Algoritma dan Pemrograman Lanjut

Pertemuan Ke-10
Pointer 1

Disusun Oleh :
Wilis Kaswidjanti, S.Si.,M.Kom.

Jurusan Teknik Informatika
Fakultas Teknologi Industri
Universitas Pembangunan Nasional "Veteran"
Yogyakarta

Algoritma dan Pemrograman Lanjut

Judul Materi : Pointer 1

Deskripsi Materi : Materi ini membahas tipe data pointer, pendeklarasian dan cara pengaksesannya

Tujuan Instruksional Khusus :

1. Mendefinisikan dan menggunakan tipe data pointer
2. Mendeskripsikan tipe data pointer
3. Memahami kegunaan pointer

Referensi :

- Buku Teks

  Munir, Rinaldi (2005), *Algoritma dan Pemrograman dalam Bahasa Pascal dan C*, Buku 2, Edisi Ketiga, Penerbit Informatika Bandung.

  Charibaldi, N. (2004), *Modul Kuliah Algoritma Pemrograman II*, Edisi Kedua, Yogyakarta

- Buku Acuan/Referensi

  Brassard, Gilles (1999), *Fundamentals of algorithma*, PrinteceHall.

  Jarne, Stroustrup B. (1997), *C++ Programming language*, AT &T.

  Kristanto, Andri (2003), *Algoritma pemrograman C++*, Graha Ilmu.

  Schildt,Herbert (2000), *The Complete Reference C++*, McGraw-Hill.

  Sedgewick, R. (2000), *Algoritma Third edition In C part 5*, Addison Wesley.

POINTER 1

PENDAHULUAN

Variabel pointer sering disebut sebagai variabel yang menunjuk obyek lain, karena variabel pointer atau pointer adalah variabel yang berisi alamat di memori komputer dari suatu obyek lain, yaitu obyek yang ditunjuk oleh pointer yang mempunyai nilai tertentu.

ISI

A. Bentuk umum deklarasi variable pointer :

Algoritma (Pseodocode) :

nama_pointer : pointer to tipedata

C++ :

```
tipedata  *nama_pointer;
```

(*deklarasi pointer null*)

```
nama_pointer = (tipedata *) malloc(size_t size);
```

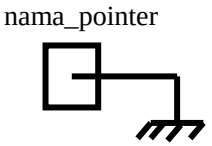
(*deklarasi pointer kosong***)**

Dengan :

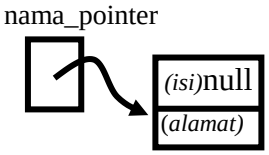
- tipedata adalah tipe dasar nilai yang berada di memori yang ditunjuk oleh *pointer*.
- nama_pointer adalah nama variable *pointer*.
- (*) adalah operator memori yang fungsinya untuk mengembalikan nilai variable pada alamatnya yang ditentukan oleh *operand*.
- `malloc(size_t size)` adalah deklarasi pengalokasian memori kosong dengan ukuran kapasitas sebesar `size`

Ilustrasi :

- pointer null



- pointer kosong



Contoh :

Algoritma (Pseodocode) :

p : pointer to integer

nilai : pointer to real

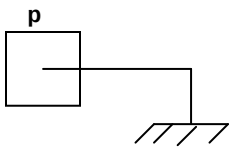
s : pointer to char

C++ :

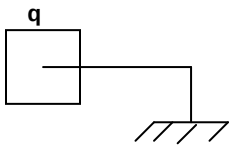
```
int *p;
float *nilai;
char *s;
```

Contoh lain dalam bahasa C++ dan dengan ilustrasi :

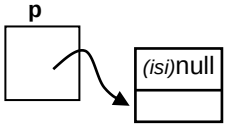
```
int *p;
```



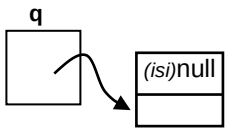

```
int *q;
```




```
p=(int *) malloc(sizeof(int));
```



```
q=(int *) malloc(sizeof(int));
```



B. Pengaksesan dengan Pointer

Untuk mengakses nilai/isi pada memori yang ditunjuk oleh pointer dipakai simbol '*'

