

P o i n t e r

Algoritma Pemrograman &
Struktur Data

Tujuan Pembelajaran

- Mahasiswa memahami tentang pointer pada C/C++ dan mampu menggunakannya
- Mahasiswa memahami dan mampu menggunakan pointer

Apakah pointer

- Pointer ?????
- Pointer adalah fundamental di C/C++
- Jika anda tidak mampu menggunakan pointer, maka anda kehilangan kemampuan untuk memanfaatkan kekuatan dan fleksibilitas yang disediakan C/C++.

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**.

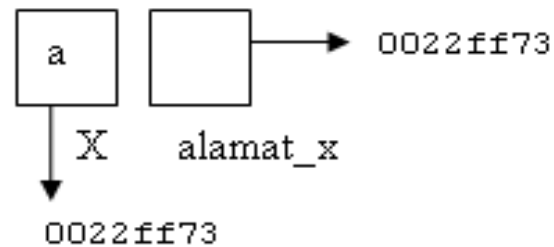
C/C++ and Pointer

- C/C++ banyak menggunakan pointer, karena :
 - Satu-satunya cara untuk mengekspresikan komputasi
 - Menghasilkan kode yang ringkas dan efisien
- Pointer digunakan dalam :
 - Array
 - Fungsi
 - Struktur

Deskripsi Pointer

- Pointer adalah variabel yang menyimpan alamat memori variabel lainnya.
- Operator ‘&’ digunakan untuk mendapatkan alamat dari variabel
- Operator ‘*’ digunakan untuk mendapatkan objek/nilai yang ditunjuk (*pointed to*) oleh pointer

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.

Mendeklarasikan Pointer

- Sama seperti variabel, pointer harus dideklarasikan sebelum digunakan.
- Contoh : **int *p;**
- Artinya : p adalah pointer to int
- int menyatakan tipe data yang ditunjuk oleh **p**.

- Saat pointer dideklarasikan, dia tidak menunjuk kemanaapun. Anda harus membuatnya menunjuk ke sesuatu sebelum digunakan. Contoh :

```
int *ip;  
*ip = 100; //akan error saat dijalankan
```

seharusnya,

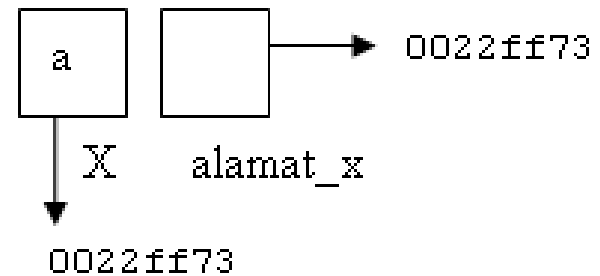
```
int *ip, i=100;  
*ip = &i;
```

- Format deklarasi pointer :

tipe_data *nama_var

- Pointer tidak dapat menunjuk :
 - Konstanta, contoh :
 - *p=3;
 - Variabel register, krn tidak berada di memori
 - Ekspresi, contoh :
 - *p= 3*7;

Contoh Program



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

```
int main(){
    char *alamat_x;
    char X;
    X = 'a';
    alamat_x = &X;
    printf("X, yaitu %c, disimpan pada alamat %p atau %x dalam hexa",X,alamat_x,alamat_x);
    getch();
}
```

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

X, yaitu a, disimpan pada alamat 0022FF73 atau 22ff73 dalam hexa

Format “%p” digunakan untuk menampilkan alamat pointer!

