LAPORAN POINTER

Oleh:

Nama	: L HAFIDL ALKHAIR
NIM	: 2023903430060
Kelas	: TRKJ 1.C
Jurusan	: Teknologi Informasi dan Komputer
Program Studi	: Teknologi Rekayasa Komputer Jaringan
Dosen Pembimbing	: Indrawati, SST. MT

JURUSAN TEKNOLOGI, KOMPUTER, DAN INFORMASI

PRODI TEKNOLOGI REKAYASA KOMPUTER DAN JARINGAN

POLITEKNIK NEGERI LHOKSEUMAWE

TAHUN AJARAN 2023/2024

LEMBAR PENGESAHAN

No. Praktikum	: 04 /TIK/TRKJ-1C/ Data Structure And Algorithms Practice
Judul	: Laporan Pointer
Nama	: L HAFIDL ALKHAIR
NIM	: 2023903430060
Kelas	: TRKJ-1C
Jurusan	: Teknologi Informasi Dan Komputer
Prodi	: Teknologi Rekayasa Komputer Jaringan
Tanggal Praktikum	: 4 Oktober 2023
Tanggal Penyerahan	: 9 Oktober 2023

Buketrata, 9 Oktober 2023

Dosen Pembimbing,

Indrawati, SST.MT

Nip. 19740815 200112 2 001

A. Tujuan

- Pengelolaan Memori yang Efisien
- Manipulasi Data yang Fleksibel
- Pemahaman yang Mendalam tentang Array dan String
- Penggunaan Fungsi yang Lebih Fleksibel
- Implementasi Struktur Data Dinamis

B. Dasar teori

Pointer adalah salah satu konsep dasar dalam pemrograman yang memungkinkan pengaksesan langsung ke alamat memori dari variabel atau objek lain dalam komputer. Dengan kata lain, pointer adalah variabel yang menyimpan alamat memori lokasi variabel lain.

Berikut adalah beberapa konsep dasar yang perlu dipahami tentang pointer:

1. Alamat Memori:

   Setiap variabel dalam program disimpan di lokasi memori tertentu dalam komputer. Alamat memori adalah lokasi unik di mana variabel disimpan.
2. Deklarasi Pointer:

   Pointer dideklarasikan dengan menambahkan tanda asterisk (*) sebelum nama variabel. Misalnya, int *ptr; mendeklarasikan pointer untuk variabel bertipe integer.
3. Operasi Pointer:

Referensi (&): Operator referensi digunakan untuk mendapatkan alamat memori dari variabel. Contoh: int angka = 10; int *ptr = &angka; menyimpan alamat angka dalam ptr.

Dereferensi (*): Operator dereferensi digunakan untuk mengakses nilai yang disimpan pada alamat memori yang ditunjuk oleh pointer. Contoh: int nilai = *ptr; mengambil nilai dari alamat yang ditunjuk oleh ptr.

4. Alokasi Memori Dinamis:

   Pointer memungkinkan alokasi memori dinamis menggunakan fungsi seperti malloc(), calloc(), dan realloc(). Memori yang dialokasikan dapat diakses dan dikelola melalui pointer.

5. Penggunaan Pointer dalam Fungsi:

   Pointer digunakan dalam fungsi sebagai parameter untuk mengubah nilai variabel di luar fungsi tersebut. Dengan ini, fungsi dapat mengoperasikan variabel asli melalui referensi pointer.

6. Pointer dan Array:

   Array dapat diakses melalui pointer. Pointer ke elemen pertama array bertipe T dapat dideklarasikan sebagai T *ptr = array;. Dengan menggunakan pointer, elemen-elemen array dapat diakses secara berurutan.

7. Pointer ke Pointer (Pointer to Pointer):

   Pointer juga dapat menunjuk ke pointer lain. Pointer ke pointer dideklarasikan dengan menggunakan dua tanda asterisk (**). Misalnya, int **ptr_ptr; adalah pointer yang menunjuk ke pointer bertipe integer.

8. Array Multidimensi dan Pointer:

   Dalam array multidimensi, elemen-elemen dapat diakses melalui pointer. Contoh: int matrix[3][3]; int (*ptr)[3] = matrix; menginisialisasi pointer ptr yang menunjuk ke array dua dimensi.