Contoh :

```
*p = 10;
*q = 20;
```

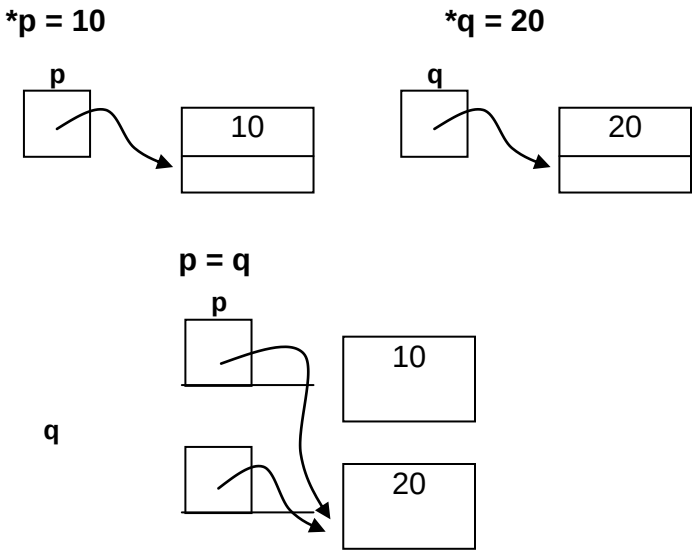
Pointer menunjuk memori yang ditunjuk pointer lain :

Contoh :

```
p = q;
```

berarti p menunjuk ke alamat memori yang ditunjuk oleh q, dan dengan demikian p dan q menunjuk alamat memori yang sama.

Ilustrasi :



Contoh program Pointer 1:

```
#include <iostream.h>
#include <alloc.h>
#include <stdlib.h>
void main()
{
```

```
  int  *p, *q;
  p=(int *)malloc(sizeof(int));
  q=(int *)malloc(sizeof(int));
  *p=10;
  *q=20;
  cout<<"Isi info pointer :\n";
  cout<<"*p = "<<*p<<endl;
  cout<<"*q = "<<*q<<endl;
  cout<<"\nAlamat register pointer :\n";
  cout<<"p = "<<p<<endl;
  cout<<"q = "<<q<<endl;
  p=q;
  cout<<"\nKondisi akhir isi info pointer :\n";
  cout<<"*p = "<<*p<<endl;
  cout<<"*q = "<<*q<<endl;
}
```

Output :

```
Isi info pointer :
*p = 10
*q = 20

Alamat register pointer :
p = 0x212f01d2
q = 0x212f01ca

Kondisi akhir isi info pointer :
*p = 20
*q = 20
```

C. Operator Pointer

Ada beberapa operator yang bisa digunakan dalam pointer, yaitu :

1. Operator alamat (yang dilambangkan dengan symbol &)
2. Operator unary yang mengembalikan alamat dari *operandnya.*

Pointer menunjuk variabel statis :

Misalkan px adalah variabel bertipe pointer yang akan berisi alamat variabel lain yang bertipe integer, maka dideklarasikan :

```
int x;
int *px;
```

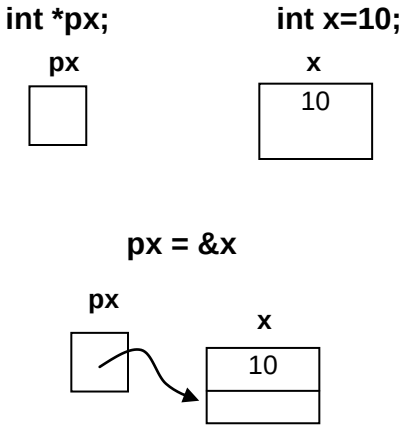
Untuk mengatur pointer agar menunjuk ke variabel lain, mula-mula pointer harus diisi dengan alamat dari variabel yang akan ditunjuk.

Operator ‘&’ digunakan untuk menyatakan alamat variabel statis yang akan ditunjuk.

Contoh :

```
px = &x;
```

Ilustrasi :



Jika suatu variabel statis sudah ditunjuk oleh pointer, isi variabel tersebut dapat diakses melalui variabel itu sendiri (**pengaksesan langsung**) atau melalui pointer (**pengaksesan tidak langsung**).

- Pengaksesan langsung dilakukan langsung oleh variabel statisnya (bukan pointernya).

  Contoh :

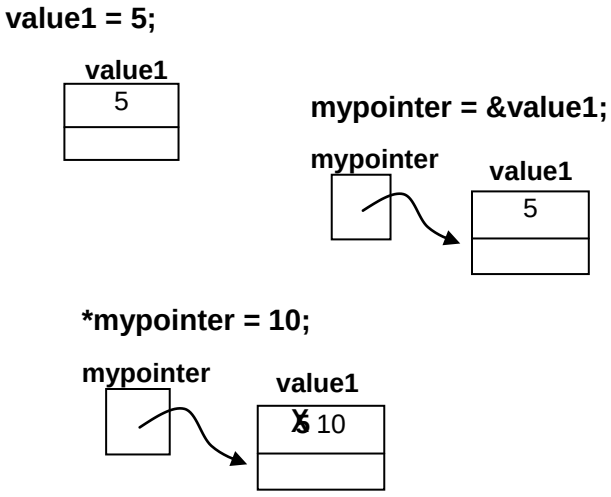
  ```
  x = 10
  ```

- Operator indirection (pengaksesan tidak langsung), berupa simbol '*'