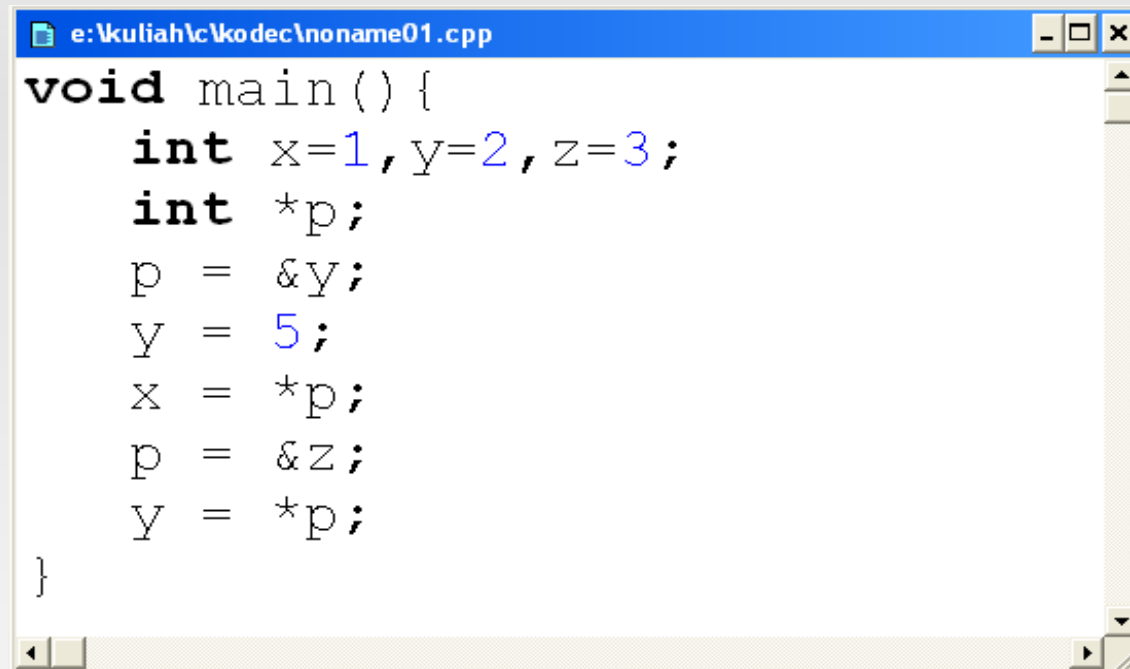
(Inactive C:\TCWIN45\BIN\WONAME00.EXE)

Nilai variabel X, yaitu a, disimpan pada alamat 2527:220D

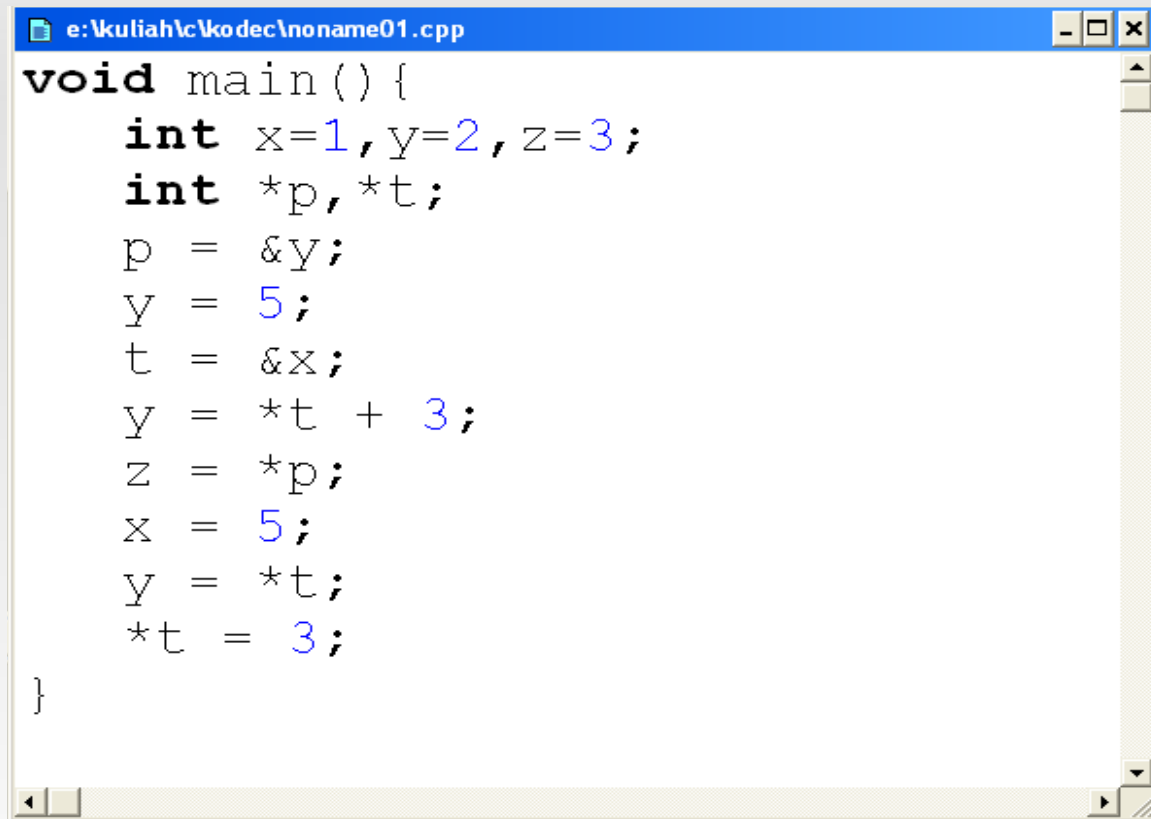
Contoh Program

```
#include <stdio.h>
void main() {
    int i,j;
    int *p; /*p adalah pointer to int*/
    i=6;
    p=&i; //set p menjadi alamat dari i
    j=*p; //set j dengan nilai yang ditunjuk oleh p
    *p=5; //set nilai yang ditunjuk oleh p menjadi 5
}
```

```
#include <stdio.h>
void main() {
    char *alamatX, X, Y, Z;
    X = 'B';
    alamatX = &X;
    Y = X;
    Z = *alamatX;
    //berapakah nilai X?
    //berapakah nilai Y?
    //berapakah nilai Z?
    //X berada di alamat?
}
```



```
e:\Kuliah\c\Kodec\noname01.cpp
void main() {
    int x=1, y=2, z=3;
    int *p;
    p = &y;
    y = 5;
    x = *p;
    p = &z;
    y = *p;
}
```



```
e:\kuliah\k\kodec\l\name01.cpp
void main() {
    int x=1,y=2,z=3;
    int *p,*t;
    p = &y;
    y = 5;
    t = &x;
    y = *t + 3;
    z = *p;
    x = 5;
    y = *t;
    *t = 3;
}
```