9. String dan Pointer:

Dalam C, string adalah array karakter yang diakhiri dengan karakter null ('\0'). String dapat diakses menggunakan pointer ke karakter (char *). Pointer ini memungkinkan manipulasi dan pengaksesan karakter dalam string.

Pemahaman yang baik tentang pointer penting karena pengelolaan memori yang tepat, efisiensi akses data, dan kemampuan untuk bekerja dengan struktur data yang kompleks semuanya bergantung pada konsep pointer. Pointer adalah dasar untuk pemahaman yang mendalam tentang banyak konsep dalam ilmu komputer, termasuk struktur data, alokasi memori dinamis, dan fungsi pemrograman tingkat rendah.

C. Alat dan bahan

Di dalam pembuatan sebuah Bahasa pemrograman tentu saja memerlukan alat dan bahan-bahan supaya program tersebut dapat di jalankan dengan baik dan tidak mengalami yang namanya error saat program atau kode tersebut di jalankan. Adapun alat dan bahan sebagai berikut:

a. Laptop atau device

b. Tools atau aplikasi pemrograman (disini kita memakai aplikasi Dev-C++)

c. Reverensi ataupun ide dalam membuat sebuah program

D. Program Pointer

1. Contoh Pointer Untuk Memanipulasi nilai variable

```c
#include <stdio.h>

int main()
{
    int angka = 10;
    int *pointerAngka; // Deklarasi pointer

    pointerAngka = &angka;
 // Inisialisasi pointer dengan alamat variabel ang
ka

    printf("Nilai variabel angka: %d\n", angka);
    printf("Alamat variabel angka: %p\n", &angka);
    printf("Nilai yang disimpan dalam pointer: %p\n
", pointerAngka);
    printf(
"Nilai variabel yang diakses melalui pointer: %d\n"
, *pointerAngka);
 // Mengakses nilai melalui pointer

    // Mengubah nilai variabel melalui pointer
    *pointerAngka = 20;
    printf(
"Nilai variabel angka setelah diubah melalui pointe
r:
%d\n", angka);

    return 0;
}
```

Hasil Program

```
PS D:\C\output> cd 'd:\C\output'
PS D:\C\output> & .\'c di vscode.exe'
Nilai variabel angka: 10
Alamat variabel angka: 0061FF18
Nilai yang disimpan dalam pointer: 0061FF18
Nilai variabel yang diakses melalui pointer: 10
Nilai variabel angka setelah diubah melalui pointer: 20
```

Analisa ;

- int *pointerAngka; - Mendeklarasikan pointer yang akan menunjuk ke variabel bertipe int.
- pointerAngka = &angka; - Menginisialisasi pointer dengan alamat variabel angka.
- printf("Nilai variabel yang diakses melalui pointer: %d\n", *pointerAngka); - Menggunakan operator dereference (*) untuk mengakses nilai variabel yang ditunjuk oleh pointer.
- *pointerAngka = 20; - Mengubah nilai variabel angka melalui pointer. Dalam hal ini, nilai variabel angka berubah menjadi 20 karena pointer menunjuk ke alamat variabel tersebut.

Semoga penjelasan ini membantu! Jangan ragu untuk bertanya jika Anda memiliki pertanyaan lainnya.

2. Pointer untuk bekerja dengan array:

```c
#include <stdio.h>

int main()
{
    int angka[5] = {1, 2, 3, 4, 5};
    int *pointerAngka = angka;
 // Inisialisasi pointer dengan alamat awal array

    printf("Isi array: ");
    for (int i = 0; i < 5; ++i)
    {
        printf("%d ", *(pointerAngka + i));
 // Mengakses elemen array melalui pointer
    }

    // Mengubah nilai elemen array melalui pointer
    *(pointerAngka + 2) = 10;
 // Mengubah nilai elemen ketiga menjadi 10

    printf("\nIsi array setelah perubahan: ");
    for (int i = 0; i < 5; ++i)
    {
        printf("%d ", *(pointerAngka + i));
    }

    return 0;
}
```

Hasil program :

```
PS D:\C\output> & .\'c di vscode.exe'
Isi array: 1 2 3 4 5
Isi array setelah perubahan: 1 2 10 4 5
PS D:\C\output>
```

Analisa :

- int angka[5] = {1, 2, 3, 4, 5}; - Mendeklarasikan dan menginisialisasi array angka dengan 5 elemen.
- int *pointerAngka = angka; - Menginisialisasi pointer dengan alamat awal array angka.
- Dalam loop pertama, program menggunakan pointer untuk mengakses dan mencetak nilai elemen-elemen array.
- *(pointerAngka + 2) = 10; - Mengubah nilai elemen ketiga (indeks 2) dari array angka menjadi 10 menggunakan pointer.
- Dalam loop kedua, program mencetak nilai elemen-elemen array setelah perubahan dilakukan.

