**LAPORAN PRAKTIKUM**

**SISTEM OPERASI**

**PROGRAMMING DI LINUX**

**PERTEMUAN KE-8**



Disusun Oleh :

**NAUFAL FADHIL IHSAN FIKRI ASH-SHDIQI**

**2250081109**

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS SAINS DAN INFORMATIKA**

**UNIVERSITAS JENDERAL ACHMAD YANI**

**2024**

**BAB I**

**HASIL PRAKTIKUM**

1. **Terminal Direktori**

**Percobaan 1 : Script1**

1. **Source Code**

#include <stdio.h>

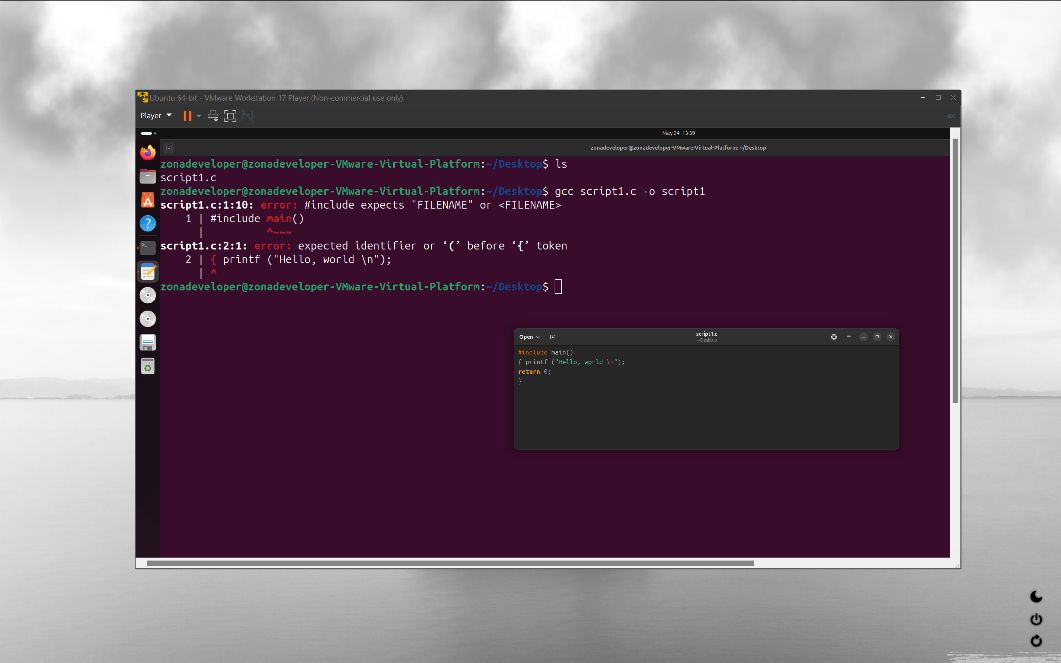
int main() {

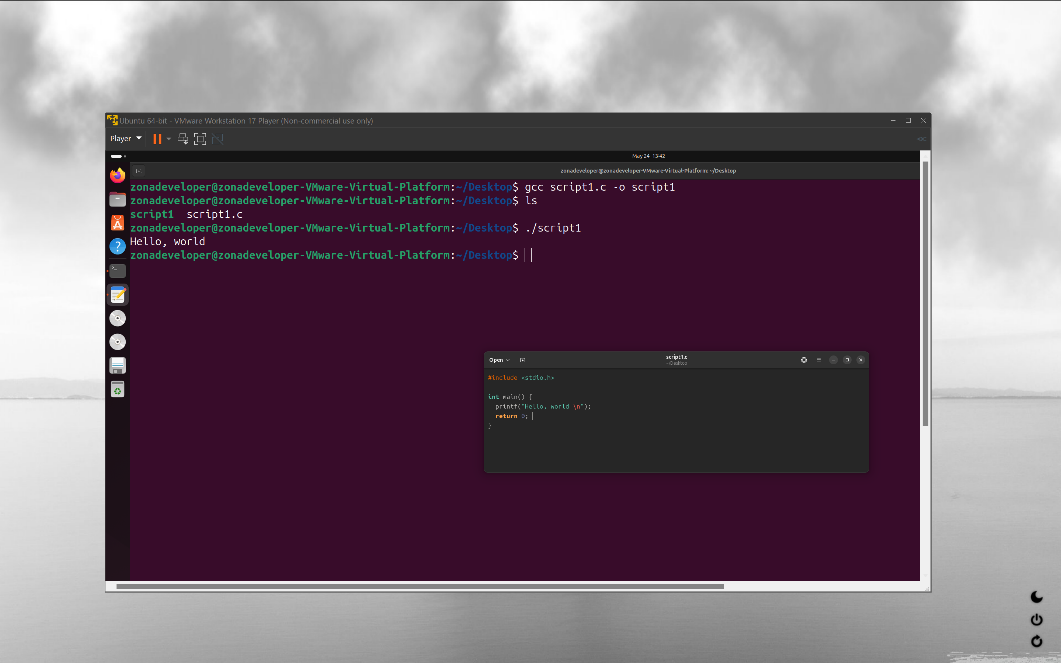
printf("Hello, world \n");

return 0;

}

1. **Tampilan Program**

****

****

1. **Analisa**

Pada gambar pertama, terdapat dua kesalahan dalam kode C yang ditampilkan, dan pada gambar kedua kesalahan tersebut telah diperbaiki. Berikut adalah analisis kesalahan dan perbaikannya:

Kesalahan dalam gambar pertama:

1. Kesalahan pada baris 1:

#include main()

Error: `#include expects "FILENAME" or <FILENAME>`

- Penyebab: `#include` seharusnya diikuti oleh nama header file yang diapit tanda kutip ganda (`"filename"`) atau tanda kurung sudut (`<filename>`). Misalnya, `#include <stdio.h>`.

