

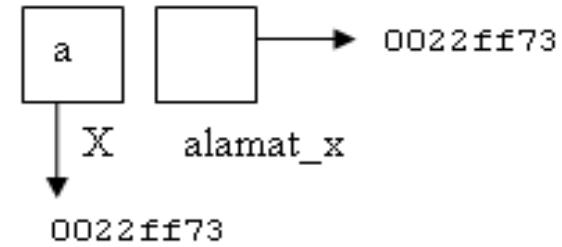
POINTER

TIB21

Pointer

- Pointer adalah suatu **variabel penunjuk**, berisi nilai yang menunjuk alamat suatu lokasi memori tertentu.
- Jadi pointer **tidak** berisi nilai data, melainkan berisi suatu **alamat memori** atau **null** jika tidak berisi data.
- Pointer yang tidak diinisialisasi disebut **dangling pointer**
- Lokasi memori tersebut bisa diwakili sebuah **variabel** atau dapat juga berupa nilai alamat memori **secara langsung**.

Ilustrasi Pointer



- Kita memiliki variabel `X` yang berisi nilai karakter `'a'`
- Oleh kompiler C, nilai `'a'` ini akan disimpan di suatu alamat tertentu di memori.
- Alamat variabel `X` dapat diakses dengan menggunakan statemen **`&X`**.
- Jika kita ingin menyimpan alamat dari variabel `X` ini, kita dapat menggunakan suatu variabel
 - misalnya **`char alamat_x = &X;`**
- `alamat_x` adalah suatu variabel yang berisi alamat dimana nilai `X`, yaitu `'a'` disimpan.
- Variabel `alamat_x` disebut variabel pointer atau sering disebut **pointer** saja.

DEKLARASI

- DeklarasiAlgoritma (Pseudocode) nama_pointer : pointer to typedata

- C++ :

typedata *nama_pointer;
(deklarasi pointer null)

nama_pointer =(typedata *) malloc(size_t size);
(deklarasi pointer kosong)

Contoh

Pseudocode

p : pointer to integer

nilai : pointer to reals

s: pointer to char

C++ :

```
int *p;
```

```
float *nilai;
```

```
char *s;
```

Deklarasi sebuah pointer kosong pada memory pada C++

```
*malloc(size_t size)
```

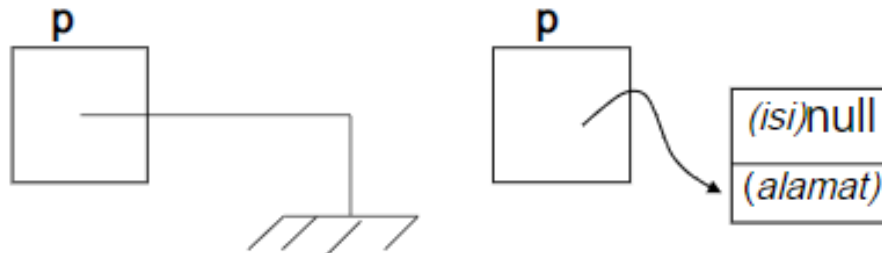
Contoh:

```
int *p;
```

```
int *q;
```

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

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



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

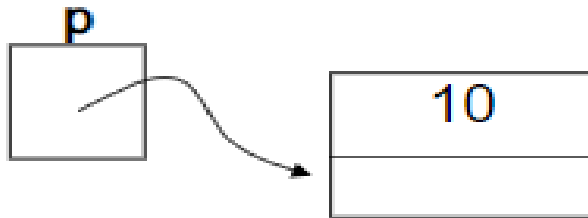
- Contoh :

`p = q;`

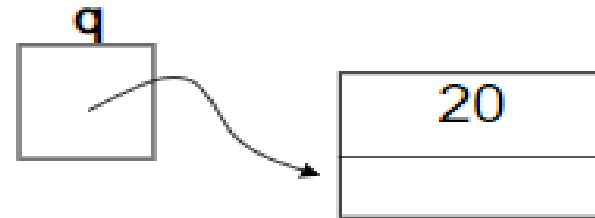
berarti p menunjuk ke alamat memori yang ditunjuk oleh q, dan dengan demikian p dan q menunjuk alamat memori yang sama.