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>

Operasi pada Pointer

- Variabel pointer dapat dioperasikan sebagaimana variabel biasa, antara lain :
 - Operasi assignment
 - Operasi aritmatika
 - Operasi logika

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

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

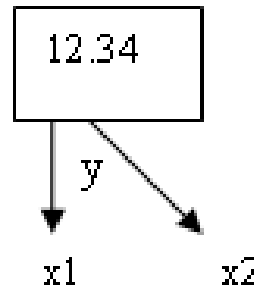
Operasi Assignment

```
#include <stdio.h>
void main() {
    float y, *x, *z;
    y=12.34;
    x=&y;
    z=x; //assignment antar var pointer
    //berapakah nilai x?
    //berapakah nilai y?
}
```

Assignment, sebuah alamat dapat ditunjuk oleh lebih dari satu pointer

```
#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();
}
```



x1 dan x2 sama-sama menunjuk ke y

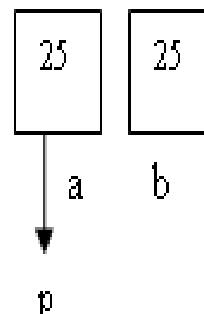
F:\Documents and Settings\Administrator\My Documents\coba2.exe

```
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

```
#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();
}
```



p tetap menunjuk ke a bukan ke b

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

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

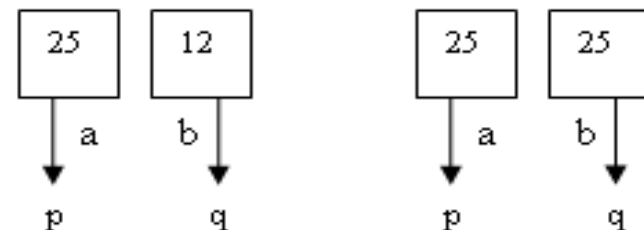
Mengoperasikan isi variabel dengan menyebutkan alamatnya dengan pointer