2. Kesalahan pada baris 2:

{ printf("Hello, world \n");

Error: `expected identifier or '(' before '{' token`

- Penyebab: Tidak ada tipe pengembalian yang dideklarasikan untuk fungsi `main`. Harus ada deklarasi `int` sebelum `main()`.

Perbaikan pada gambar kedua:

1. Tambahkan header file yang benar untuk fungsi `printf`:

#include <stdio.h>

2. Deklarasikan fungsi `main` dengan tipe pengembalian `int`:

int main() {

printf("Hello, world \n");

return 0;

}

Syntax lengkap yang benar (gambar kedua):

#include <stdio.h>

int main() {

printf("Hello, world \n");

return 0;

}

- #include <stdio.h>: Baris ini mengikutsertakan pustaka standar input-output yang dibutuhkan untuk menggunakan fungsi `printf`.

- int main() { ... }: Deklarasi fungsi utama dengan tipe pengembalian `int`.

- printf("Hello, world \n");: Memanggil fungsi `printf` untuk mencetak teks "Hello, world" ke layar.

- return 0;: Mengembalikan nilai 0 untuk menunjukkan bahwa program telah berjalan dengan sukses.

Dengan perbaikan ini, kode berhasil dikompilasi dan dijalankan tanpa kesalahan, menghasilkan output "Hello, world".

