MENGENAL TIPE DATA POINTER

Yuanita Frinka Dwi Arimbi

185100033

Fakultas Komputer

yuanitafrinka.student@umitra.ac.id

Abstract

Saat programmer mendeklarasikan sebuah variabel (peubah) di dalam program, maka program akan membuat sebuah lokasi memori untuk menyimpan data dengan alamat tertentu. Lokasi ini adalah lokasi yang tetap selama program berjalan. Sehingga, apabila tidak dipergunakan, lokasi tersebut tetap mengurangi jumlah memori secara keseluruhan. Delphi menyediakan tipe data lain untuk mengatasi kekurangan ini. Tipe data tersebut bernama pointer.

Pointer adalah penunjuk pada lokasi memori tertentu (dalam hal ini sama dengan variabel biasa), tetapi penunjukkan memorinya bisa diubah ke alamat lain (istilah kerennya dinamis). Jika penunjuk tadi tidak digunakan lagi, kita dapat menghapusnya dari memori dengan perintah dispose. Sedangkan membuat variabel pointer, saya memakai perintah new

Kata Kunci : Tipe data pointer

A. PENDAHULUAN

Bahasa pemrograman, merupakan adalah suatu komando atau perintah yang dibuat manusia untuk membuat komputer menjadi memiliki fungsi tertentu. bahasa pemrograman ini adalah suatu susunan aturan penulisan (sintaks) yang membentuk kode kode yang kemudian akan diterjemahkan oleh program kompilasi menjadi bahasa rakitan (assembly) dan terus diolah hingga dimengerti oleh mesin. Komputer hanya mengerti bahasa mesin.

Bagi pemula, belajar bahasa pemrograman memang sulit, namun bukan berarti itu penghalang bagi kita untuk enggan belajar. Di Fakultas Informatika yang notabene khusus mempelajari ilmu-ilmu komputer sekalipun, tidak semua mahasiswanya pandai dalam bahasa pemrograman tertentu. Hal itu dikarenakan tidak semua orang mempunyai pemikiran dan kreativitas yang sama sehingga daya nalar terhadap sebuah bahasa pemrograman didapatkan hasil yang berbeda-beda untuk setiap orang. Perlu anda ketahui bahwa Program yang ditulis dengan menggunakan bahasa pascal mempunyai ciri yang terstruktur sehingga mudah dipahami maupun dikembangkan oleh pemogram. Pada suatu bahasa pemrograman umumnya telah

menyediakan tipe – tipe data yang sederhana(simple) maupun yang terstruktur. Termasuk di bahasa pascal.

B. PEMBAHASAN / STUDI KASUS

Pengertian Pointer

Dalam ilmu komputer, pointer adalah tipe data bahasa pemrograman yang nilainya mengacu untuk menunjuk langsung nilai lain yang tersimpan di tempat lain dalam suatu memori komputer menggunakan alamat yang ada. Untuk bahasa pemrograman tingkat tinggi, pointer secara efektif dapat mengambil tempat pada general purpose registers di bahasa tingkat rendah seperti bahasa assembly atau kode mesin, tetapi bila memori tersedia. Pointer adalah implementasi sederhana, lebih konkret tentang tipe data referensi yang lebih abstrak. Beberapa bahasa mendukung beberapa jenis pointer, walaupun ada yang memiliki lebih banyak pembatasan penggunaannya dari pada yang lain. Sebagai analogi, sebuah nomor halaman dalam buku dapat dianggap sebagai pointer ke halaman yang sesuai; dereferencing seperti pointer akan dilakukan dengan membalik ke halaman dengan nomor halaman yang diberikan.

Pointer dapat meningkatkan kinerja data secara signifikan agar operasi berulang seperti traversing string, tabel lookup,

tabel kontrol dan struktur pohon. Secara khusus, sering jauh lebih efisien dalam waktu dan space untuk pointer dari pada menggunakan variable tipe data biasa.

Pointer juga digunakan untuk menyimpan alamat dari titik masuk untuk memanggil subrutin dalam pemrograman prosedural dan untuk run-time penghubung ke dynamic link libraries (DLL). Dalam pemrograman berorientasi objek, pointer ke fungsi yang digunakan untuk mengikat/menjilid method, ini sering disebut tabel method virtual.