Ilustrasi

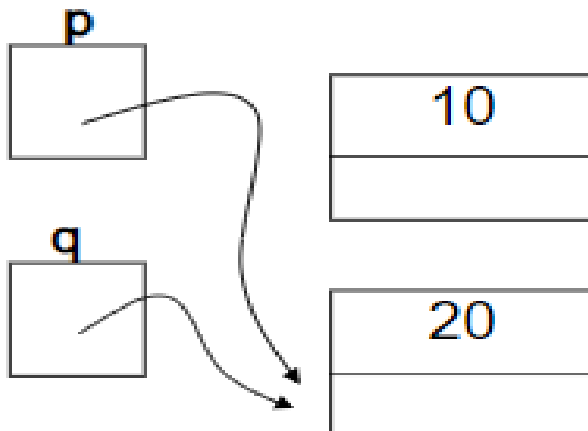
$*p = 10$



$*q = 20$

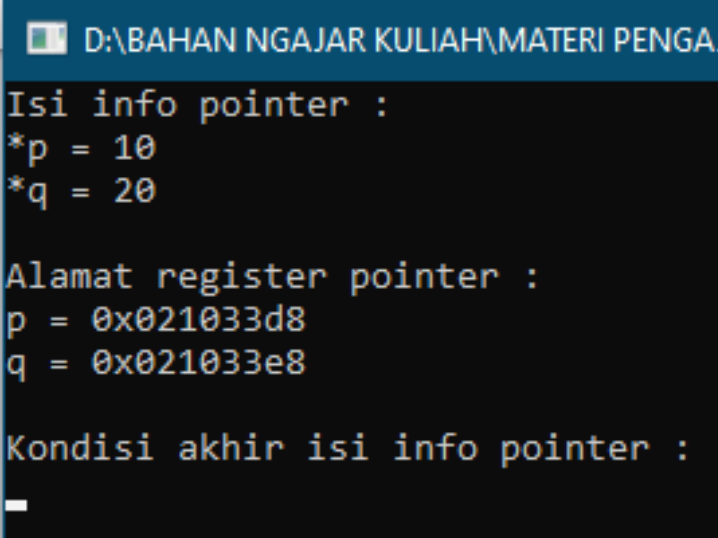


$p = q$



//contoh program Pointer1.cpp

- `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";`
- `getch();`
- `}`



D:\BAHAN NGAJAR KULIAH\MATERI PENGA.

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

Alamat register pointer :
p = 0x021033d8
q = 0x021033e8

Kondisi akhir isi info pointer :
_

Pointer vs Variabel Biasa

Variabel Biasa	Pointer
Berisi data/nilai	Berisi alamat memori dari suatu variabel tertentu
Operasi yang bisa dilakukan seperti layaknya operasi biasa: +, -, *, /	<p>Membutuhkan operator khusus: "&" yang menunjuk alamat dari suatu variabel tertentu. Operator "&" hanya dapat dilakukan kepada variabel dan akan menghasilkan alamat dari variabel itu. Contoh: <code>p = &n</code></p> <p>Yang kedua : Operator "*". Operator ini bersifat menggunakan nilai dari alamat variabel yang ditunjuk oleh pointer tersebut. Contoh: <code>int *p;</code></p>
Bersifat statis	Bersifat dinamis
Deklarasi: <code>int a;</code>	Deklarasi: <code>int *a</code>