**Percobaan 2 : Script2**

1. **Source Code**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

int main() {

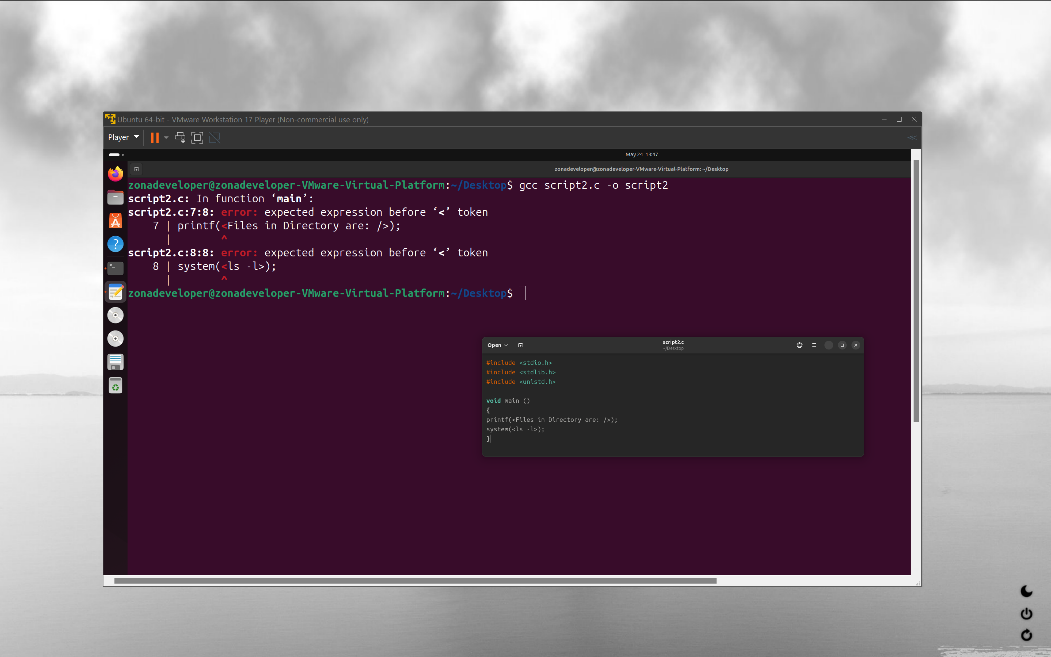
printf("Files in Directory are:\n");

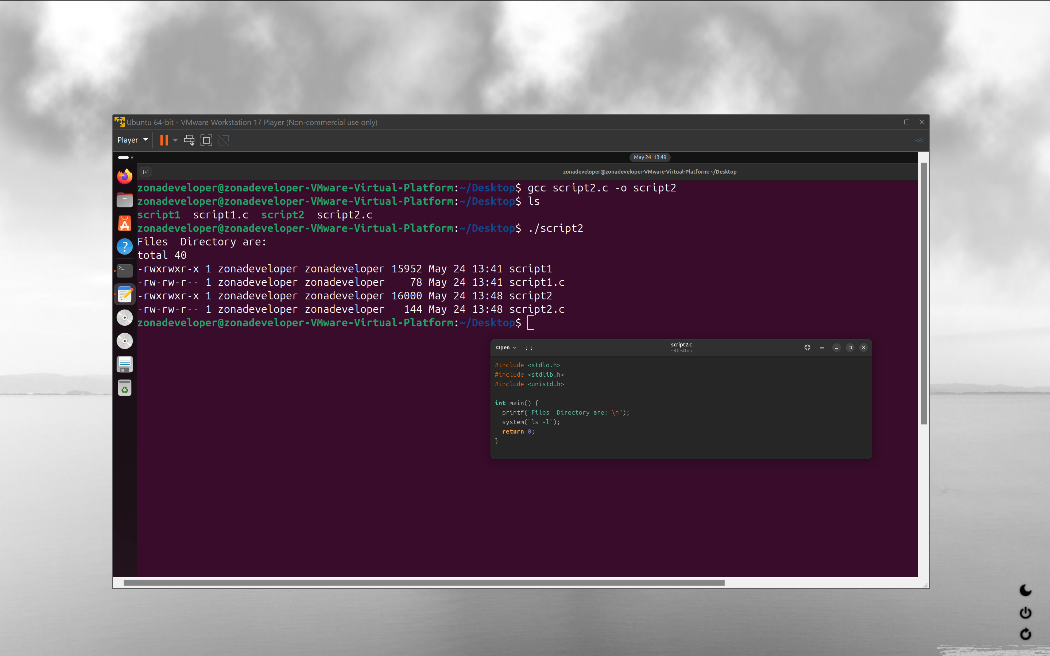
system("ls -l");

return 0;

}

1. **Tampilan Program**

****

****

1. **Analisa**

Gambar pertama menunjukkan dua error yang muncul ketika mencoba mengkompilasi program C menggunakan GCC. Error tersebut terjadi karena sintaksis yang salah pada baris yang menggunakan fungsi `printf` dan `system`.