```
#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
```

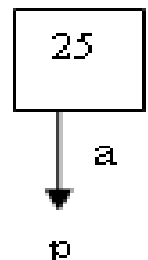
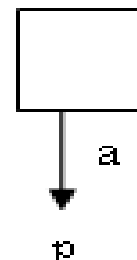


Mengisi dan mengganti variabel yang ditunjuk oleh pointer

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

C:\ F:\Documents and Settings

nilai a = 25



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

Operasi Aritmatika

- Variabel pointer dapat ditambah atau dikurangi dengan nilai integer.
- Operasi pertambahan dengan suatu nilai integer merupakan suatu peningkatan nilai pointer yang menunjukkan lokasi data berikutnya di memori

- Misalnya pointer X menyimpan alamat 1000, maka :
 - Jika x adalah int, maka p++ akan menunjuk alamat 1002 (krn int berukuran 2 byte)
 - Jika x adalah char, maka p++ akan menunjuk alamat 1001 (krn ukuran char adalah 1 byte)

```

#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++;
    }
    */
}

```

```
#include <stdio.h>
#include <stdlib.h>
void main() {
    int a[4]={10,20,30,40};
    int *pa;
    pa = a;
    printf("pa = %d\n",pa);
    printf("pa++ = %d\n",++pa);
}
```

```
#include <stdio.h>
#include <stdlib.h>
void main() {
    int a[4]={10,20,30,40};
    int *p1,*p2;
    p1 = &a[0];
    p2 = &a[2];
    printf("%d\n",p2-p1);
}
```

Operasi Logika

- Dua variabel pointer dapat dibandingkan jika keduanya mempunyai tipe yang sama atau keduanya bernilai null.

```
#include <stdio.h>
void main() {
    int bil1=100,bil2=100,*pb1,*pb2,*pb3;

    pb1 = &bil1;
    pb2 = &bil2;
    pb3 = pb1;
    if (pb1>pb2)
        puts("Alamat pb1 lebih rendah dari pb2.");
    else
        puts("Alamat pb1 lebih tinggi dari pb2.");

    if (pb1==pb3)
        puts("Alamat pb1 sama dengan pb3.");
    else
        puts("Alamat pb1 tidak sama dengan pb3.");
}
```


Operasi Pointer pd Array

- Bagaimana mengoperasikan array menggunakan pointer?
- Nama array yang ditulis tanpa indeks, menyatakan alamat elemen pertamanya.
- Gunakan operasi aritmatika pada pointer

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!

Mengakses Elemen Array

- Misalnya array X dan pointer P=X :
 - Alamat masing2 elemennya dpt dituliskan :
 - Elemen ke-1 : &X[0] atau X atau X+0 atau P atau P+0
 - Elemen ke-2 : &X[1] atau X+1 atau P atau P+1
 - Elemen ke-n : &X[n-1] atau X+(n-1) atau P+(n-1)
 - Isi array dpt diakses sbb :
 - Elemen ke-1 : X[0] atau *(X+0) atau *P atau *(P+0)
 - Elemen ke-2 : X[1] atau *(X+1) atau *(P+1)
 - Elemen ke-n : X[n-1] atau *(X+n-1) atau *(P+n-1)

Contoh-contoh

```
#include <stdio.h>
void main(){
    char s[]="INFORMATIKA";
    char *ps;

    ps = s;
    printf("Karakter ke-1 %c\n",*ps); //atau *(ps+0)
    printf("Karakter ke-2 %c", *(ps+1));
}
```

```
#include <stdio.h>
void main() {
    int data[4];
    int *pi;

    pi = data;
    *pi = 2;
    *(pi+1) = 6;
    *(pi+2) = 9;
    *(pi+3) = 11;
}
```

- Bagaimanakah menulis seluruh isi array menggunakan pointer?

Pada array 1D

g:\ 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();
}

```

GA F:\Documents and Settings

1 2 3 4 5

0 10 20 30 40

Perbedaan Array & Pointer

- Pointer adalah variabel, sehingga jika **pa** adalah pointer dan **a** adalah array **dapat** kita lakukan **pa = a** (yang identik dengan pernyataan **pa = &a[0]**) dan **pa++**
- Array bukanlah variabel sehingga **tidak dapat** kita lakukan **a = pa** dan **a++**

Array of Pointer

- Kita dapat membuat sebuah array of pointer oleh karena pointer juga adalah variabel.
- Contoh : menyimpan string yang berbeda panjangnya