Sementara "pointer" telah digunakan untuk merujuk pada referensi yang ada pada umumnya, hal ini berlaku lebih baik bagi struktur data yang secara eksplisit memungkinkan antarmuka (interface) pointer untuk dimanipulasi (secara aritmetik melalui aritmetik pointer) sebagai alamat memori. Karena pointer memungkinkan akses yang baik dilindungi maupun tidak dilindungi ke memori yang dituju.

Pointer primitif adalah pointer sering disimpan dalam format yang sama dengan integer. Namun, mencoba untuk dereference atau "mencari" pointer yang nilainya tidak pernah menjadi alamat memori yang valid akan menyebabkan program menjadi crash.

Deskripsi Formal:

Dalam ilmu komputer, pointer adalah macam referensi. Sebuah data primitif (primitive) adalah setiap datum yang dapat dibaca dari atau ditulis ke memori komputer menggunakan satu akses memori (misalnya, kedua byte dan kata (word) yang primitif). Sebuah data agregat (aggregate) adalah sekelompok data

primitif yang logis berdekatan di memori dan yang dipandang secara kolektif sebagai satu datum (misalnya, agregat bisa saja 3 byte logis berdekatan, maka nilai-nilai itu lah yang mewakili 3 koordinat titik di dalam ruang memori), ketika seluruh agregat yang terdiri dari jenis primitive yang sama, agregat dapat disebut sebagai sebuah array, dalam arti, multi-byte primitif adalah array dari byte. Dalam konteks definisi ini, byte adalah primitif terkecil, masing-masing menentukan alamat memori byte yang berbeda. Alamat memori dari byte pertama pada sebuah datum dianggap sebagai alamat memori (atau base alamat memori) dari seluruh datum.

Sebuah pointer memori adalah primitive, nilai yang dimaksudkan untuk digunakan sebagai alamat memori itu disebut

sebagai pointer petunjuk ke alamat memori. Hal ini juga disebut bahwa pointer menunjuk ke datum [dalam memori] ketika nilai pointer berupa alamat memori datum itu. Secara lebih umumnya, pointer adalah jenis data referensi, dan agar memperoleh pointer dereference maka datum pointer disimpan di lokasi yang berbeda di dalam memori. Fitur yang membedakan pointer dengan jenis referensi lain adalah nilai pointer yang dimaksud agar ditafsirkan sebagai alamat memori, yang merupakan konsep 'tingkat rendah'.

Saat membuat struktur data seperti daftar, deret dan percabangan, perlu memiliki pointer untuk membantu mengelola bagaimana struktur dapat diimplementasikan dan dikendalikan.

Contoh umum dari pointer adalah start pointer, end pointer, dan stack pointer. Pointer ini dapat menjadi absolute (alamat fisik yang sebenarnya atau alamat virtual di memori virtual) atau relative (sebuah offset dari alamat awal

absolute ("dasar") yang biasanya menggunakan bit yang lebih sedikit dari pada alamat lengkap, tetapi biasanya juga akan membutuhkan satu tambahan operasi aritmatika untuk menyelesaikannya).

Dua byte offset, berisi 16-bit unsigned integer, dapat digunakan untuk menyediakan pengalamatan relative sampai dengan 64 kilobyte struktur data.

Secara umum, meskipun, skema seperti itu banyak masalah, dan untuk kemudahan bagi programmer maka ruang alamat datar biasanya lebih disukai. Sebuah byte offset, seperti nilai ASCII karakter heksadesimal dapat digunakan untuk menunjuk nilai integer alternatif dalam array. Dengan cara ini, karakter dapat sangat efisien diterjemahkan dari 'data mentah' ke indeks sekuensial dapat digunakan dan kemudian ke alamat absolut tanpa menggunakan tabel lookup. Digunakan dalam mengontrol tabel Kontrol tabel digunakan untuk mengontrol aliran program (program flow), biasanya membuat ekstensif menggunakan pointer. Pointer biasanya tertanam dalam sebuah entry tabel. Misalnya, digunakan untuk menyimpan entry point bagi subrutin yang akan dijalankan, dan berdasarkan kondisi tertentu yang didefinisikan dalam entri tabel yang sama. Namun pointer hanya dapat berupa indeks untuk memisahkan data yang lain tetapi

terikat, tabel terdiri dari suatu array dari alamat yang aktual atau alamat sendiri (tergantung pada konstruksi bahasa pemrograman yang tersedia).