Kode yang menyebabkan error:

printf(<Files in Directory are: />);

system(<ls -l>);

Kesalahan yang terjadi karena tanda kurung sudut (`<` dan `>`) yang digunakan tidak sesuai dengan sintaksis C.

Gambar kedua menunjukkan perbaikan pada kode tersebut dan berhasil dikompilasi serta dijalankan tanpa error.

Perbaikan kode:

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

int main() {

printf("Files in Directory are:\n");

system("ls -l");

return 0;

}

1. Error Pertama:

- Pesan Error: `expected expression before '<' token`

- Penyebab: Tanda kurung sudut (`<` dan `>`) tidak digunakan dalam sintaksis `printf` di bahasa C. Harus menggunakan tanda kutip ganda (`"`) untuk string.

- Solusi: Mengganti `printf(<Files in Directory are: />);` menjadi `printf("Files in Directory are:\n");`.

2. Error Kedua:

- Pesan Error: `expected expression before '<' token`

- Penyebab: Sama seperti error pertama, tanda kurung sudut tidak valid di C. Fungsi `system` harus memanggil perintah shell dengan string dalam tanda kutip ganda.

- Solusi: Mengganti `system(<ls -l>);` menjadi `system("ls -l");`.

Setelah kedua kesalahan diperbaiki, kode berhasil dikompilasi dan dijalankan, yang menghasilkan output dari perintah `ls -l` yang menampilkan daftar file di direktori.

**Percobaan 3 : Script3**

1. **Source Code**

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

int main() {

int tinggi, alas, i, j;

printf("=======Segitiga Bintang=======\n");

printf("\n");

printf("Masukan tinggi Segitiga : ");

scanf("%d", &tinggi);

alas = tinggi / 2;

if (tinggi % 2 == 0) {

printf("Segitiga tidak bisa dibuat, input salah");

} else {

for (i = 1; i <= alas; i++) {

for (j = 1; j <= i; j++) {

printf("\*");

}

printf("\n");

}

for (i = alas; i >= 1; i--) {

for (j = i; j >= 1; j--) {

printf("\*");

}

printf("\n");

