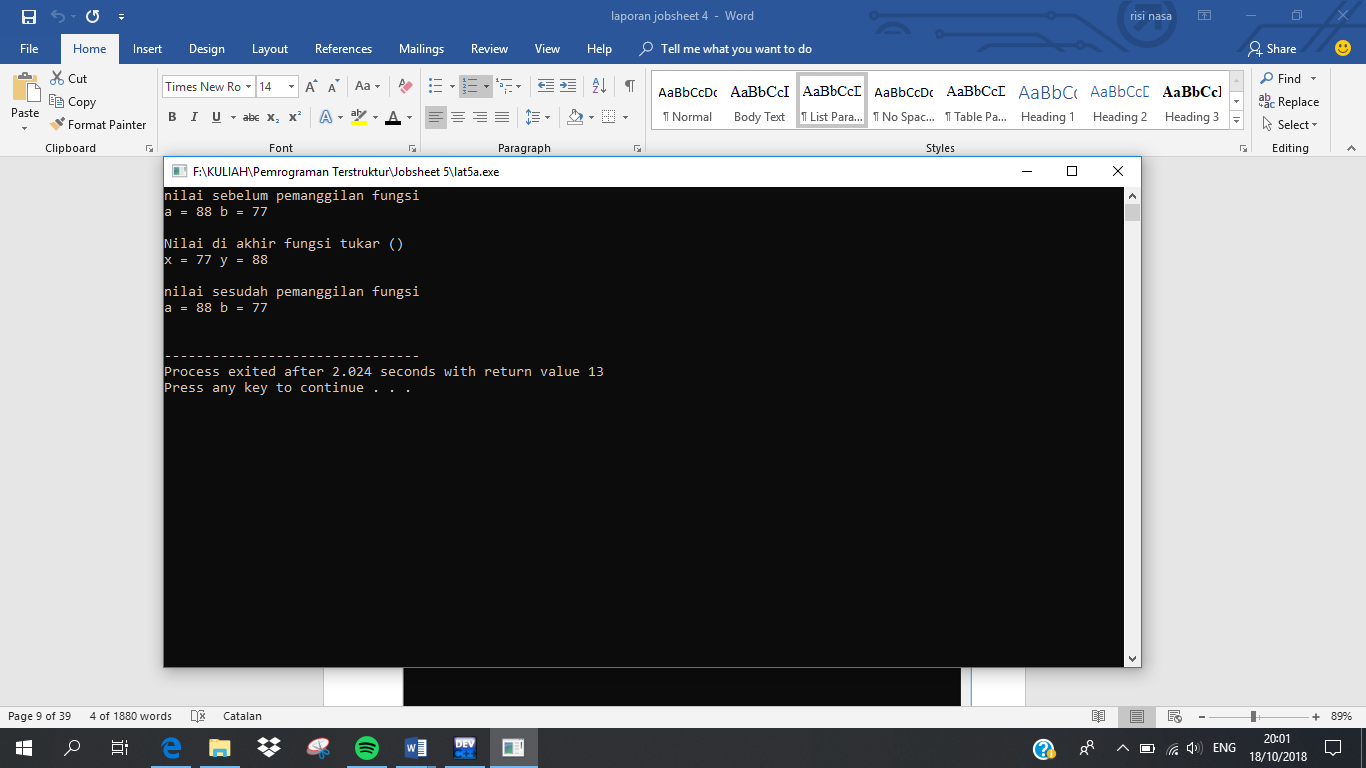


Setelah di compile tidak terdapat error

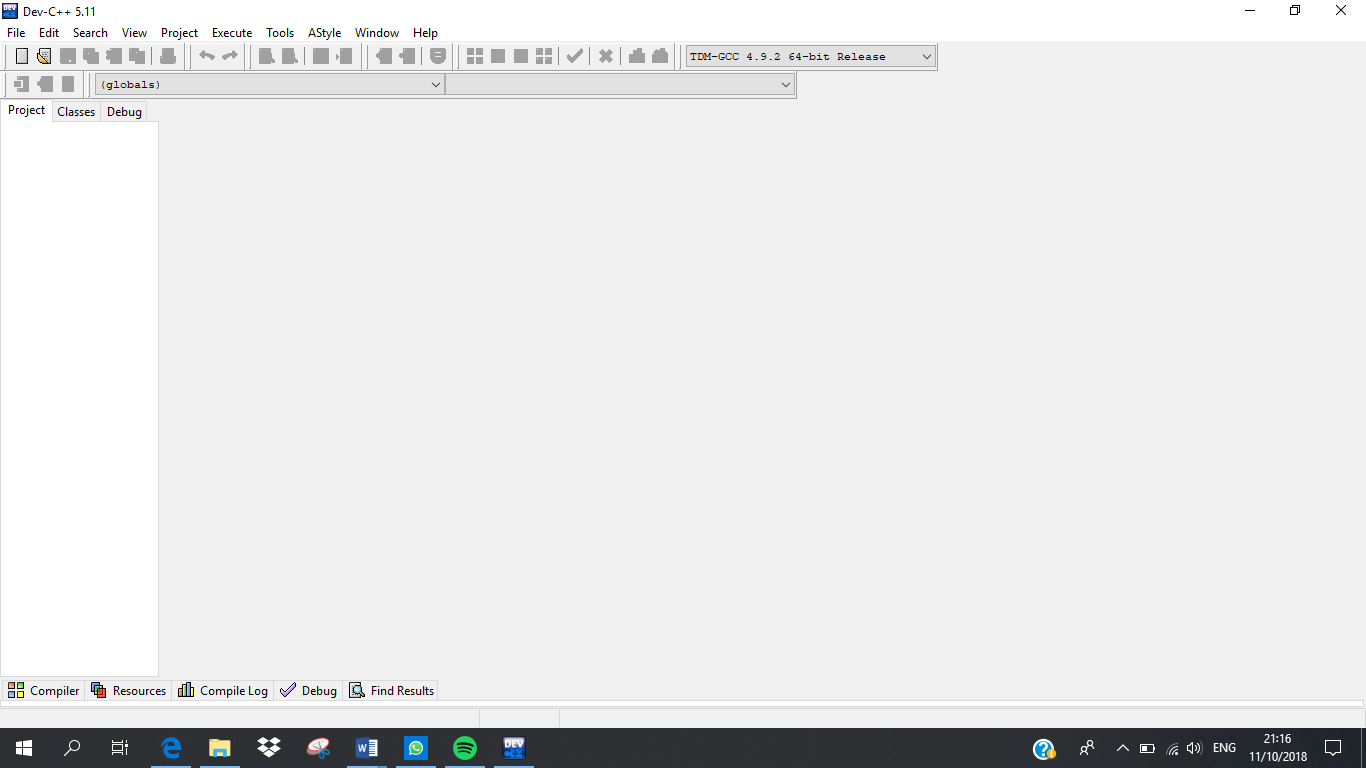
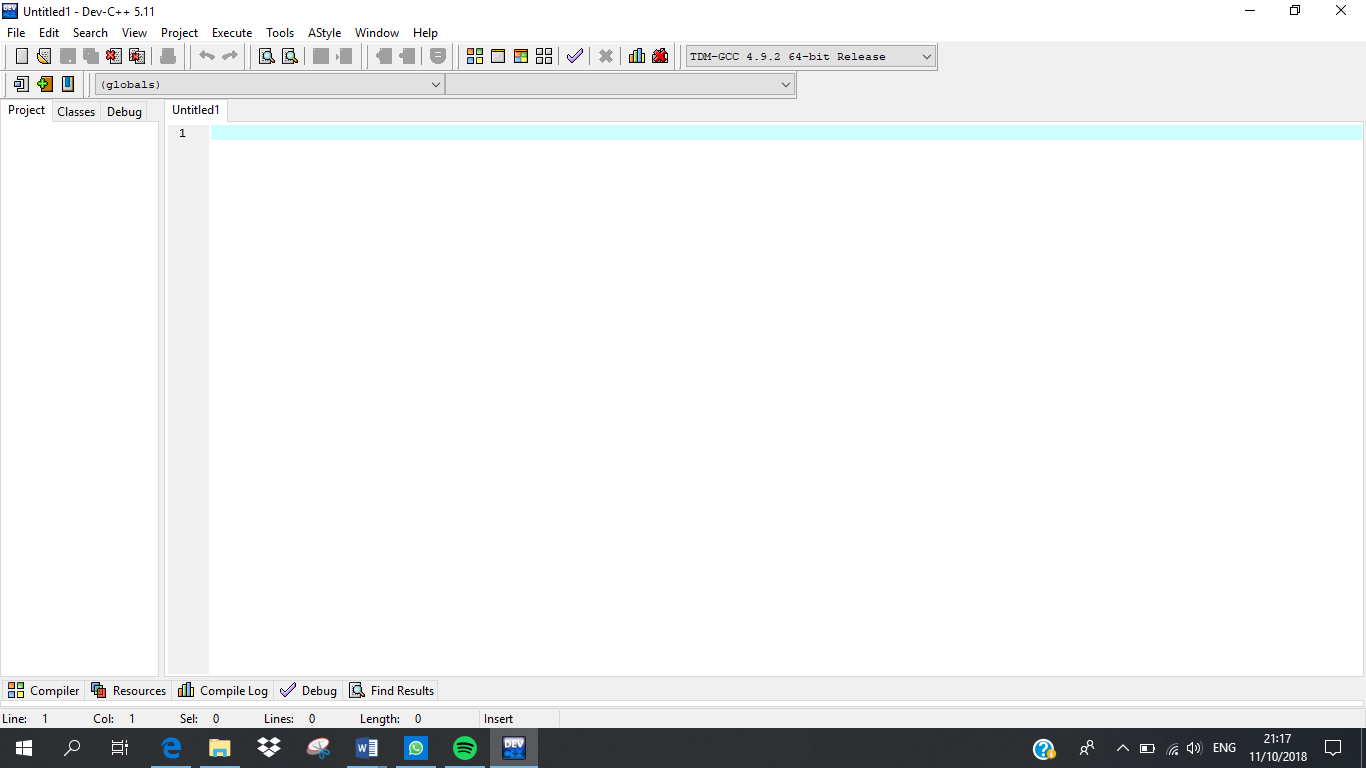
* 1. Run program tersebut.