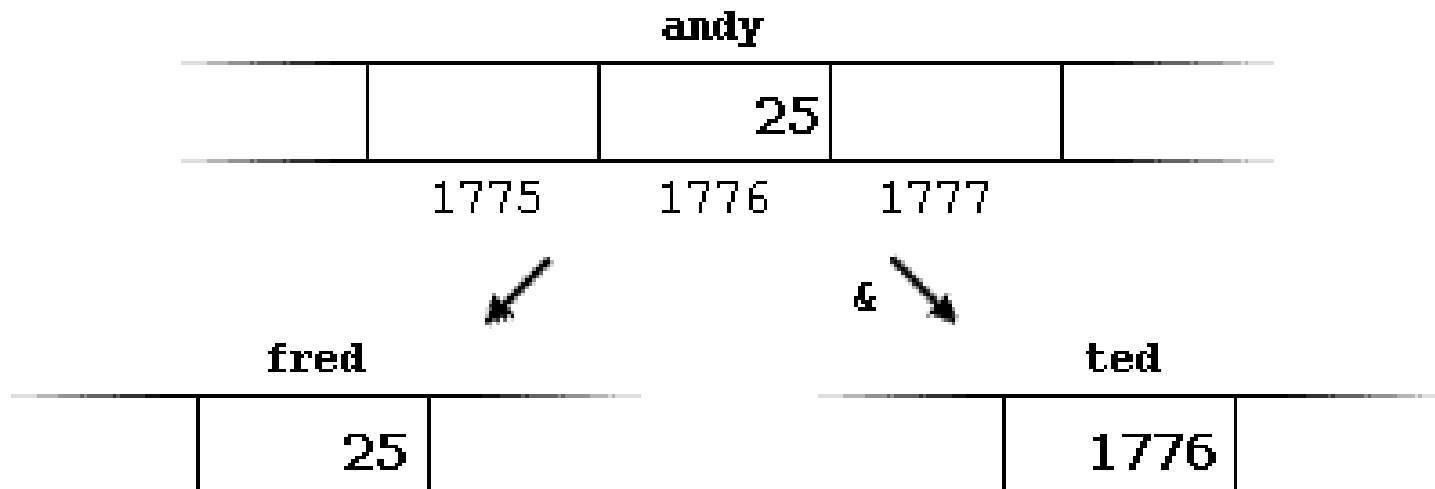
Operator Pointer

Operator *	Mendapatkan nilai data dari variabel pointer	Contoh: <pre>int *alamat; int nilai = 10; alamat = &nilai; printf("%d", *alamat);</pre>	Hasil: 10
Operator &	Mendapatkan alamat memori dari variabel pointer	Contoh: <pre>int *alamat; int nilai = 10; alamat = &nilai; printf("%p", alamat);</pre>	Hasil: 22FF70

OPERATOR ALAMAT

- `Andy = 25`
- `Fred = Andy;`
- `Ted = &Andy;`

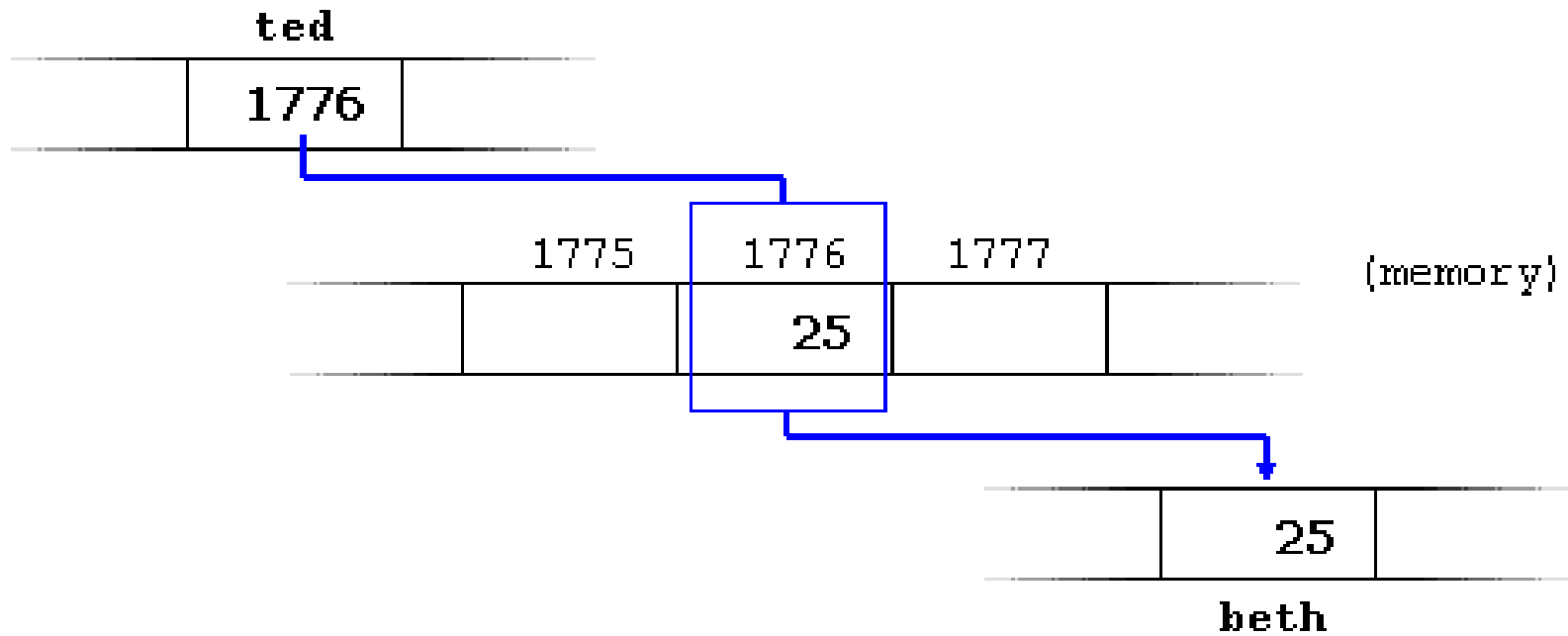
Andy akan memberikan variable ted alamat dari variable andy, karena variable andy diberi awalan karakter ampersand (&), maka yang menjadi pokok disini adalah alamat dalam memory, bukan isi variable