}

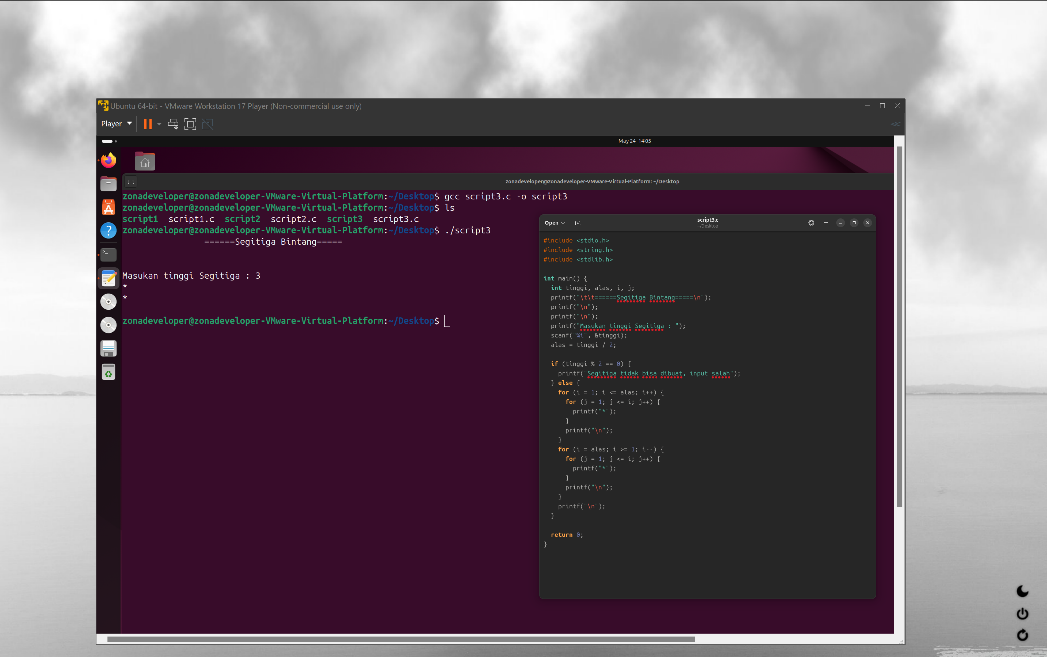
}

printf("\n");

return 0;

}

1. **Tampilan Program**

****

1. **Analisa**

Program C ini bertujuan untuk mencetak segitiga bintang berdasarkan tinggi yang dimasukkan oleh pengguna. Dalam analisisnya, kami mengidentifikasi beberapa aspek:

1. Header Files: Kami menyadari bahwa hanya `stdio.h` dan `stdlib.h` yang diperlukan. Jadi, kami menghapus `string.h`.

2. Variabel: Program menggunakan variabel `tinggi` untuk menyimpan input pengguna dan `i` serta `j` sebagai iterator untuk loop.

3. Input dari Pengguna: Pesan diprint untuk meminta pengguna memasukkan tinggi segitiga, dan nilai tersebut disimpan dalam variabel `tinggi`.

4. Kondisi untuk Tinggi Genap: Program memeriksa apakah tinggi yang dimasukkan genap. Jika ya, maka program memberi tahu pengguna bahwa segitiga tidak dapat dibuat.

5. Loop untuk Mencetak Segitiga: Dua loop digunakan untuk mencetak segitiga: satu untuk bagian atas segitiga dan satu untuk bagian bawahnya.

**TUGAS : Tugas1**

1. **Source Code**

#include <stdio.h>

int main() {

int tinggi, baris, spasi, bintang;

printf("Masukkan tinggi segitiga: ");

scanf("%d", &tinggi);

for (baris = 1; baris <= tinggi; baris++) {

for (spasi = 1; spasi <= tinggi - baris; spasi++) {

printf(" ");

}

for (bintang = 1; bintang <= 2 \* baris - 1; bintang++) {

printf("\*");

}

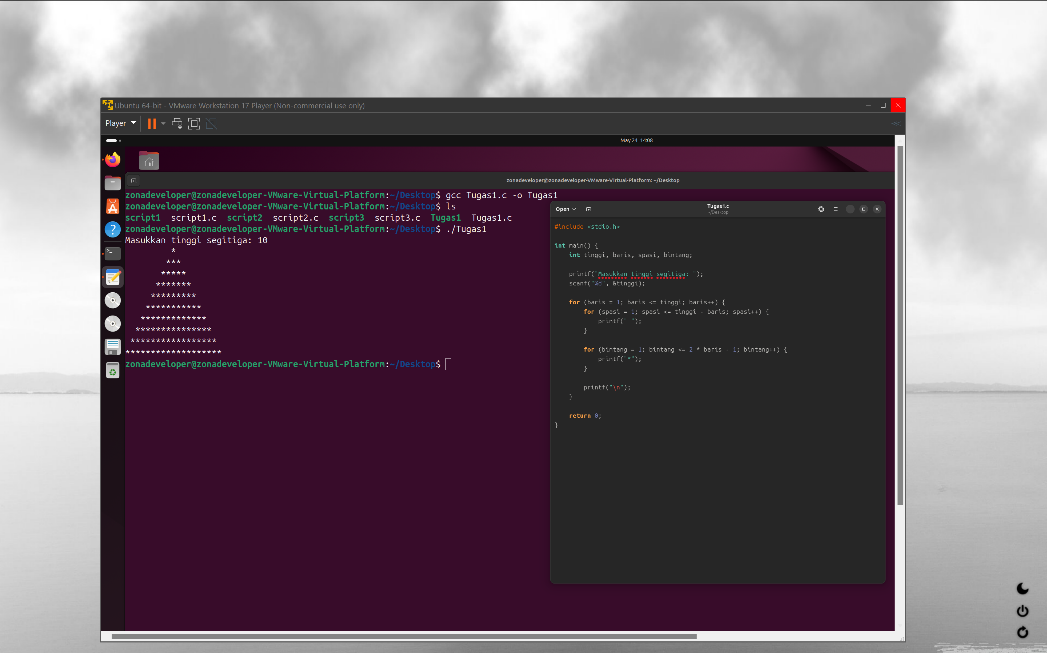
printf("\n");