Pointer juga dapat digunakan untuk titik (kembali) ke entri tabel sebelumnya (seperti dalam pengolahan loop) atau maju melewati beberapa entri tabel (seperti pada switch atau "awal" keluar dari loop).

Abstraksi pointer yang sangat tipis di atas merupakan kemampuan

pengalamatan yang dikembangkan dengan arsitektur paling modern. Dalam skema sederhana, alamat atau indeks numerik, ditugaskan bagi setiap unit memori dalam sistem, di mana unit ini biasanya baik byte atau kata (word), secara efektif dapat mengubah semua memori ke array yang sangat besar. Kemudian, jika kita memiliki suatu alamat, sistem itu menyediakan operasi untuk mengambil nilai yang tersimpan dalam unit memori pada alamat (biasanya menggunakan register tujuan umum didalam mesin yang digunakan). Dalam kasus biasa, pointer itu cukup besar untuk menampung alamat lebih dari beberapa unit memori yang berada di dalam sistem. Ini memperkenalkan kemungkinan bahwa program mungkin berusaha untuk mengakses alamat bila tidak ada unit memori, baik karena memori tidak cukup dipasang (yaitu di luar jangkauan memori yang tersedia) atau arsitektur tidak mendukung alamat tersebut. Kasus pertama mungkin, dalam platform tertentu seperti arsitektur Intel x86, disebut kesalahan segmentasi (segfault). Kasus kedua adalah mungkin dalam implementasi pada AMD64, di mana pointer yang

memiliki panjang 64 bit dan alamat hanya memperpanjang sampai 48 bit. Disini pointer harus sesuai dengan aturan-aturan tertentu (alamat kanonik), jadi jika pointer nonkanonik adalah dereferenced, akan berakibat prosesor melakukan kesalahan perlindungan umum (general protection fault).

Di sisi lain, beberapa sistem memiliki lebih banyak unit memori dari pada alamat. Dalam hal ini, skema yang lebih kompleks seperti segmentasi memori atau paging dikelola untuk menggunakan bagian yang berbeda dari memori pada waktu yang berbeda pula. Inkarnasi terakhir dari arsitektur x86 mendukung hingga 36 bit dari alamat memori fisik, yang dipetakan ke ruang alamat 32-bit linear melalui mekanisme paging PAE. Jadi, hanya 1 / 16 dari total memori yang mungkin dapat diakses pada suatu waktu. Contoh lain dalam kelompok komputer yang sama adalah 16-bit dalam keadaan terlindungi dari prosesor 80286 yang meskipun mempunyai 16 MiB memori fisik, dapat mengakses hingga 1 GB dari memori virtual, namun kombinasi dari 16-bit alamat dan segmen register dapat mengakses lebih dari 64 KiB dalam satu struktur data rumit. Beberapa pembatasan aritmetik pointer ANSI mungkin memiliki model memori tersegmentasi dari kelompok prosesor ini.

Dalam rangka untuk mengembangkan interface yang konsisten, beberapa arsitektur menyediakan I / O yang dipetakan ke dalam memori, yang memungkinkan beberapa alamat untuk menuju ke unit memori sementara yang lain mengacu pada register perangkat perangkat lain di komputer.

Ada konsep analog seperti file offset, indeks array, dan referensi object remote yang melayani beberapa tujuan yang sama sebagai alamat untuk jenis object lainnya.

Pointer secara langsung didukung tanpa pembatasan dalam bahasa seperti PL / I, C, C++, Pascal, dan kebanyakan bahasa perakitan. Ketika berhadapan dengan

array, operasi pencarian kritis biasanya perhitungan alamat disebut tahap perhitungan alamat yang melibatkan pembentukan sebuah array.

Dalam struktur data lain, seperti daftar terhubung (linked list), pointer digunakan sebagai referensi secara eksplisit yang mengikat salah satu bagian dari struktur yang lain. Pointer digunakan untuk menumpangi parameter bagi referensi. Hal ini berguna jika programmer menginginkan modifikasi fungsi untuk parameter, untuk dapat dilihat ke pemanggilan fungsi itu. Ini juga berguna untuk menampilkan nilai hasil perkalian dari dalam fungsi yang telah dibuat.