OPRATOR 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

```
andy == 25  
&andy == 1776  
ted == 1776  
*ted == 25
```



Format deklarasi pointer

- `type * pointer_name;`
- Contoh :
- `main ()`
- `{`
- `int value1 = 5, value2 = 15;`
- `int * mypointer;`
- `mypointer = &value1;`
- `*mypointer = 10;`
- `mypointer = &value2;`
- `*mypointer = 20;`
- `cout << "value1==" << value1 << "/ value2==" << value2;`
- `getch();`
- `}`

Borland C++

File Edit Search View Project Script Tool Debug Options Window Help



```
#include <iostream>
#include <conio>
#include <stdio>
#include <iomanip>

int main ()
{ int value1 = 5, value2 = 15;
  int * mypointer;
  mypointer = &value1;
  *mypointer = 10;
  mypointer = &value2;
  *mypointer = 20;
  cout << "value1==" << value1 << "\nvalue2==" << value2;
  getch();
}
```

C:\BC5\BIN\arra...

value1==10
value2==20

Running

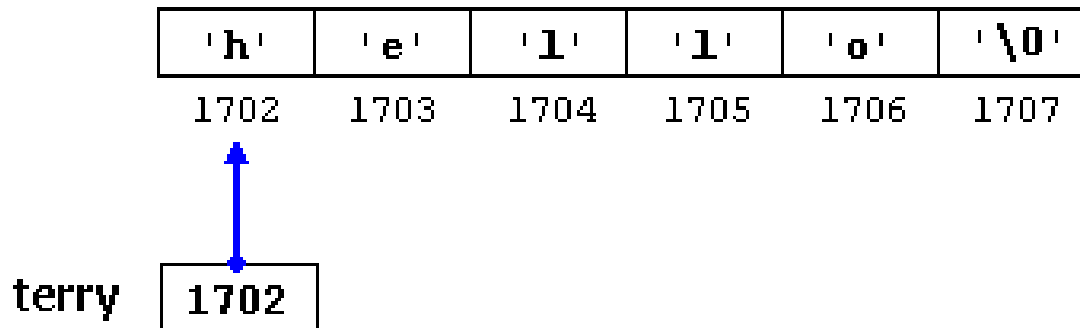
15:55:28

POINTER ARRAY

- Identifier suatu array equivalen dengan alamat dari elemen pertama, pointer equivalen dengan alamat elemen pertama yang ditunjuk.
- `int numbers [20];`
- `int *p;`
- Sama seperti :
- `p = numbers;`