Hasil Program

### Langkah melakukan pemanggilan dengan referensi :

* 1. Jalankan Dev C++
  2. Alt+F New
  3. Ketik :

/\* -------------------------------------\*

\* Program lat5b.c \*

\* Dibuat oleh : Raras \*

\* Tanggal : 06 November 2011 \*

\* Topik : pemanggilan \*

\* dg referensi \*

\* -------------------------------------\*/

#include <stdio.h>

#include <conio.h>

void tukar (int \*px, int \*py);

main()

{

int a, b;

a = 88;

b = 77;

printf("nilai sebelum pemanggilan fungsi\n");

printf("a = %d b = %d\n\n", a, b);

tukar(&a, &b);

/\*parameter : alamat a dan alamat b\*/

printf("nilai sesudah pemanggilan fungsi\n");

printf("a = %d b = %d\n\n", a, b);

getch();

}

void tukar (int \*px, int \*py)

{

int z;

/\*variabel sementara\*/

z = \*px;

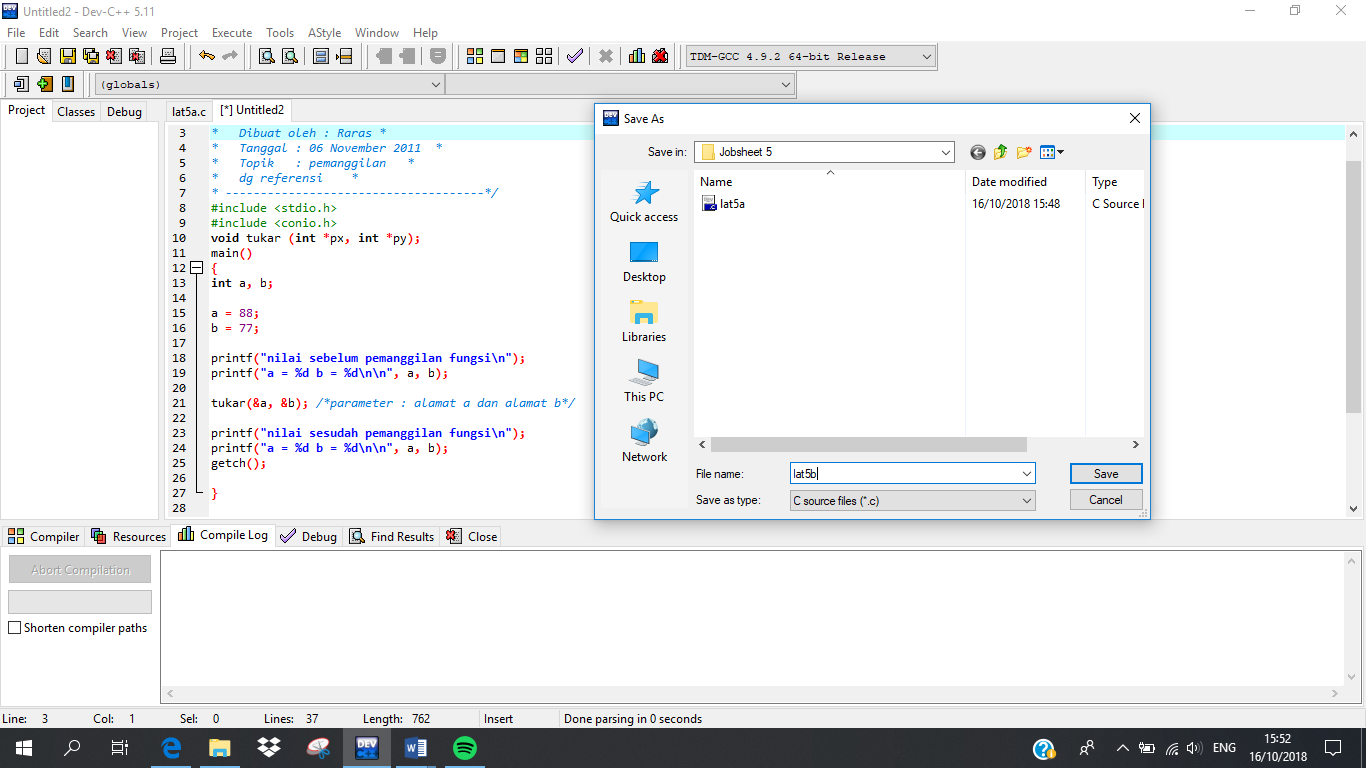
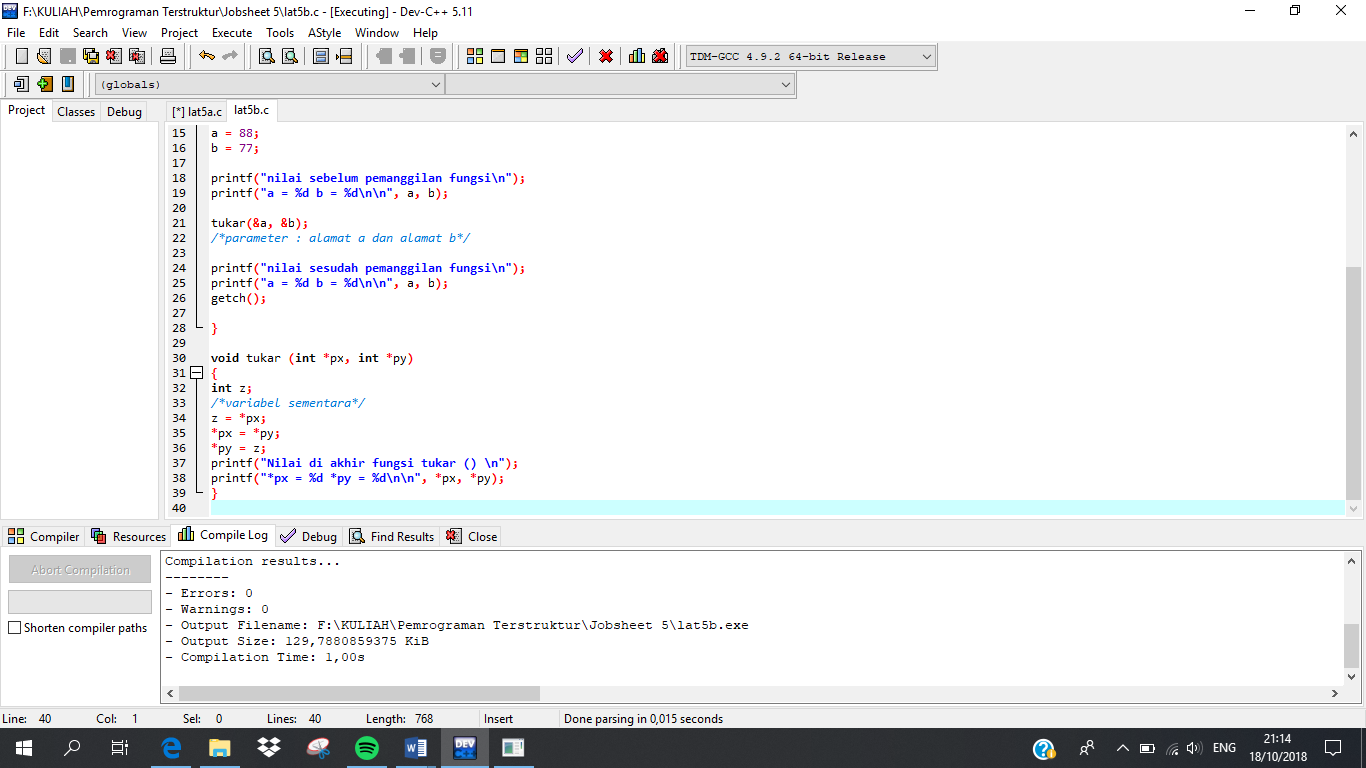
\*px = \*py;

\*py = z;

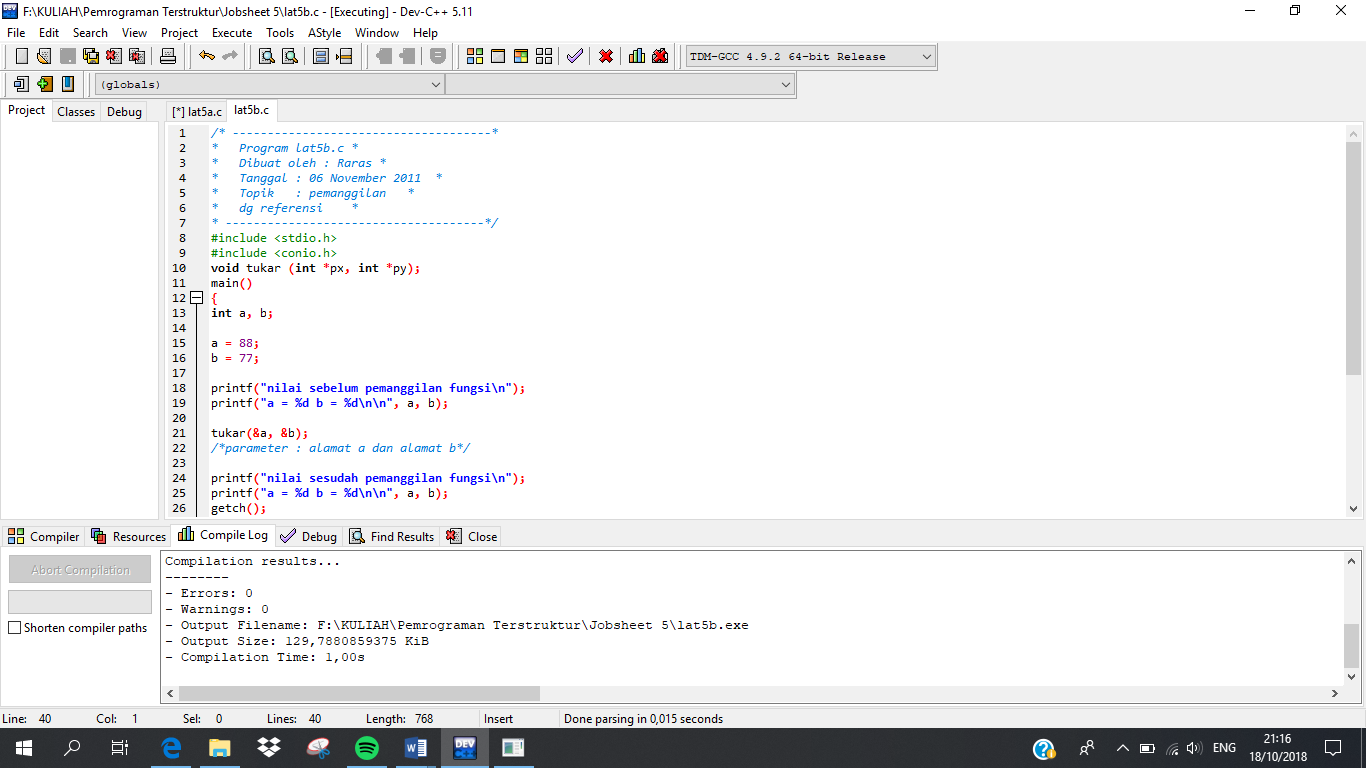
printf("Nilai di akhir fungsi tukar () \n");