Pointer juga dapat digunakan untuk mengalokasikan atau DEALLOCATE variabel dinamis dan array dalam memori. Karena variable akan sering menjadikan memori penuh setelah selesai ngerjakan perintah, itu bisa menjadi pemborosan memori bila kita menyimpannya, dan oleh karena itu praktik yang baik untuk DEALLOCATE itu (menggunakan referensi pointer asli) ketika tidak lagi diperlukan. Kegagalan untuk melakukannya dapat mengakibatkan kehabisan memori (di mana memori bebas yang tersedia akan secara bertahap berkurang atau dalam kasus yang parah dengan secara cepat akan penuh).

Menurut Frieyadie. 2007. Edisi Revisi Buku Pemrograman C++ dengan Borland C++ 5.02. Pointer adalah sebuah variabel yang berisi alamat lain. Suatu pointer dimaksudkan untuk menunjukan ke suatu alamat memori sehingga alamat dari suatu variabel dapat diketahui dengan mudah.

Contoh Program:

```
#include <stdio.h>           //Mengakses
file pada library stdio.h
#include <stdlib.h>          //Mengakses
file pada library stdlib.h
int main (void){             //Fungsi main
yang harus ada dalam sebuah program
C
    int A = 5, H;
    int *ptrA, *ptrH;         //Deklarasi
pointer sebagai integer
    ptrA = &A;                //Alamat lokasi
memori dari variabel A disimpan pada
ptrA
    H = *ptrA;                //Nilai data
yang alamat lokasi memorinya
disimpan pada ptrA,

disimpan di variabel H
    ptrH = &H;                //Alamat lokasi
memori dari variabel H disimpan pada
ptrH
                                //          %x
menampilkan nilai hexa
    printf("A=%d\n &A=%x \n ptrA=%x
\n *ptrA=%d", A, &A, ptrA, *ptrA);
    printf("\n\nH=%d\n      &H=%X\n
ptrH=%X\n *ptrH=%d\n\n", H, &H,
ptrH,*ptrH);    system ("pause");
//File yang terdapat pada stdlib.h
return (0);           //Nilai balikan
fungsi main() adalah 0
}
```

Arti pointer dalam bahasa sehari-hari adalah petunjuk atau dapat dikatakan sebagai penentu atau dapat diartikan sebagai tipe data yang nilainya

mengarah pada nilai yang terdapat pada sebuah area memori (alamat memori). Namun dalam bahasa C, Pointer bisa berfungsi sebagai variabel array berarti pointer sebagai penunjuk elemen array ke-0 dalam variabel C.

Kegunaan Pointer Di C++ :

Kegunaan pointer yang utama adalah untuk menyimpan alamat memori dari sebuah variabel (data type atau object dari class), menyimpan alamat dari sebuah fungsi (function pointer) dan dapat berfungsi untuk mengirimkan "Parameter yang berupa variabel" ke dalam fungsi, artinya nilai variabel bisa diubah di dalam fungsi serta dapat membuat variabel dinamis.

Tipe Data Pointer :

Tipe_data merupakan tipe dari data yang ditunjuk, bukan tipe dari pointer-nya. Contoh :

1. Mensubstitusikan address sebuah variabel ke pointer dengan memakai address operator &

```
int x;
int *ptr;
ptr = &x;
```

2. Mensubstitusikan address awal sebuah array ke pointer

```
char t[5]; char *ptr; ptr = t;
```

3. Mensubstitusikan address salah satu elemen array dengan address operator

```
char t[5] ; char *ptr; ptr = &t[3];
```

4. Mensubstitusikan address awal character string ke pointer char

```
char *ptr;
ptr = "jakarta"
```

5. Mensubstitusikan NULL pada pointer.

NULL ada pointer kosong, menunjukkan suatu status dimana pointer itu belum diinisialisasikan dengan sebuah address tertentu.

Operator Pointer :

Ada 2 operator pointer yang dikenal secara luas, yaitu operator & dan operator *.

Operator &

Operator & merupakan operator alamat. Pada saat pendeklarasian variable, user tidak diharuskan menentukan lokasi sesungguhnya pada memory, hal ini akan dilakukan secara otomatis oleh kompiler dan operating sysem pada saat run-time. Jika ingin mengetahui dimana suatu variable akan disimpan, dapat dilakukan dengan memberikan tanda ampersand (&) didepan variable , yang berarti "address of". Contoh : `ted = &input;` Penulisan tersebut berarti akan memberikan variable ted alamat dari variable input. Karena variabel input diberi awalan karakter ampersand (&), maka yang menjadi pokok disini adalah alamat dalam memory, bukan isi variable.