Contoh

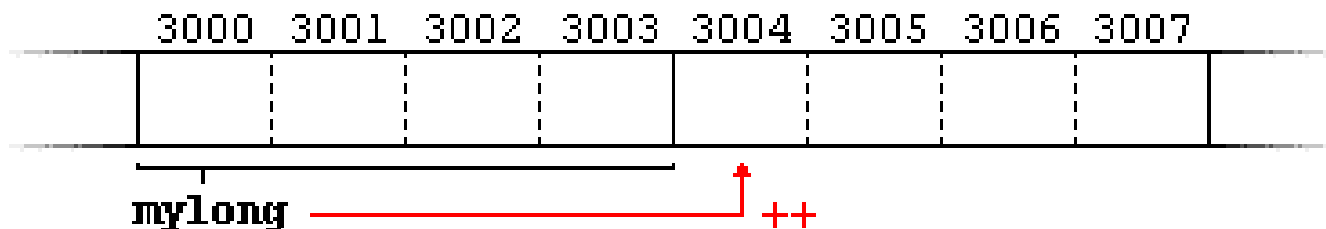
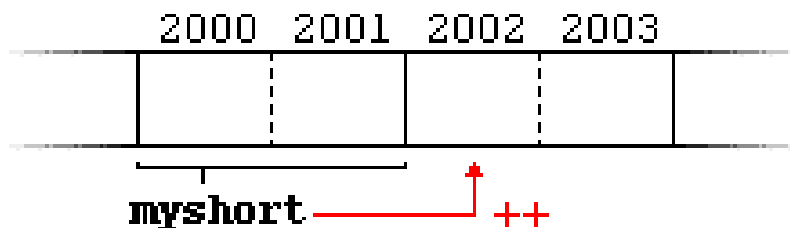
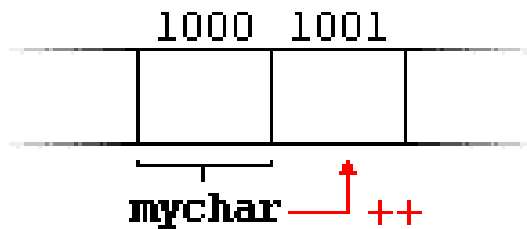
- Seperti pada array, inisialisasi isi dari pointer dapat dilakukan dengan deklarasi seperti contoh berikut :
- `char * terry = "hello";`
- Misalkan "hello" disimpan pada alamat 1702 dan seterusnya, maka deklarasi tadi dapat digambarkan sbb :



POINTER ARITMATIKA

- char memerlukan 1 byte, short memerlukan 2 bytes dan long memerlukan 4. Terdapat 3 buah pointer :
- `char *mychar;`
- `short *myshort;`
- `long *mylong;`
- ekspresi diatas akan menunjuk pada lokasi dimemory masing-masing 1000, 2000 and 3000, sehingga jika dituliskan :
- `mychar++;`
- `myshort++;`
- `mylong++;`

- mychar, akan bernilai 1001, myshort bernilai 2002, dan mylong bernilai 3004. Alasannya adalah ketika terjadi pertambahan maka akan ditambahkan dengan tipe yang sama seperti yang didefinisikan berupa ukuran dalam bytes.