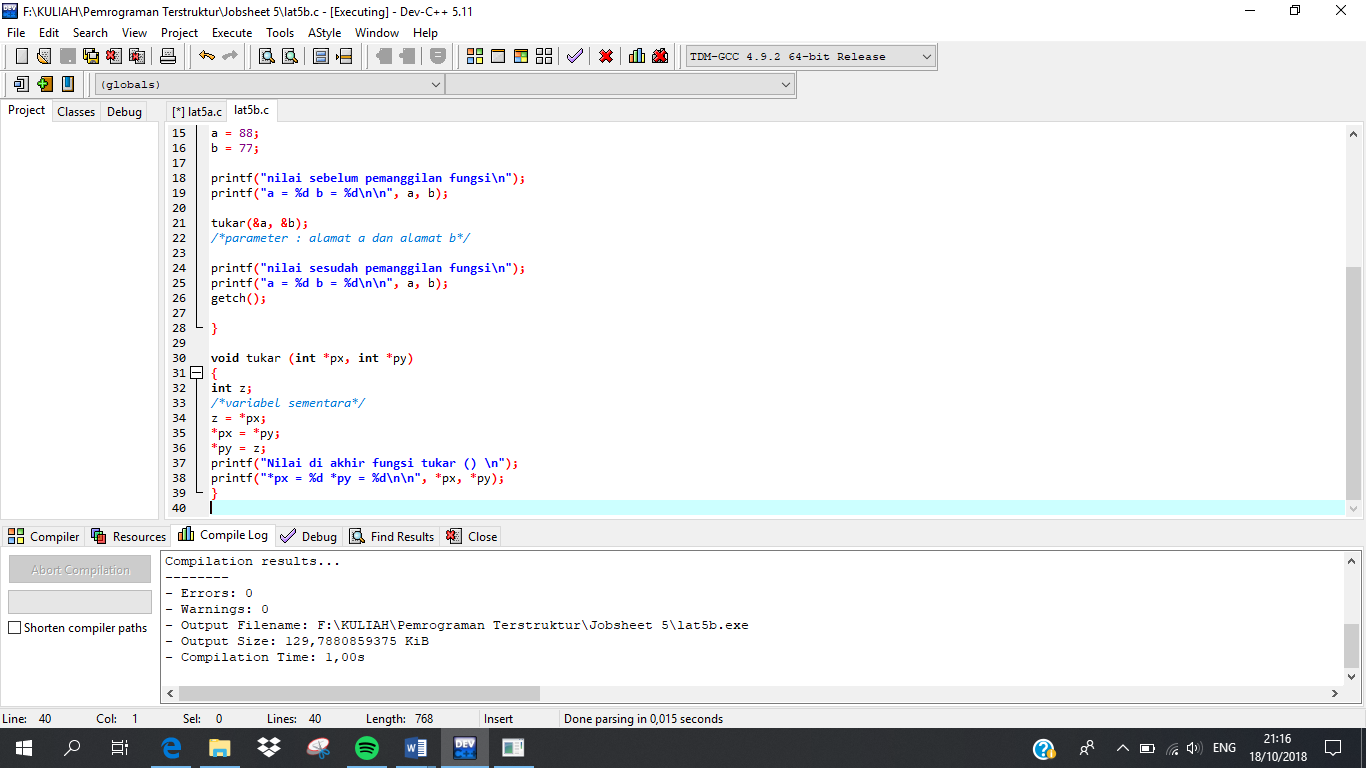
printf("\*px = %d \*py = %d\n\n", \*px, \*py);

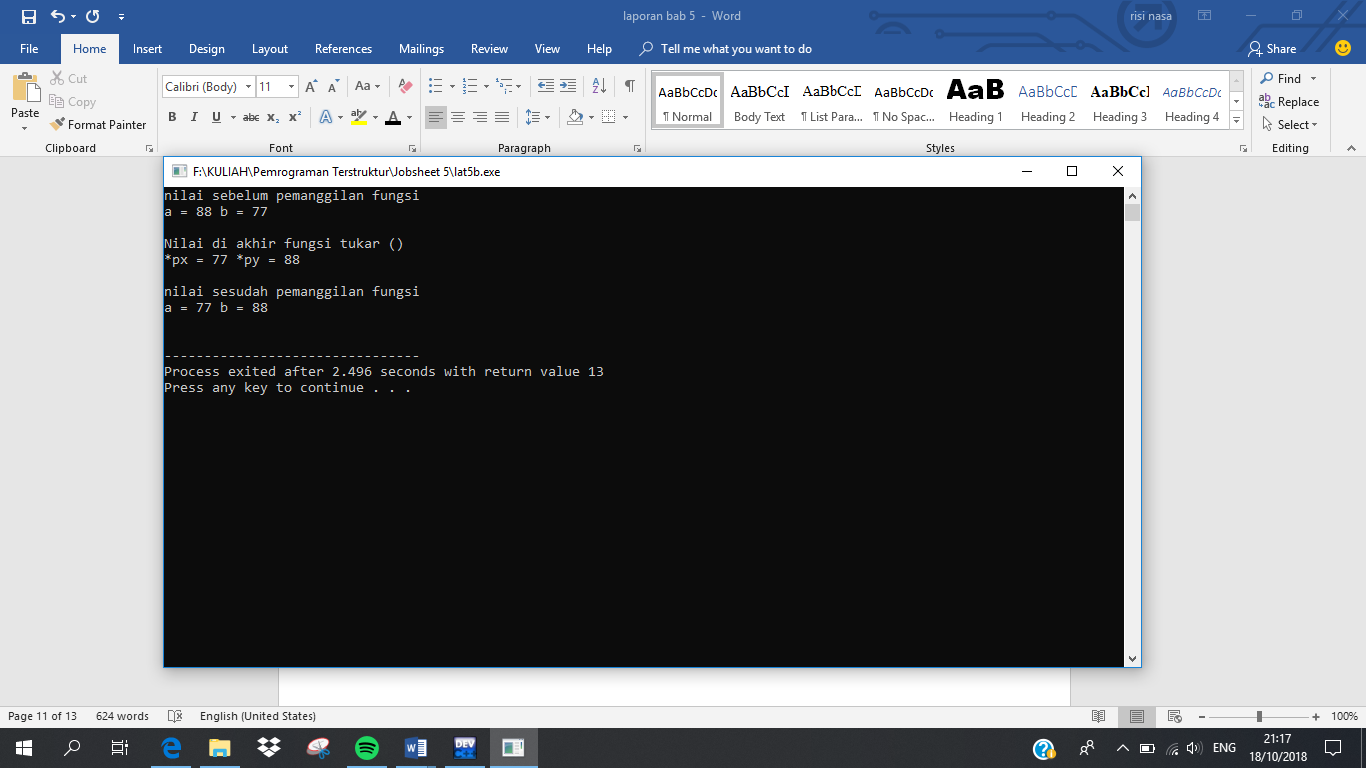
}

* 1. Alt+F, save dengan nama Lat5b.c
  2. Lalu Compile hingga tidak terdapat error dan dapat di run.