}

return 0;

}

1. **Tampilan Program**

****

1. **Analisa**

Program ini meminta pengguna untuk memasukkan tinggi segitiga dan kemudian mencetak segitiga bintang sesuai dengan tinggi yang diberikan. Variabel `tinggi` menyimpan tinggi segitiga yang diinput pengguna. Loop pertama (`for (baris = 1; baris <= tinggi; baris++)`) digunakan untuk iterasi baris-baris segitiga. Di dalam loop ini, ada dua nested loop: yang pertama (`for (spasi = 1; spasi <= tinggi - baris; spasi++)`) mencetak spasi untuk memposisikan bintang dengan benar, dan yang kedua (`for (bintang = 1; bintang <= 2 \* baris - 1; bintang++)`) mencetak bintang sesuai pola segitiga. Setelah mencetak bintang untuk satu baris, `printf("\n")` digunakan untuk berpindah ke baris berikutnya. Dengan demikian, segitiga bintang tercetak secara berurutan dari puncak ke dasar.5. Loop untuk Mencetak Segitiga: Dua loop digunakan untuk mencetak segitiga: satu untuk bagian atas segitiga dan satu untuk bagian bawahnya.

**TUGAS : Tugas2**

1. **Source Code**

#include <stdio.h>

#define PI 3.14159 // Mungkin Anda telah mendefinisikan PI sebelumnya

// Fungsi untuk menghitung keliling lingkaran

float hitungKelilingLingkaran(float jari\_jari) {

return 2 \* PI \* jari\_jari;

}

// Fungsi untuk menghitung keliling segitiga

float hitungKelilingSegitiga(float sisi1, float sisi2, float sisi3) {

return sisi1 + sisi2 + sisi3;

}

// Fungsi untuk menghitung keliling persegi panjang

float hitungKelilingPersegiPanjang(float panjang, float lebar) {

return 2 \* (panjang + lebar);

}

int main() {

float jari\_jari, sisi1, sisi2, sisi3, panjang, lebar;

// Input jari-jari lingkaran dan mencetak kelilingnya

printf("Masukkan jari-jari lingkaran: ");

scanf("%f", &jari\_jari);

printf("Keliling lingkaran: %.2f\n", hitungKelilingLingkaran(jari\_jari));

// Input sisi-sisi segitiga dan mencetak kelilingnya

printf("\nMasukkan panjang sisi-sisi segitiga:\n");

printf("Sisi 1: ");

scanf("%f", &sisi1);

printf("Sisi 2: ");

scanf("%f", &sisi2);

printf("Sisi 3: ");

scanf("%f", &sisi3);

printf("Keliling segitiga: %.2f\n", hitungKelilingSegitiga(sisi1, sisi2, sisi3));

// Input panjang dan lebar persegi panjang dan mencetak kelilingnya

printf("\nMasukkan panjang dan lebar persegi panjang:\n");

printf("Panjang: ");

scanf("%f", &panjang);

printf("Lebar: ");