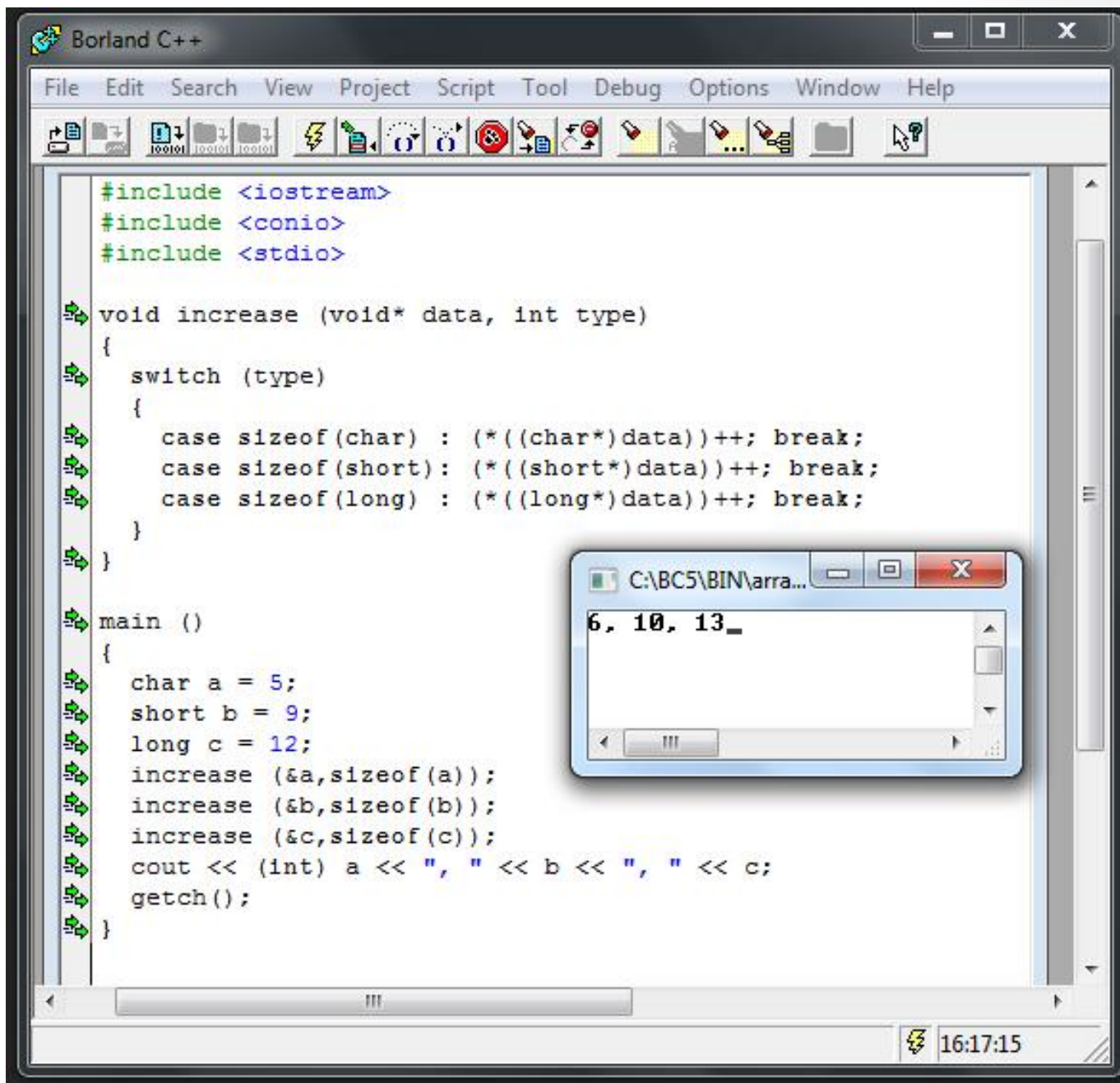
VOID POINTER

- void pointers dapat menunjuk pada tipe data apapun, nilai integer value atau float, maupun string atau karakter.
- Keterbatasannya adalah tidak dapat menggunakan operator asterisk (*), karena panjang pointer tidak diketahui, sehingga diperlukan operator type casting atau assignments untuk mengembalikan nilai void pointer ke tipe data sebenarnya

Contoh

```
• void increase (void* data, int type)
• {
•   switch (type)
•   {
•     case sizeof(char) : (*((char*)data))++; break;
•     case sizeof(short) : (*((short*)data))++; break;
•     case sizeof(long) : (*((long*)data))++; break;
•   }
• }

• imain ()
• {
•   char a = 5;
•   short b = 9;
•   long c = 12;
•   increase (&a,sizeof(a));
•   increase (&b,sizeof(b));
•   increase (&c,sizeof(c));
•   cout << (int) a << " , " << b << " , " << c;
•   getch();
• }
```



Contoh

- Pointer dideklarasikan dengan cara:
`type_data *nama_variabel_pointer;`

- Contoh inisialisasi pointer, contoh:

```
#include <stdio.h>
```

```
#include <conio.h>
```

```
int main(){  
    float nilai,*alamat=&nilai;  
    nilai = 25.2;  
    printf("nilai %.2f berada di alamat memori %p atau %x dalam hexa\n",nilai,alamat,alamat);  
    printf("isi nilai dari pointer alamat adalah %.2f",*alamat);  
    getch();  
}
```



The screenshot shows a Windows command prompt window with the title bar "F:\Documents and Settings\Administrator\My Documents\coba2.exe". The window contains the following output from the C program:

```
nilai 25.20 berada di alamat memori 0022FF74 atau 22ff74 dalam hexa  
isi nilai dari pointer alamat adalah 25.20
```

Aturan

- variabel pointer dapat dideklarasikan dengan **tipe data apapun**.
- Pendeklarasian variabel pointer dengan tipe data tertentu digunakan untuk menyimpan alamat memori yang berisi data sesuai dengan tipe data yang dideklarasikan, **bukan** untuk berisi nilai bertipe data tertentu.
- Tipe data digunakan sebagai lebar data untuk alokasi memori (misal char berarti lebar datanya 1 byte, dst)
 - jika suatu variabel pointer dideklarasikan bertipe float, berarti variabel pointer tersebut **hanya bisa** digunakan untuk menunjuk alamat memori yang berisi nilai bertipe float juga.