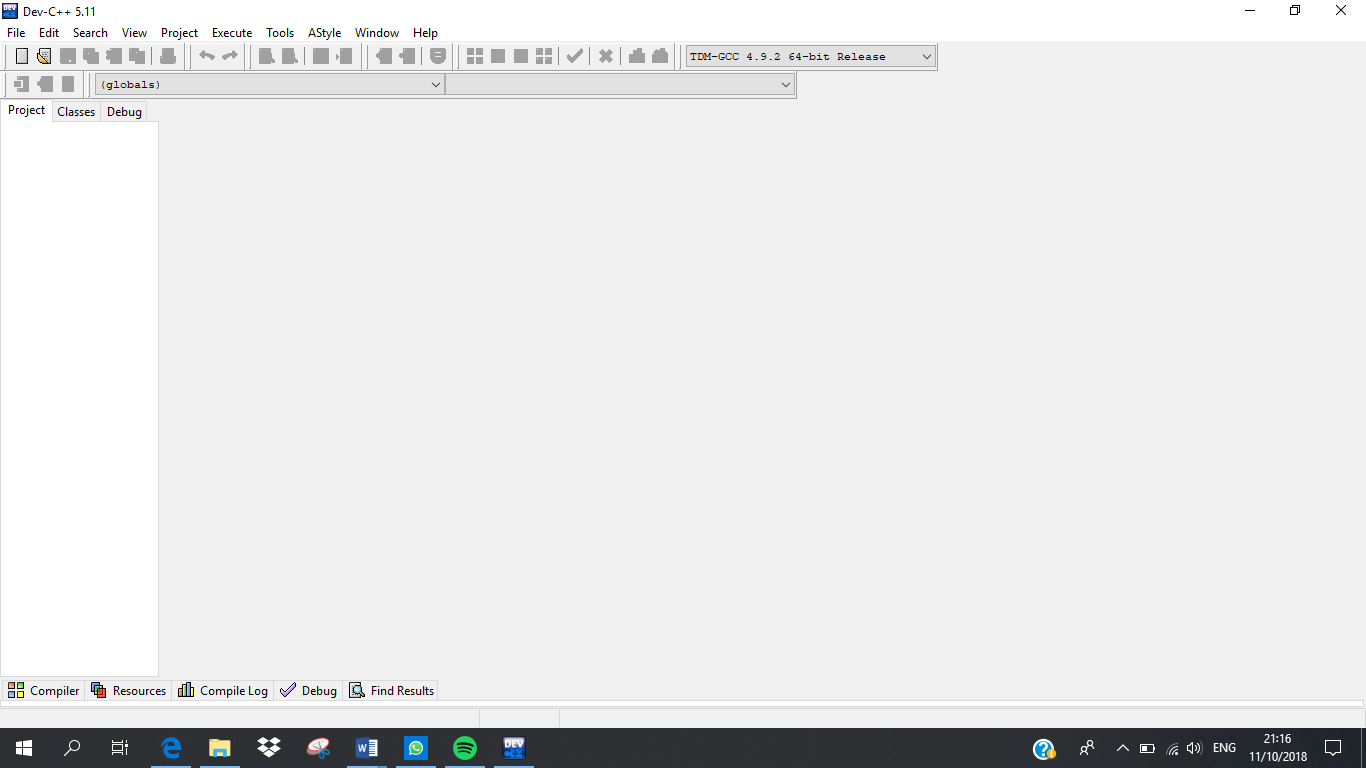
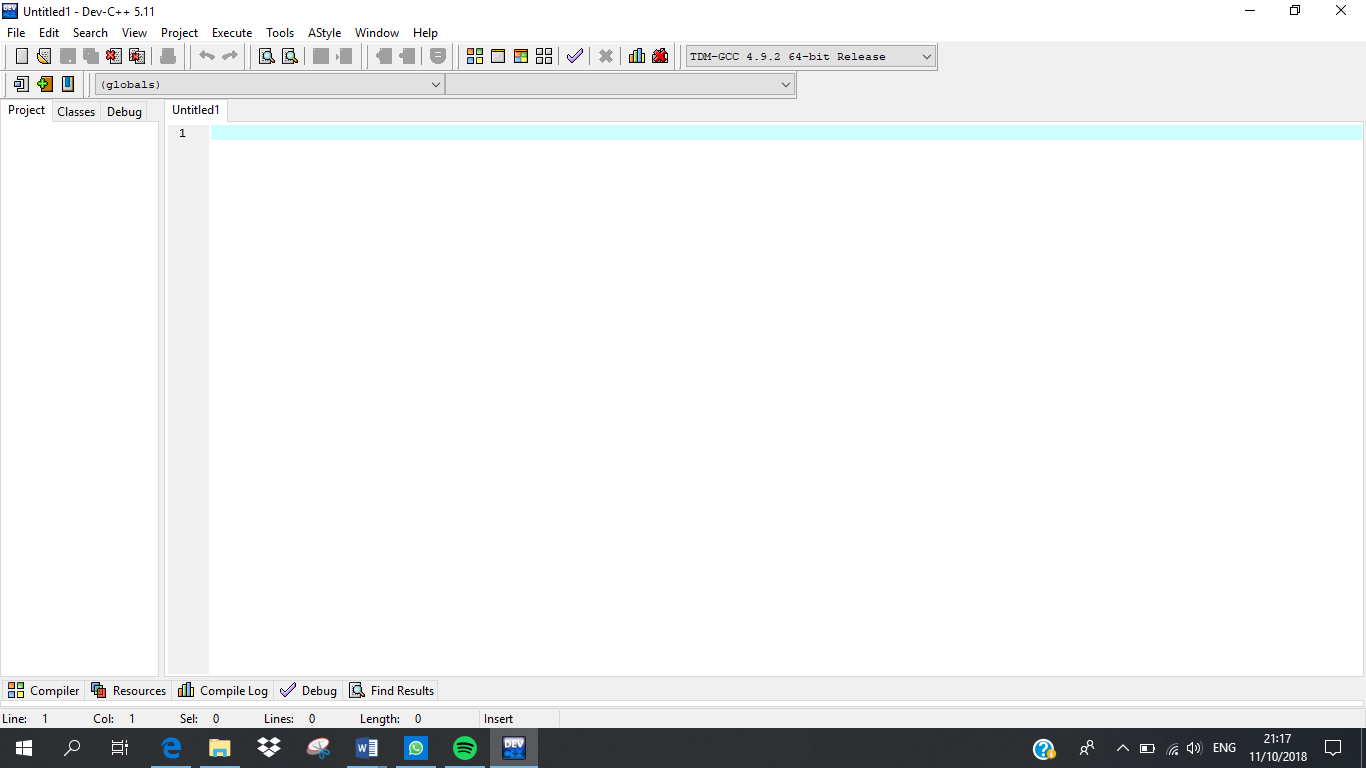
Setelah di compile tidak terdapat error

* 1. Run program tersebut.



Hasil Program

### Langkah menghasilkan fungsi rekursi (fungsi yang memanggil dirinya sendiri)

* 1. Jalankan Dev C++
  2. Alt+F New
  3. Ketik :

/\* ------------------------------------ \*

\* Program lat5c.c \*

\* Dibuat oleh : Raras \*

\* Tanggal : 06 November 2011 \*

\* Topik : fungsi rekursi \*

\* pada faktorial \*

\* ------------------------------------ \*/

#include <stdio.h>

#include <conio.h>

int faktorial (int);

main()

{

int x;

puts("MENCARI FAKTORIAL DARI X!");

printf("Masukkan nilai x (bulat positif) : ");

scanf("%d", &x);

printf("Faktorial dari %d = %d\n", x, faktorial (x));

getch();

}

int faktorial (int m) /\*definisi fungsi\*/

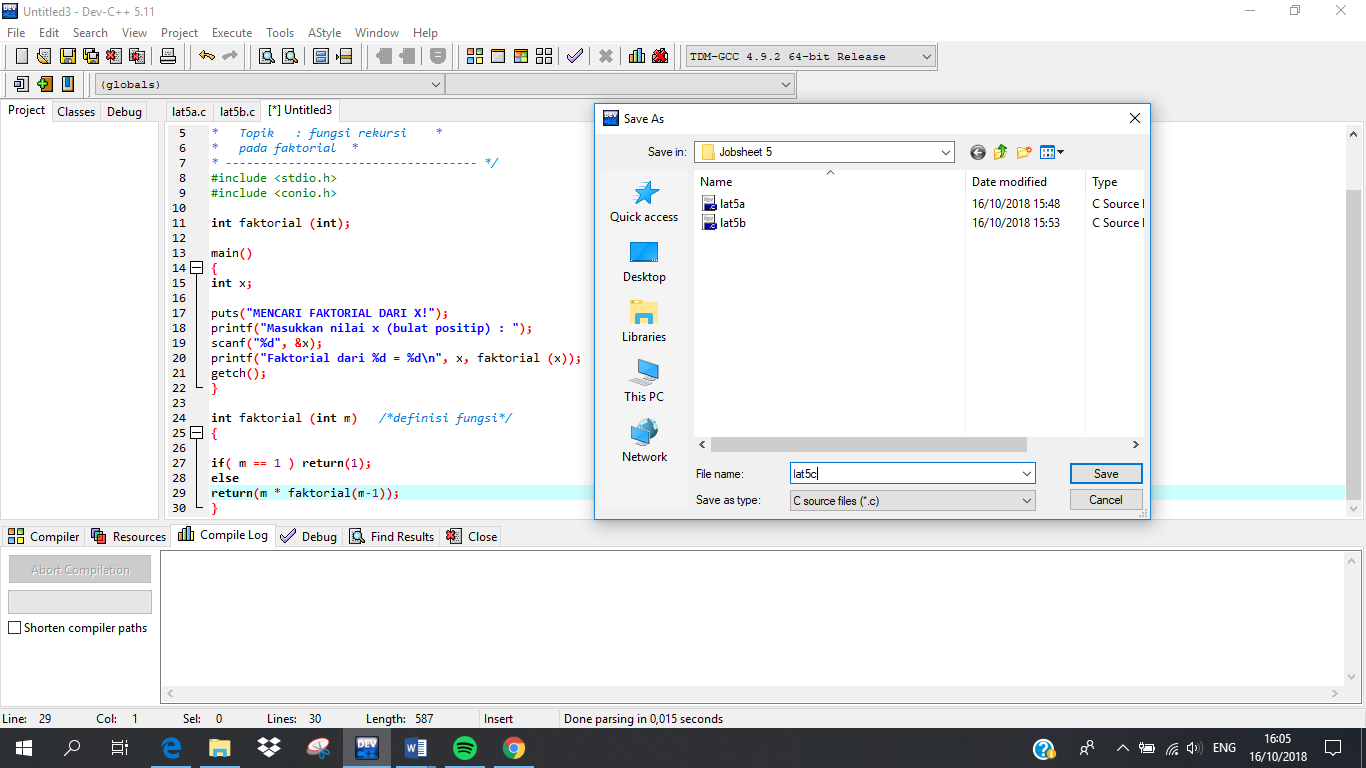
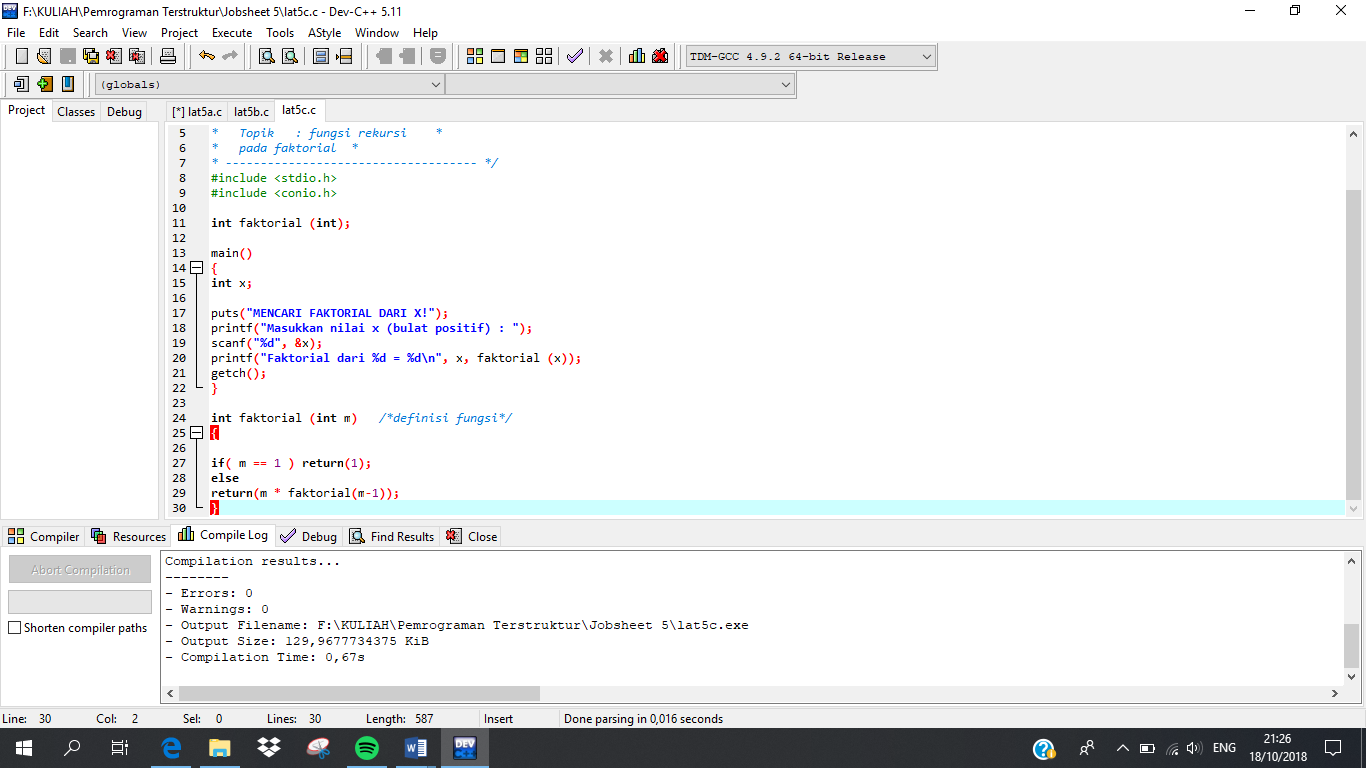
{