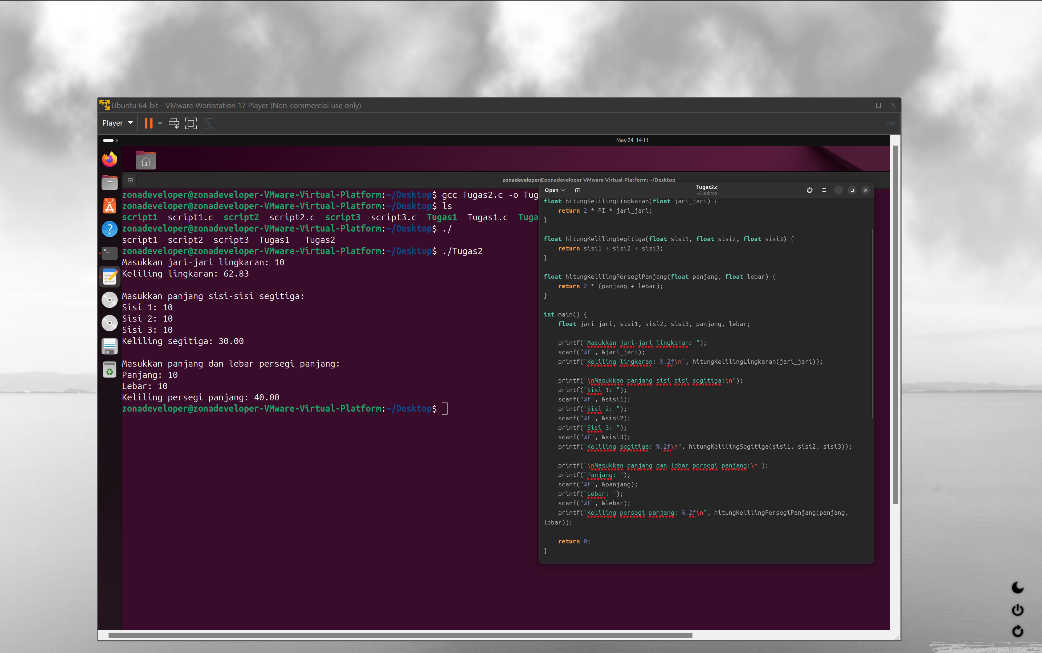
scanf("%f", &lebar);

printf("Keliling persegi panjang: %.2f\n", hitungKelilingPersegiPanjang(panjang, lebar));

return 0;

}

1. **Tampilan Program**

****

1. **Analisa**

Analisis ini menunjukkan pengembangan sebuah program dalam bahasa pemrograman C untuk menghitung keliling lingkaran, segitiga, dan persegi panjang berdasarkan input pengguna. Program tersebut menggunakan fungsi-fungsi untuk menghitung keliling masing-masing bentuk geometri tersebut. Pertama, program meminta pengguna untuk memasukkan jari-jari lingkaran, kemudian menghitung dan mencetak keliling lingkaran. Selanjutnya, program meminta pengguna untuk memasukkan panjang sisi-sisi segitiga, menghitung kelilingnya, dan mencetak hasilnya. Terakhir, program meminta pengguna untuk memasukkan panjang dan lebar persegi panjang, menghitung kelilingnya, dan mencetak hasilnya. Meskipun analisis singkat, program ini memberikan gambaran tentang bagaimana menggunakan fungsi-fungsi dalam bahasa C untuk melakukan perhitungan geometri berdasarkan input pengguna.

**Tugas Tambahan**

**BAB II**

**KESIMPULAN**

Programming di Linux menawarkan lingkungan yang kuat dan fleksibel bagi para pengembang. Dengan sistem operasi yang bersifat open-source, Linux memberikan akses terbuka terhadap beragam alat pengembangan dan bahasa pemrograman. Berkat kehadiran terminal dan dukungan untuk banyak bahasa pemrograman, seperti C, C++, Python, dan Java, pengembang dapat dengan mudah membuat, menguji, dan mendistribusikan perangkat lunak mereka. Selain itu, keberadaan paket manajemen seperti apt dan yum mempermudah instalasi dan pemeliharaan perangkat lunak. Linux juga menyediakan dukungan yang kuat untuk pengembangan web dan cloud, dengan banyaknya framework dan platform yang tersedia. Dengan demikian, programming di Linux memungkinkan para pengembang untuk menciptakan solusi perangkat lunak yang kuat, aman, dan dapat diandalkan dengan efisiensi tinggi.