  Contoh :

  ```
  *px = 10
  ```

Ilustrasi :



Contoh Program Pointer 2 :

```
#include<iostream.h>
main()
{ int value1 = 5, value2 = 15;
  int * mypointer;
  mypointer = &value1;
  *mypointer = 10;
  cout << "\nvalue1 = " << value1;
  cout << "\n*mypointer = " << *mypointer;
  mypointer = &value2;
  *mypointer = 20;
  cout << "\nvalue2 = " << value2;
  cout << "\n*mypointer = "<<*mypointer;
}
```

Output :

```
value1 = 10
*mypointer = 10
value2 = 20
*mypointer = 20
```

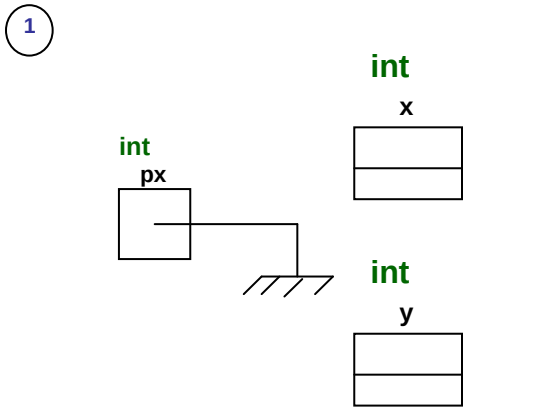
Contoh program Pointer 3 :

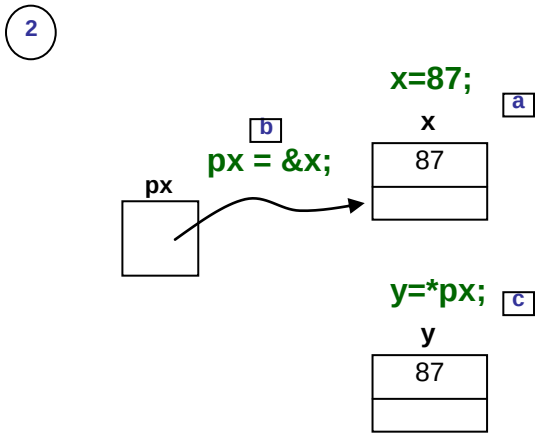
```
#include <iostream.h>
main()
{
  int x,y;  //x dan y bertipe int
  int *px;  //deklarasi px, pointer yang menunjuk obyek
            //bertipe int
a x=87;
b px=&x;    //px berisi alamat dari x
c y =*px;   //y berisi nilai yang ditunjuk px

  cout<<"alamat x= "<<&x<<endl;
  cout<<"nilai  x= "<<x<<endl;
  cout<<"alamat yang ditunjuk oleh px= "<<px<<endl;
  cout<<"nilai yang ditunjuk oleh  px= "<<*px<<endl;
  cout<<"alamat y= "<<&y<<endl;
  cout<<"nilai  y= "<<y;
}
```

Output :

```
alamat x= 0x301f2452
nilai  x= 87
alamat yang ditunjuk oleh px= 0x301f2452
nilai yang ditunjuk oleh  px= 87
alamat y= 0x301f2450
nilai  y= 87
```

Ilustrasi :





PENUTUP

Penerapan pointer yang paling umum yaitu menciptakan variable dinamis, yang memungkinkan untuk memakai memori bebas (memori yang belum dipakai) selama eksekusi program.

SOAL-SOAL

1. Apa output dari program berikut :

```
#include<iostream.h>
main()
{ int p = 5, q = 15;
  int *m;
  m = &q;
  q = 10;
  cout << "\np = " << p;
  cout << "\nq = " << q;
```

```
  cout << "\n*m = " << *m;
  *mypointer = p;
  mypointer = &p;
  cout << "\np = " << p;
  cout << "\nq = " << q;
  cout << "\n*m = " << *m;
}
```

2. Apa output dari program berikut :

```
#include<iostream.h>
#include<alloc.h>
main()
{ int j = 2, k = 3;
  int *p,*q,*r;
  p =(int *)malloc(sizeof(int));
  q =(int *)malloc(sizeof(int));
  r = p;
  *p = k;
  *q = j;
  k = *q;
  r = &j;
  cout << "\nj = " << j;
  cout << "\nk = " << k;
  cout << "\n*p = " << *p;
  cout << "\n*q = " << *q;
  cout << "\n*r = " << *r;
}
```

3. Apa output dari program berikut :

```
#include<stdio.h>
#include<alloc.h>
main()
{ int m = 10;
  int *p,*q ;
  p =(int *)malloc(sizeof(int));
  *p = m;
  q = p;
  m = *q + *p;
  p = &m;
  printf("\nm = %d",m);
  printf("\n*p = %d",*p);
  printf("\n*q = %d",*q);
}
```