if( m == 1 ) return(1);

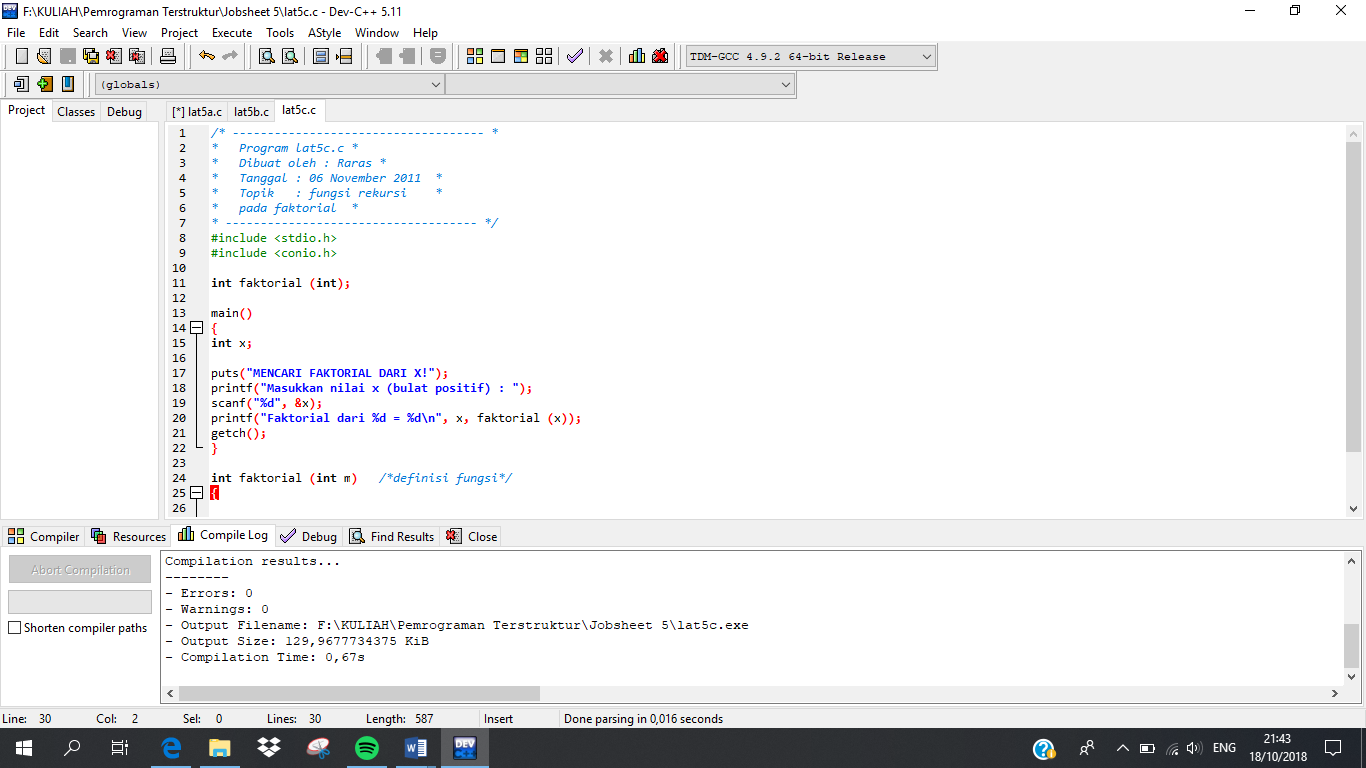
else

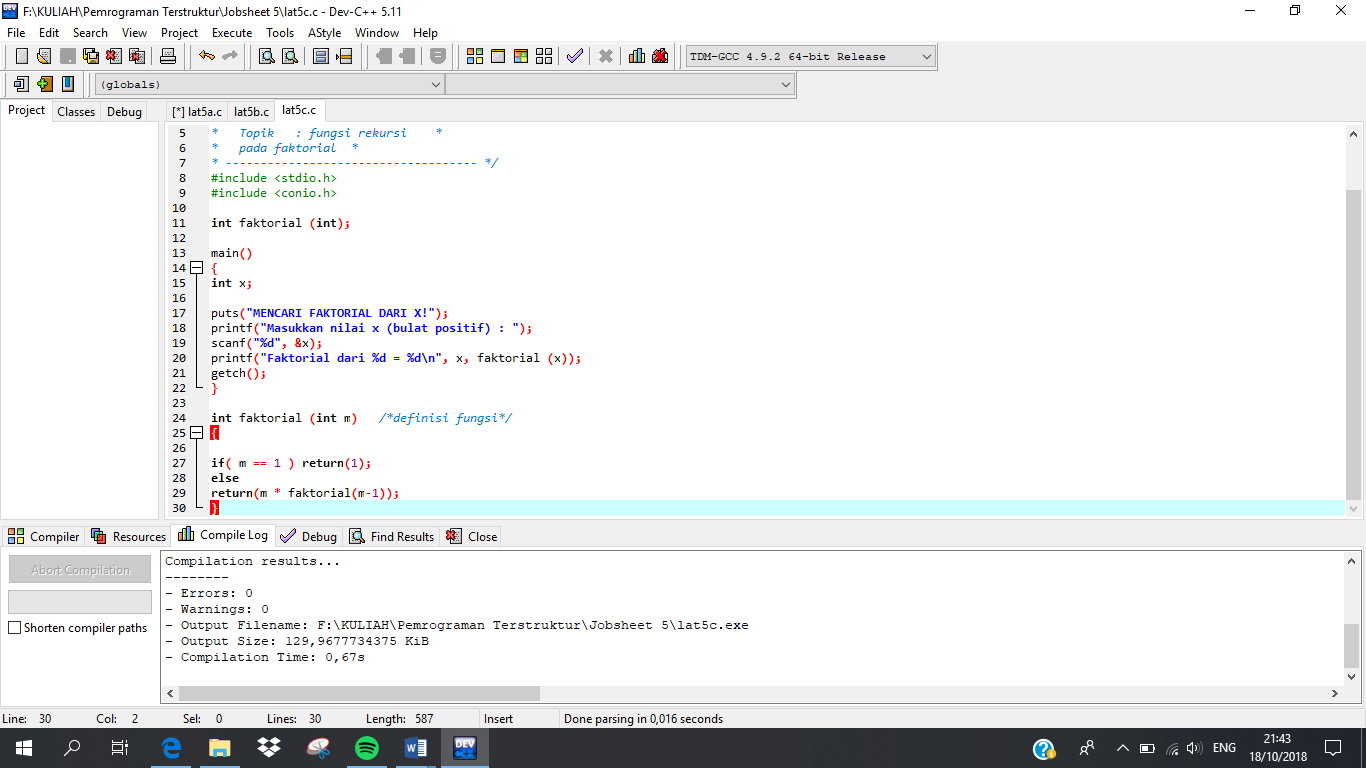
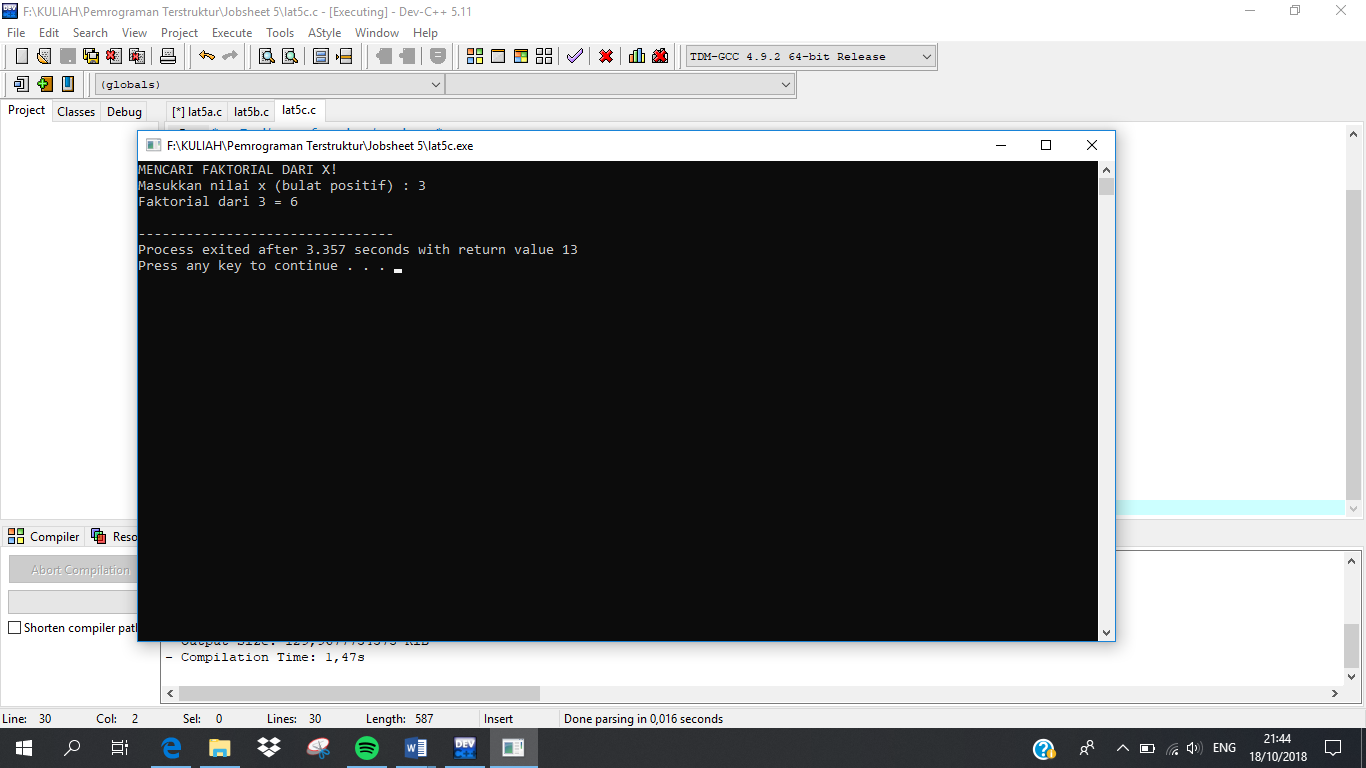
return(m \* faktorial(m-1));

}

* 1. Alt+F, save dengan nama Lat5c.c
  2. Lalu Compile hingga tidak terdapat error dan dapat di run.

Setelah di compile tidak terdapat error

* 1. Run program tersebut.

Hasil Program

### PERTANYAAN

* 1. Apa yang dimaksud dengan call by value dan call by reference?
* Call by Value

Adalah pemanggilan dengan nilai, nilai dari parameter aktual akan disalin ke parameter formal. Dengan cara ini nilai parameter aktual tidak bisa berubah sekalipun nilai parameter formal berubah.

* Call by Reference

Merupakan upaya untuk melewatkan alamat dari suatu variabel ke dalam fungsi. Cara ini dapat dipakai untuk mengubah isi suatu variabe di luar fungsi pelaksanaan pengubahan dilakukan di dalam fungsi.

* 1. Jelaskan tentang penggolongan variabel berdasarkan kelas penyimpanan!
* Variabel lokal

Variabel lokal adalah variabel yang dideklarasikan dalam fungsi.

Sifat Variabel Lokal :

a) Secara  otomatis  diciptakan  ketika  fungsi  dipanggil dan  akan  sirna  (lenyap)  ketika eksekusi terhadap fungsi berakhir.

b) Hanya dikenal oleh fungsi tempat variabel tersebut dideklarasikan

c) Tidak ada inisialisasi secara otomatis (saat variabel diciptakan, nilainya tak menentu).

* Variabel Eksternal

Variabel eksternal merupakan variabel yang dideklarasikan di luar fungsi.

Sifat Variabel Eksternal :

a) Dapat diakses oleh semua fungsi

b) Kalau tak diberi nilai, secara otomatis diinisialisasi dengan nilai sama dengan nol.

* Variabel Statis

Variabel  statis  dapat  berupa  variabel  internal  (didefinisikan  di  dalam  fungsi) maupun variabel eksternal.

Sifat variable statis :

a) Kalau variabel statis bersifat internal, maka variabel hanya dikenal oleh fungsi tempat variabel dideklarasikan

b) Kalau variabel statis bersifat eksternal, maka variabel dapat dipergunakan oleh semua fungsi yang terletak pada file yang sama, tempat variabel statis dideklarasikan

c)Berbeda  dengan  variabel  lokal,  variabel  statis  tidak  akan  hilang  sekeluarnya  dari fungsi (nilai pada variabel akan tetap diingat).