Misalkan input diletakkan pada alamat 1776 kemudian dituliskan instruksi sbb :

```
input = 25; fred = input; ted = &input;
```

Operator *

Operator * merupakan operator reference. Dengan menggunakan pointer,kita dapat mengakses nilai yang tersimpan secara langsung dengan memberikan awalan operator asterisk (*)

pada identifier pointer, yang berarti "value pointed by". Contoh :

```
beth = *ted;
```

(dapat dikatakan:"beth sama dengan nilai yang ditunjuk oleh ted") `beth = 25,`

karena ted dialamat 1776, dan nilai yang berada pada alamat 1776 adalah 25.

Macam – macam Pointer

Pointer Bertipe Void

Pada C++ terdapat pointer yang dapat menunjuk ke tipe data apapun, pointer semacam ini dideklarasikan dengan tipe void sehingga sering dikenal dengan istilah Void Pointer. Berikut ini contoh listing program yang menggunakan void pointer.

Pointer Aritmetika

Elemen-elemen array biasanya diakses melalui indeksny, sebenarnya ada cara lain yang lebih efisien, yaitu dengan menggunakan pointer. Pointer semacam ini disebut dengan istilah pointer aritmetika. Konsep dasar dari pointer aritmetika ini adalah melakukan operasi aritmetika terhadap variabel yang bertipe pointer.

Pointer NULL

Pada saat program dijalankan, pointer akan menunjuk ke alamat acak pada memori, sehingga diperlukan inisialisasi agar hal tersebut tidak terjadi. Dalam C++ terdapat sebuah cara untuk membuat pointer tidak menunjuk ke alamat manapun, yaitu dengan mengisikan pointer tersebut dengan nilai NULL. Karena hal inilah maka pointer tersebut sering dinamakan pointer NULL (NULL Pointer).Sebagai contoh kita mempunyai pointer p, dan kita ingin melakukan inisialisasi pada pointer tersebut dengan nilai NULL, maka sintaknya adalah `P=NULL;`

DEKLARASI POINTER

Seperti halnya variabel yang lain, variabel pointer juga harus

dideklarasikan terlebih dahulu sebelum digunakan.

Bentuk Umum :

Tipe_data

*nama_pointer;

//Contoh9_1

```
int *px;
char *sh;
Contoh Program :
#include "IOSTREAM.h"
#include "conio.h"
void main()
{ int x, y; /* x dan y bertipe int */
int *px; /* px pointer yang menunjuk objek */
```

```
clrscr();
x = 87;
px = &x; /* px berisi alamat dari x */
y = *px; /* y berisi nilai yang ditunjuk px */
cout<<"Alamat x ="<<&x <<"\n";
cout<<"Isi px = \n", px);
cout<<"Isi x = \n", x);
cout<<"Nilai yang ditunjuk oleh px = \n", *px);
cout<<"Nilai y = \n", y);
getch();
}
```

Operasi-operasi Pointer

Operasi Penugasan

Suatu variabel pointer seperti halnya variabel yang lain, juga bisa mengalami operasi penugasan. Nilai dari suatu variabel pointer dapat disalin ke variabel pointer yang lain.

Operasi Aritmatika

Suatu variabel pointer hanya dapat dilakukan operasi aritmatika dengan nilai integer saja. Operasi yang biasa dilakukan adalah operasi penambahan dan pengurangan.

Operasi Logika

Operasi logika juga dapat dilakukan pada sebuah variabel pointer.

OPERASI POINTER

* Operasi Penugasan

Suatu variabel pointer seperti halnya variabel yang lain, juga bisa mengalami operasi penugasan. Nilai dari suatu variabel pointer dapat disalin ke variabel pointer yang lain.