Contoh yang salah

```
#include <stdio.h>
#include <conio.h>

int main() {
    long int nilai = 9002;
    int *alamat_salah;
    alamat_salah = &nilai;
    printf("nilainya adalah %ld\n", *alamat_salah);
    return 0;
}
```

Message

Compiling NONAME00.CPP:

Error NONAME00.CPP 5: Cannot convert 'long __ss *' to 'int *' in function main()

Warning NONAME00.CPP 7: 'nilai' is assigned a value that is never used in function main()

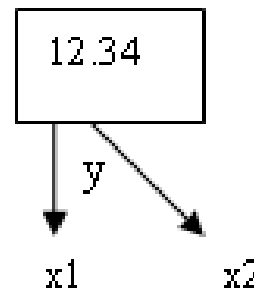
Line	File	Message
	F:\Documents and Settings\Administr...	In function 'int main()':
7	F:\Documents and Settings\Administr...	invalid conversion from 'long int*' to 'int'

Operasi pada Pointer

Operasi assignment

- Antar variabel pointer dapat dilakukan operasi assignment.
 - Contoh 1: Assignment dan sebuah alamat dapat ditunjuk oleh lebih dari satu pointer
 - Contoh 2: Mengisi variabel dengan nilai yang ditunjuk oleh sebuah variabel pointer
 - Contoh 3: Mengoperasikan isi variabel dengan menyebut alamatnya dengan pointer
 - Contoh 4: Mengisi dan mengganti variabel yang ditunjuk oleh pointer

Assignment, sebuah alamat dapat ditunjuk oleh lebih dari satu pointer



x1 dan x2 sama-sama menunjuk ke y

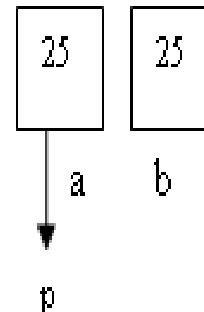
```
#include <stdio.h>
#include <conio.h>

int main() {
    float y, *x1, *x2;
    y = 12.34;
    x1 = &y;
    x2 = x1;    //operasi pemberian nilai
    printf("nilai y yang ditunjuk oleh x1 adalah %2.2f di alamat %p\n", y, &y);
    printf("nilai y yang ditunjuk oleh x2 adalah %2.2f di alamat %p\n", *x2, x2);
    getch();
}
```

F:\Documents and Settings\Administrator\My Documents\coba2.exe

```
nilai y yang ditunjuk oleh x1 adalah 12.34 di alamat 0022FF74
nilai y yang ditunjuk oleh x2 adalah 12.34 di alamat 0022FF74
```

Mengisi variabel dengan nilai yang ditunjuk oleh sebuah variabel pointer



p tetap menunjuk ke a bukan ke b

```
#include <stdio.h>
#include <conio.h>

int main() {
    int *p, a=25, b;
    p = &a;
    b = *p;
    printf("nilai a = %d di alamat %p\n", a, p);
    printf("nilai b = %d di alamat %p\n", b, p);
    getch();
}
```

F:\Documents and Settings\Administrator\My Documents\coba2.exe

```
nilai a = 25 di alamat 0022FF70
nilai b = 25 di alamat 0022FF70
```

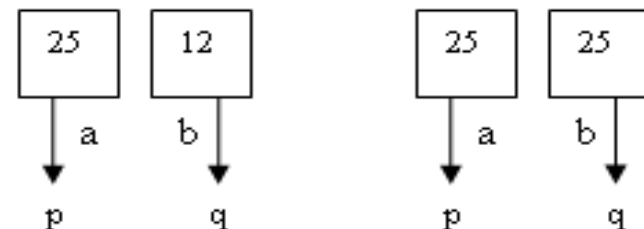
Mengoperasikan isi variabel dengan menyebut alamatnya dengan pointer

```
#include <stdio.h>
#include <conio.h>
```

```
int main(){
    int a=25,b=12;
    int *p,*q;
    p = &a;
    q = &b;
    printf("nilai yang ditunjuk p = %d di alamat %p\n", *p,p);
    printf("nilai yang ditunjuk q = %d di alamat %p\n", *q,q);
    *q = *p;
    printf("nilai yang ditunjuk p = %d di alamat %p\n", *p,p);
    printf("nilai yang ditunjuk q = %d di alamat %p\n", *q,q);
    getch();
}
```