d) Inisialisasi akan dilakukan hanya sekali, yaitu saat fungsi dipanggil yang pertama kali.

* Variabel Register

Variabel register adalah variabel yang nilainya disimpan dalam register dan bukan dalam memori RAM.

Variabel yang seperti ini hanya bisa diterapkan pada variabel yang lokal atau parameter formal, yang bertipe char atau int.

Variabel register biasa diterapkan pada  variabel  yang  digunakan  sebagai  pengendali loop.

Tujuannya  untuk  mempercepat proses  dalam loop.    Sebab  variabel  yang  dioperasikan  pada  register  memiliki  kecepatan yang jauh lebih tinggi daripada variabel yang diletakkan pada RAM.

* 1. Buat program menggunakan fungsi untuk mnyelesaikan soal berikut dan simpan dalam file lat5d.c :

Menampilkan menu untuk menghitung :

1. Rumus Phytagoras
2. Rumus abc
3. Keluar

Gunakan struktur case dan masing-masing perhitungan dibuat fungsinya sehingga bisa dipanggil dengan kode program di dalam struktur case!

* + - 1. Ketik :

#include <stdio.h>

#include <conio.h>

#include <stdlib.h>

#include <math.h>

int phytagoras (int, int, int);

float abc (float, float, float);

main()

{

while(1)

{

int a, b, c;

int kode;

puts("Silahkan pilih menu dibawah :");

puts("1. Rumus Phytagoras");

puts("2. Rumus abc");

puts("3. Selesai");

printf("Masukkan pilihan anda (1-3) : ");

scanf("%d", &kode);

if(kode < 1 || kode > 3)

{

puts("Pilihan anda salah, silakan coba lagi\n\n");

continue;

}

switch(kode)

{

case 1 :

phytagoras (a, b, c);

break;

case 2 :

abc (a, b, c);

break;

case 3 :

{

exit(0);

getch();

}

}

}

getch();

}

int phytagoras (int a, int b, int c)

{

printf("Masukkan Nilai a : ");

scanf("%d", &a);

printf("Masukkan Nilai b : ");

scanf("%d", &b);

c = sqrt((a\*a) + (b\*b));

printf("Nilai c = %d\n\n", c);

}

float abc (float a, float b, float c)

{

float x, x1, x2, D;

printf ("Masukkan nilai a : ");

scanf("%f", &a);

printf ("Masukkan nilai b : ");

scanf("%f", &b);

printf ("Masukkan nilai c : ");

scanf("%f", &c);

D = ((b\*b) - 4\*a\*c);

if (D < 0)

printf("Akar imajiner\n\n");

if (D == 0)

{

x = (-b/2\*a);

printf("Akar :\tX1 = X2 = %.0f\n\n", x);

}

if(D > 0)

{

x1 = ((-b)+(sqrt(D)))/2\*a;