//Contoh9_2

```
#include "iostream.h"
#include "conio.h"
void main()
{ float *x1,y, *x2;
clrscr();
y = 13.45;
x1 = &y; /* Alamat dari y
disalin ke variabel x1 */
x2 = x1; /* Isi variabel x1
disalin ke variabel x2 */
cout<<"Nilai variabel y = "<<y<<"
ada di alamat "<<x1<<"\n";
cout<<"Nilai variabel y = "<<y<<"
ada di alamat "<<x2<<"\n";
getch();
}
```

* Operasi Aritmatika

Suatu variabel pointer hanya dapat dilakukan operasi aritmatika dengan nilai integer saja. Operasi yang biasa dilakukan adalah operasi penambahan dan pengurangan. Operasi penambahan dengan suatu nilai menunjukkan lokasi data berikutnya (index selanjutnya) dalam memori. Begitu juga operasi pengurangan.

//Contoh9_3

```
#include <iostream.h>
#include <conio.h>
void main()
{ int nilai[3], *penunjuk;
clrscr();
nilai[0] = 125;
```

```

nilai[1] = 345;
nilai[2] = 750;
penunjuk = &nilai[0];
cout<<"Nilai "<<*penunjuk<<" ada di
alamat memori " <<penunjuk<<"\n";
cout<<"Nilai " <<*(penunjuk+1) <<"
ada di alamat memori "
<<penunjuk+1<<"\n";
cout<<"Nilai " <<*(penunjuk+2) <<"
ada di alamat memori "
<<penunjuk+2<<"\n";
getch();
}

```

* Operasi Logika

Suatu pointer juga dapat dikenai operasi logika.

```

//Contoh9_4
#include<iostream.h>
#include "conio.h"
void main()
{ int a = 100, b = 200, *pa, *pb;
clrscr();
pa = &a;
pb = &b;
cout<<"nilai pa= "<<pa<<" nilai pb=
"<<pb<<"\n";
if(pa < pb)
cout<<"pa menunjuk ke memori lebih
rendah dari pb\n";
if(pa == pb)
cout<<"pa menunjuk ke memori yang
sama dengan pb\n";
if(pa > pb)
cout<<"pa menunjuk ke memori lebih
tinggi dari pb\n";
getch();}

```

C. ID SECURITY

QWTD4452377-ASP-5244107

D. KESIMPULAN

Pada C++ sebuah pointer adalah sebuah objek yang menyimpan sebuah alamat

(contohnya lokasi memori) dimana terdapat data lain yang disimpan. Alamat memori merupakan integer, jadi sebuah pointer biasanya direpresentasikan dengan (unsigned) int. Yang membuat pointer lebih dari integer yang biasa adalah kita dapat mengakses data yang ditunjuk oleh pointer tersebut. Operator pada pointer ada dua, yaitu operator* dan operator&. Pointer sama seperti variabel lain, harus dideklarasikan terlebih dahulu sebelum digunakan.

Bahasa pemrograman assembly dibagi atas 3 tipe data, yaitu tipe data terstruktur, tipe data sederhana, dan tipe data pointer

- Tipe data sederhana dibagi atas tiga, yaitu tipe data integer, real karakter, dan tipe data boolean
- Tipe data terstruktur terbagi atas tipe data array, dan tipe data record
- Tipe data pointer merupakan tipe data yang berisi alamat pada register.

E. DISKUSI

Saya bersama teman saya mendiskusikan hasil dari materi ini dengan baik.

Saya : Apakah materi saya dapat membantu anda?

Dwi : Iya materi ini sangat membantu saya dalam proses belajar memahami tipe data pointer

Dwi : Apa macam-macam tipe data pointer?

Saya : Pointer ada 2 macam, yaitu:

1. Pointer bertipe Deklarasi : `^tipe_data;`
2. Pointer tak bertipe Deklarasi : `pointer;`

Dwi : Apakah artikel ini pantas untuk dipublikasikan?

Saya : Tentu saja pantas, karna ini sangat membantu proses pembelajaran dan bahasanya pun mudah dipahami dan dimengerti