F:\Documents and Settings\Administrator\My Documents\coba2.exe

```
nilai yang ditunjuk p = 25 di alamat 0022FF74
nilai yang ditunjuk q = 12 di alamat 0022FF70
nilai yang ditunjuk p = 25 di alamat 0022FF74
nilai yang ditunjuk q = 25 di alamat 0022FF70
```

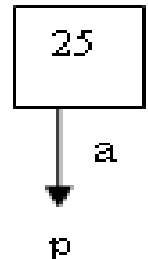
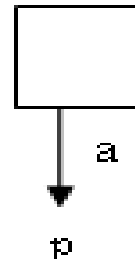


Mengisi dan mengganti variabel yang ditunjuk oleh pointer

C:\F:\Documents and Settings

nilai a = 25

```
#include <stdio.h>
#include <conio.h>
int main() {
    int a, *p;
    p=&a;
    *p=25;
    printf("nilai a = %d",a);
    getch();
}
```



Operasi pada Pointer (2)

Operasi aritmatika

- Pada pointer dapat dilakukan operasi aritmatika yang akan menunjuk suatu alamat memori baru.
- Hanya nilai **integer** saja yang bisa dioperasikan pada variabel pointer.
- Biasanya hanya operasi **penambahan/pengurangan** saja.
- Misal pointer X bertipe int (2 bytes), maka $X+1$ akan menunjuk pada alamat memori sekarang (mis. 1000) ditambah `sizeof(X)`, yaitu 2, jadi 1002.
- Lihat contoh

```

#include <stdio.h>
void main(){
    char S[] = "anton";
    char *p;
    //cara 1
    p=S;    //langsung menunjuk nama array.

    //cara 2
    //p=&S[0]; //sama, menunjuk alamat dari karakter pertama dari array

    for(int i=0;i<5;i++){
        printf("%c",*p);
        p++;
    }

    /*
    //coba ini, apa hasilnya?
    for(int i=0;i<5;i++){
        printf("%c",*p);
        p++;
    }

    //Bagaimana dengan ini?
    p--;
    for(int i=0;i<5;i++){
        printf("%c",*p);
        p--;
    }

    //Kalau ini?
    P=S;
    for(int i=0;i<5;i++){
        printf("%c",*p);
        p++;
    }
    */
}

```

Pointer pada Array

- Pada array, pointer hanya perlu menunjuk pada alamat **elemen pertama** saja karena letak alamat array sudah berurutan pada memori.
- Variabel pointer hanya perlu **increment**
- Lihat contoh-contoh!

Pada array 1D

C:\ F:\Documents and Settings\

```
p pertama : 1  
p berikutnya : 2
```

```
#include <stdio.h>  
#include <conio.h>  
int main() {  
    int a[5] = {1,2,3,4,5};  
    int *p;  
    p=a;  
    printf("p pertama : %d\n", *p);  
    p=a+1;  
    printf("p berikutnya : %d", *p);  
    getch();  
}
```

Pernyataan `p=a` artinya pointer `p` menyimpan alamat array `a`, yang alamatnya diwakili alamat elemen pertama, yaitu `a[0]`

Kita juga dapat menuliskan baris `p=a` diganti dengan `p=&a[0]`

```

#include <stdio.h>
#include <conio.h>

int main(){
    int a[5] = {1,2,3,4,5};
    int *p;
    p=a; //reset
    //tampilkan
    for(int i=0;i<5;i++){
        printf("%d ",*p);
        p++;
    }
    p=&a[0]; //reset
    //isi elemen
    for(int i=0;i<5;i++){
        *p = i*10;
        p++;
    }
    p=a; //reset
    //tampilkan
    for(int i=0;i<5;i++){
        printf("%d ",*p);
        p++;
    }
    getch();
}

```

64 F:\Documents and Settings

1 2 3 4 5

0 10 20 30 40