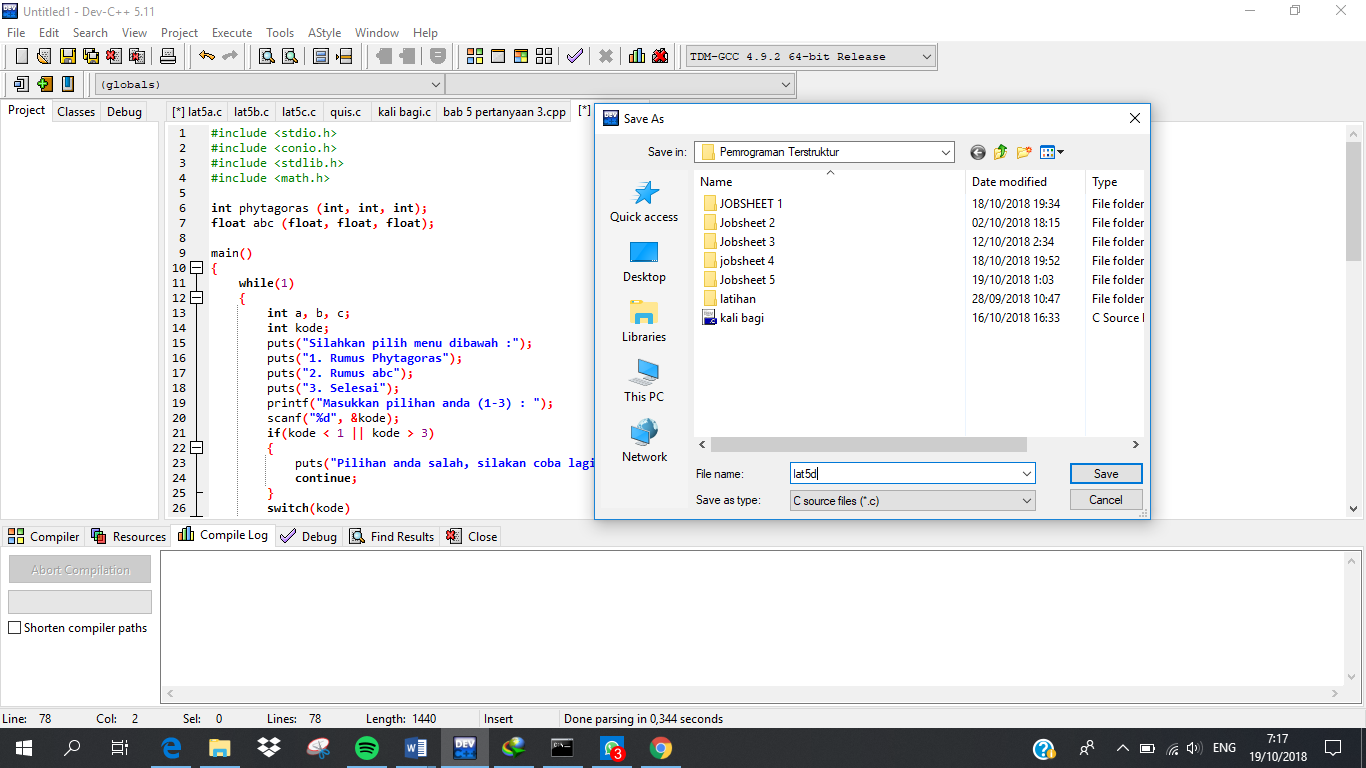
x2 =((-b)-(sqrt(D)))/2\*a;

printf("Akar:\tX1 = %.2f\n", x1);

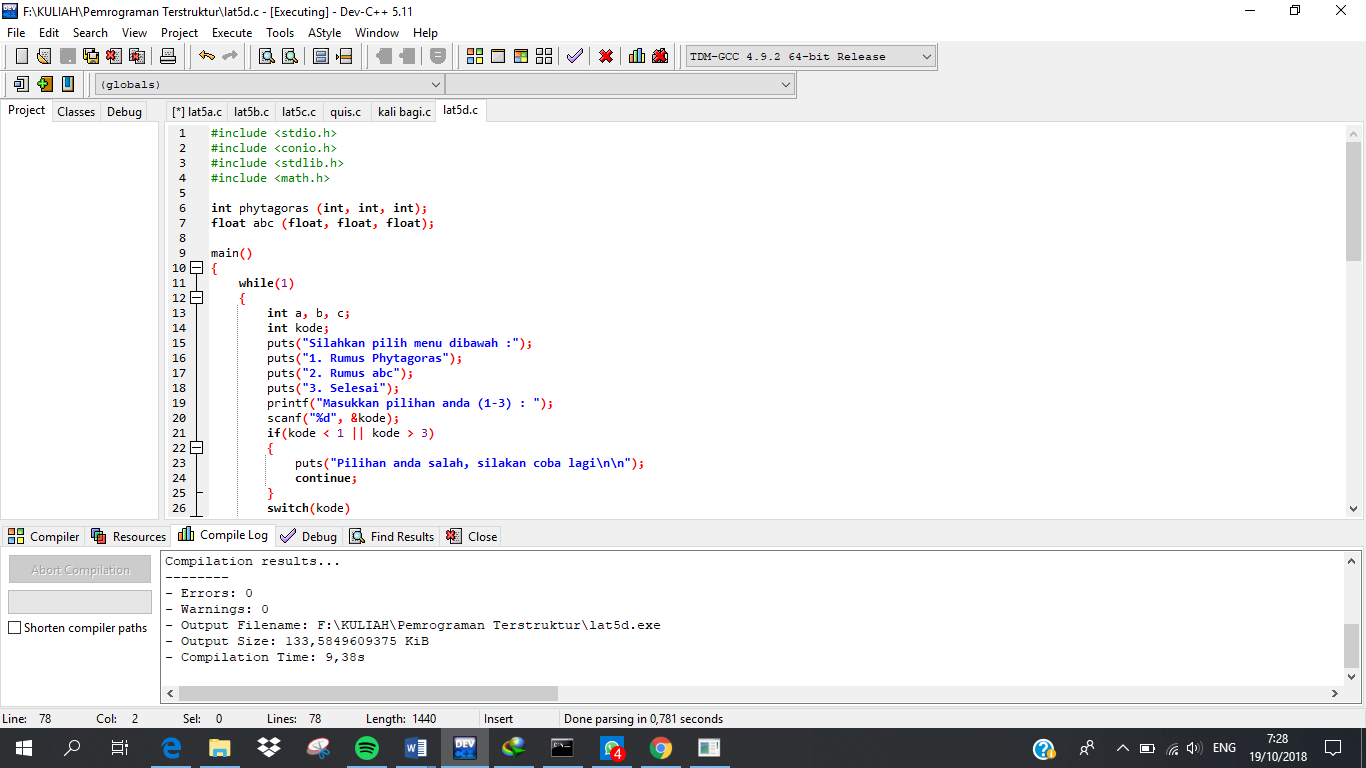
printf("\tX2 = %.2f\n\n", x2);