Harap dicatat bahwa dalam pemrograman C, indeks array dimulai dari 0. Oleh karena itu, elemen ketiga memiliki indeks 2.

Penggunaan Pointer untuk String:

```c
#include <stdio.h>

int main()
{
    char *string = "Hello, World!";
 // Mendeklarasikan string menggunakan pointer
    char *ptr = string;
             // Menggunakan pointer untuk menunjuk
ke string

    // Menggunakan pointer untuk mencetak string
    printf("String menggunakan pointer: %s\n", ptr
);


// Mengakses karakter individual menggunakan pointe
r
    printf("Karakter pertama: %c\n", *ptr);
    printf("Karakter kedua: %c\n", *(ptr + 1));

    // Menggunakan pointer untuk mengubah string
    ptr = "Welcome to C Programming!";
    printf("String setelah diubah: %s\n", ptr);

    return 0;
}
```

Hasil Program :

```
PS D:\C\output> & .\'c di vscode.exe'
Nilai yang disimpan dalam pointer: 10
```

Analisa

- char *string = "Hello, World!";: Mendeklarasikan string menggunakan pointer. Variabel string menunjuk ke alamat pertama dari string "Hello, World!".
- char *ptr = string;: Menggunakan pointer ptr untuk menunjuk ke string yang sama dengan variabel string.
- printf("String menggunakan pointer: %s\n", ptr);: Menggunakan pointer untuk mencetak string. %s digunakan untuk mencetak string.
- printf("Karakter pertama: %c\n", *ptr);: Menggunakan operator dereference (*) untuk mengakses karakter pertama dari string menggunakan pointer.
- printf("Karakter kedua: %c\n", *(ptr + 1));: Menggunakan pointer untuk mengakses karakter kedua dari string menggunakan aritmatika pointer (*(ptr + 1)).
- ptr = "Welcome to C Programming!";: Mengubah pointer untuk menunjuk ke string lain ("Welcome to C Programming!").
- printf("String setelah diubah: %s\n", ptr);: Mencetak string setelah diubah menggunakan pointer.

Penting untuk diingat bahwa ketika menggunakan pointer untuk string, pastikan bahwa string yang ditunjuk oleh pointer adalah string konstan (tidak diubah). Jika Anda perlu memodifikasi string, gunakan array karakter (char) daripada pointer ke string konstan.

PENUTUP

A. Kesimpulan

- Pointer adalah Variabel yang Menyimpan Alamat Memori:

Pointer adalah variabel khusus yang menyimpan alamat memori dari variabel lain atau objek dalam program.

- Manipulasi Nilai Melalui Alamat Memori:

Pointer memungkinkan manipulasi langsung terhadap nilai variabel melalui alamat memori, memungkinkan penggunaan memori yang lebih efisien dan pengaksesan yang cepat.

- Penggunaan Alokasi Memori Dinamis:

Pointer memungkinkan alokasi memori dinamis selama runtime menggunakan fungsi malloc(), calloc(), dan realloc(), serta membebaskan memori menggunakan fungsi free(). Hal ini memberikan fleksibilitas dalam penggunaan memori sesuai kebutuhan program.

- Penggunaan dalam Fungsi:

Pointer digunakan dalam parameter fungsi untuk memungkinkan fungsi memodifikasi nilai variabel yang diteruskan ke dalamnya.

Operasi Pointer:

Referensi (&): Digunakan untuk mendapatkan alamat memori dari variabel.

Dereferensi (*): Digunakan untuk mengakses atau memodifikasi nilai yang disimpan pada alamat memori tertentu.

DAFTAR PUSTAKA

https://www.petanikode.com/c-pointer/

https://jansutris10.medium.com/belajar-pemrograman-c-29f6ea39e56d

http://moenawar.web.id/wp-content/uploads/2018/03/SD-Lab-5-6-pointer-dan-fungsi.pdf

https://elektro.um.ac.id/wp-content/uploads/2016/04/Dasar-Pemrograman-Modul-7-Pointer.pdf