F. REFERENCE

- [1] O. M. Febriani and A. S. Putra, "Sistem Informasi Monitoring Inventori Barang Pada Balai Riset Standardisasi Industri Bandar Lampung," *J. Inform.*, vol. 13, no. 1, pp. 90–98, 2014.
- [2] A. S. Putra, "Paperplain: Execution Fundamental Create Application With Borland Delphi 7.0 University Of Mitra Indonesia," 2018.
- [3] A. S. Putra, "2018 Artikel Struktur Data, Audit Dan Jaringan Komputer," 2018.
- [4] A. S. Putra, "ALIAS MANAGER USED IN DATABASE DESKTOP STUDI CASE DB DEMOS."
- [5] A. S. Putra, "COMPREHENSIVE SET OF PROFESSIONAL FOR DISTRIBUTE COMPUTING."
- [6] A. S. Putra, "DATA ORIENTED RECOGNITION IN BORLAND DELPHI 7.0."
- [7] A. S. Putra, "EMBARCADERO DELPHI XE 2 IN GPU-POWERED FIREMONKEY APPLICATION."
- [8] A. S. Putra, "HAK ATAS KEKAYAAN INTELEKTUAL DALAM DUNIA TEKNOLOGY BERBASIS REVOLUSI INDUSTRI 4.0."
- [9] A. S. Putra, "IMPLEMENTASI PERATURAN PERUNDANGAN UU. NO 31 TAHUN 2000 TENTANG DESAIN INDUSTRI BERBASIS INFORMATION TECHNOLOGY."
- [10] A. S. Putra, "IMPLEMENTATION OF PARADOX DBASE."
- [11] A. S. Putra, "IMPLEMENTATION OF TRADE SECRET CASE STUDY SAMSUNG MOBILE PHONE."
- [12] A. S. Putra, "IMPLEMENTATION PATENT FOR APPLICATION WEB BASED CASE STUDI WWW. PUBLIKLAMPUNG.COM."
- [13] A. S. Putra, "IMPLEMENTATION SYSTEM FIRST TO INVENT IN DIGITALLY INDUSTRY."
- [14] A. S. Putra, "MANUAL REPORT & INTEGRATED DEVELOPMENT ENVIRONMENT BORLAND DELPHI 7.0."
- [15] A. S. Putra, "PATENT AS RELEVAN SUPPORT RESEARCH."
- [16] A. S. Putra, "PATENT FOR RESEARCH STUDY CASE OF APPLE. Inc."

- [17] A. S. Putra, "PATENT PROTECTION FOR APPLICATION INVENT."
- [18] A. S. Putra, "QUICK REPORT IN PROPERTY PROGRAMMING."
- [19] A. S. Putra, "REVIEW CIRCUIT LAYOUT COMPONENT REQUIREMENT ON ASUS NOTEBOOK."
- [20] A. S. Putra, "REVIEW TRADEMARK PATENT FOR INDUSTRIAL TECHNOLOGY BASED 4.0."
- [21] A. S. Putra, "TOOLBAR COMPONENT PALLETTE IN OBJECT ORIENTED PROGRAMMING."
- [22] A. S. Putra, "WORKING DIRECTORY SET FOR PARADOX 7."
- [23] A. S. Putra, "ZQUERY CONNECTION IMPLEMENTED PROGRAMMING STUDI CASE PT. BANK BCA Tbk."
- [24] A. S. Putra, D. R. Aryanti, and I. Hartati, "Metode SAW (Simple Additive Weighting) sebagai Sistem Pendukung Keputusan Guru Berprestasi (Studi Kasus: SMK Global Surya)," in *Prosiding Seminar Nasional Darmajaya*, 2018, vol. 1, no. 1, pp. 85–97.
- [25] A. S. Putra and O. M. Febriani, "Knowledge Management Online Application in PDAM Lampung Province," in *Prosiding International conference on Information Technology and Business (ICITB)*, 2018, pp. 181–187.
- [26] A. S. Putra, O. M. Febriani, and B. Bachry, "Implementasi Genetic Fuzzy System Untuk Mengidentifikasi Hasil Curian Kendaraan Bermotor Di Polda Lampung," *SIMADA (Jurnal Sist. Inf. dan Manaj. Basis Data)*, vol. 1, no. 1, pp. 21–30, 2018.
- [27] A. S. Putra, H. Sukri, and K. Zuhri, "Sistem Monitoring Realtime Jaringan Irigasi Desa (JIDES) Dengan Konsep Jaringan Sensor Nirkabel," *IJEIS (Indonesian J. Electron. Instrum. Syst.)*, vol. 8, no. 2, pp. 221–232.
- [28] D. P. Sari, O. M. Febriani, and A. S. Putra, "Perancangan Sistem Informasi SDM Berprestasi pada SD Global Surya," in *Prosiding Seminar Nasional Darmajaya*, 2018, vol. 1, no. 1, pp. 289–294.