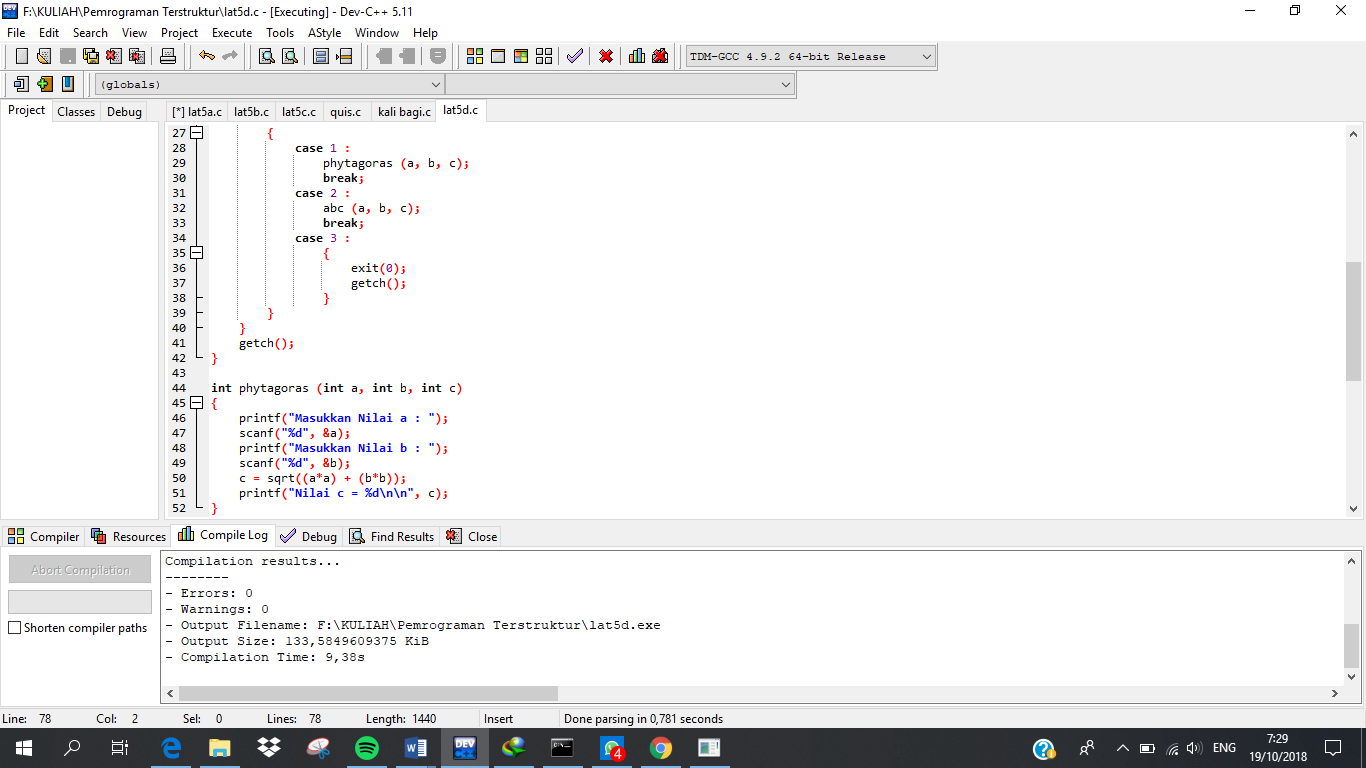
}

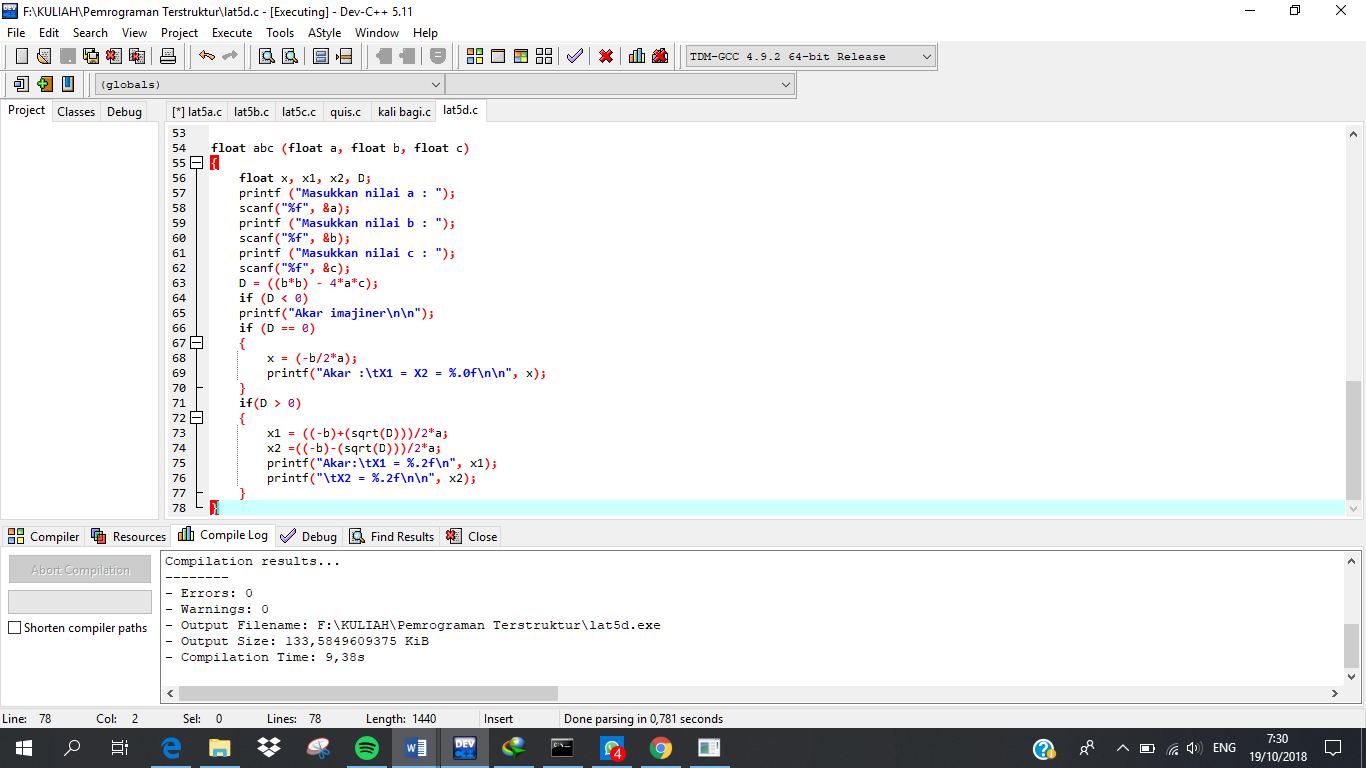
}

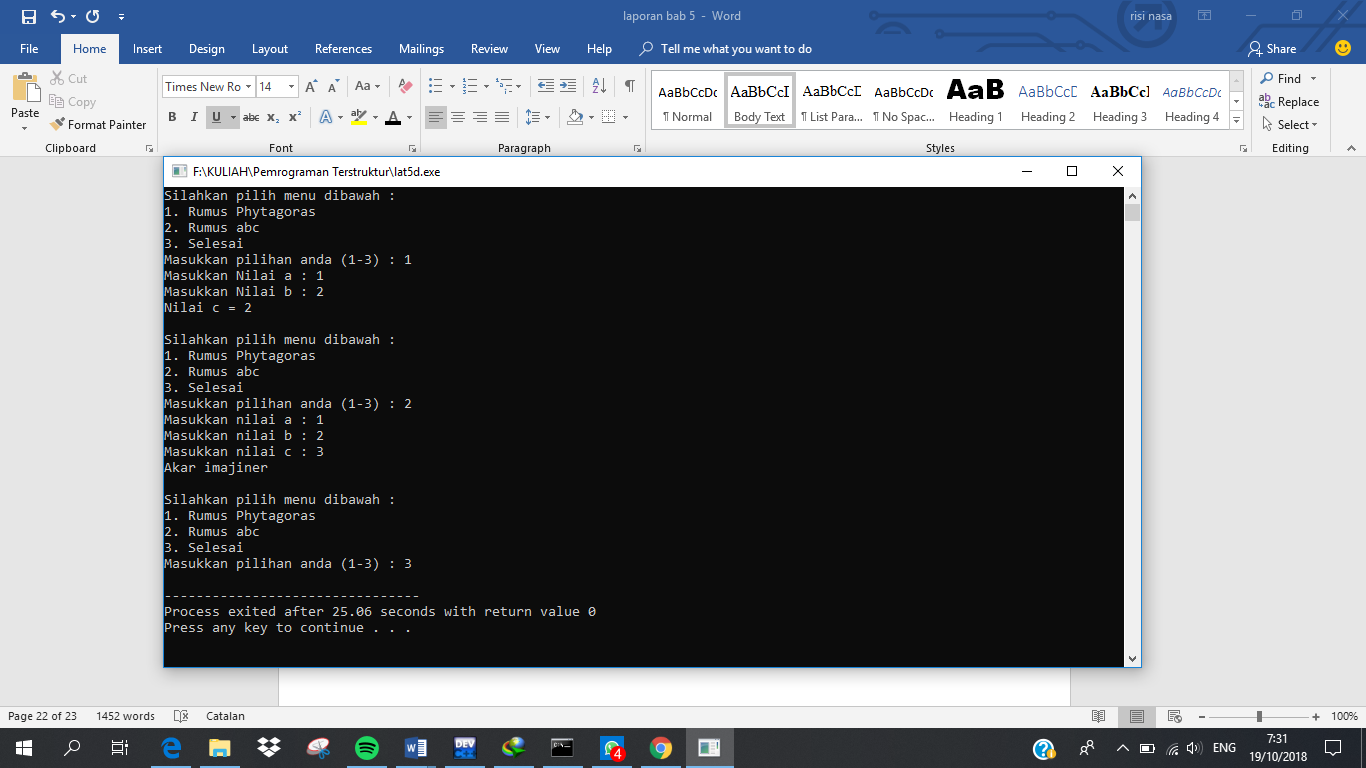


* + - 1. Save dengan nama lat5d.c
      2. Run program tersebut.







Hasil Program

### QUIZ

### Buatlah perhitungan aritmatika pertambahan, pengurangan, perkalian, pembagian dengan menggunakan fungsi dan gunakan struktur case.

### Ketik :

### #include <stdio.h>

### #include <conio.h>

### #include <stdlib.h>

### int jumlah(int, int);

### int kurang(int, int);

### int kali(int, int);

### float bagi(float, float);

### main()

### {

### int bil1, bil2, hasil;

### float bil\_a, bil\_b, hasil\_2;

### int i, kode;

### 

### while(1)

### {

### for(i=1;i<20;++i)

### printf("\xDB");

### printf("\n\nOPERASI DUA BILANGAN");

### printf("\n1.Penjumlahan");

### printf("\n2.Pengurangan");

### printf("\n3.Perkalian");

### printf("\n4.Pembagian");

### printf("\n5.Selesai");

### printf("\n\n");

### printf("Masukkan pilihan anda : ");

### scanf("%d", &kode);

### printf("\n");

### if(kode==5)

### exit(0);

### if(kode < 1 || kode > 5)

### {

### printf("Anda salah memilih");

### printf("\nTekan sembarang tombol");

### getch();

### continue;

### }

### if(kode==1)

### printf("Hasil penjumlahan = %d \n", jumlah(bil1, bil2));

### if(kode==2)

### printf("Hasil pengurangan = %d \n", kurang(bil1, bil2));

### if(kode==3)

### printf("Hasil perkalian = %d \n", kali(bil1, bil2));

### if(kode==4)

### printf("Hasil pembagian = %g \n", bagi(bil\_a, bil\_b));

### }

### }

### int jumlah(int bil1, int bil2)

### {

### //int hasil;

### printf("Masukkan bilangan pertama : ");

### scanf("%d", &bil1);

### printf("Masukkan bilangan kedua : ");

### scanf("%d", &bil2);

### return bil1+bil2;

### //printf("Hasil penjumlahan = %d", hasil);

### }

### int kurang(int bil1, int bil2)

### {

### printf("Masukkan bilangan pertama : ");

### scanf("%d", &bil1);

### printf("Masukkan bilangan kedua : ");

### scanf("%d", &bil2);

### return bil1-bil2;

### //printf("Hasil pengurangan = %d", hasil);

### }

### int kali(int bil1, int bil2)

### {

### printf("Masukkan bilangan pertama : ");

### scanf("%d", &bil1);

### printf("Masukkan bilangan kedua : ");

### scanf("%d", &bil2);

### return bil1\*bil2;

### //printf("Hasil perkalian = %d", hasil);

### }

### float bagi(float bil\_a, float bil\_b)

### {

### 

### printf("Masukkan bilangan pertama : ");

### scanf("%f", &bil\_a);

### printf("Masukkan bilangan kedua : ");

### scanf("%f", &bil\_b);

### return bil\_a/bil\_b;

### //printf("Hasil pembagian = %g", hasil\_2);

### }

### Run program tersebut

### 

### 

### 

### 

### Hasil Program

### 

### 

### 

### 

### 

### KESIMPULAN

Setelah saya melakukan praktikum bab 5 tentang fungsi saya menyimpulkan bahwa :

* 1. Ada 2 cara pemanggilan fungsi yaitu: Call by Value dan Call by reference
  2. Call by Value merupakan pemanggilan fungsi dengan melewatkan nilai parameter variabel dalam definisi fungsi.
  3. Call by reference merupakan suatu usaha melewatkan alamat dari suatu variabel ke dalam fungsi.
  4. Tipe fungsi berguna untuk menentukan tipe keluaran fungsi yang berupa salah satu tipe data c++ misalnya int atau char. Jika tipe data tidak disebutkan maka dianggap sebagai int.