Contoh : array of string

```
#include <stdio.h>
#include <stdlib.h>
void main() {
    char *s[5];
    s[0] = (char*) malloc(4 * sizeof(char));
    s[0] = "UMM";
    s[1] = (char*) malloc(12 * sizeof(char));
    s[1] = "informatika";
    puts(*s); //identik dg s[0]
    puts(*(s+1)); //identik dg s[1]
    free(s);
}
```

Pointer sbg Array

- Kita dapat membuat array yang dinamis ukurannya dengan alokasi
- Fungsi alokasi : malloc (ada di stdlib.h)

- Cara alokasi :

```
int *a;
```

```
a = (int *)malloc (5 *sizeof(int))
```

```
#include <stdio.h>
#include <stdlib.h>
void main() {
    int *a;
    a=(int*)malloc(10);
    *a = 1;
    *(a+1) = 2;
    *(a+2) = 3;
    *(a+3) = 4;
    printf("Element 0 %d", *(a+0));
    free(a);
}
```

- Array yang menyimpan N angka dari user
- Array yang menyimpan string dengan panjang dinamis

Array Multidimensi & Pointer

- Sebuah array 2 dimensi sebenarnya adalah array of array.
- Mendeklarasikan array 2 dimensi dg pointer :
tipe_data (*nama_array)[banyak_kolom]

Contoh :

int (*b)[4] → mendeklarasikan array b
dengan ukuran kolom 4

- Jika kita membuat array 2 dimensi :
int a[][20] identik dengan **int (*a)[20]**

```
#include <stdio.h>
void main() {
    int a[2][3]={{1,2,3},{4,5,6}};
    int *p;
    p=a[0];
    printf("%d",*(p+1));
}
```



```
#include <stdio.h>
#include <stdlib.h>
void main() {
    int **a;
    a = (int**) malloc(3*sizeof(*a));
    a[0] = (int*) malloc(2*sizeof(int));
    a[1] = (int*) malloc(2*sizeof(int));
    a[2] = (int*) malloc(2*sizeof(int));
    a[0][0] = 2009;
    a[0][1] = 1750;
    a[1][0] = 2008;
    a[1][1] = 1345;
    printf("%d : %d", *(*a), *((*a)+1));
    free(a);
}
```

Alokasi Dinamis

- Alokasi dinamis pada array memungkinkan kita untuk membuat array yang ukurannya disesuaikan dengan kebutuhan program.
- Pustaka : `stdlib.h`
- Fungsi : `malloc`, definisinya :
**`void *malloc(size_t
number_of_bytes)`**

- Contoh :

```
char *cp;
```

cp = malloc(100); → karena ukuran char adalah 1 byte, maka pernyataan tsb sama dengan mengalokasikan 100 elemen bertipe char untuk pointer **cp**.

- Beberapa kompiler meminta untuk adanya casting saat pemanggilan malloc.
- Contoh :
int *data;
data = (int*) malloc (100 * sizeof(int));
- Jika alokasi gagal, maka fungsi malloc akan mengembalikan **NULL**.
- Pastikan untuk mendealokasikan lagi memori yang telah selesai dg **free**

```
#include <stdio.h>
#include <stdlib.h>
void main() {
    int *p;
    p = (int*) malloc (sizeof (int));
    *p=100;
    free(p);
}
```

Latihan / Tugas

1. [Poin:5]Buatlah sebuah array bertipe char huruf A-J. Tampilkan seluruh elemennya menggunakan pointer.
2. [Poin:5]Buatlah array bertipe int yang berisi bilangan genap antara 0-10. Tampilkan seluruh elemennya menggunakan pointer.

3. [Poin:6]Buatlah program untuk membaca 5 angka dari user dan disimpan ke array. Gunakan pointer untuk menyimpan tiap nilainya.
4. [Poin:6]Buatlah program untuk membaca sebuah string dari user. Tampilkan per-karakter menggunakan pointer.

5. [Poin:6] Buatlah program yang akan menerima masukan berupa string dari user. Tanpa menggunakan fungsi `strlen`, hitunglah berapa panjang string tersebut dengan menggunakan pointer.
6. [Poin:6] Tanpa menggunakan fungsi `strcpy`, salinlah isi sebuah string ke variabel string lainnya (gunakan pointer).

7. [Poin:6] Dengan menggunakan pointer, buatlah program yang akan menggabungkan 2 buah string (tanpa menggunakan strcpy dan turunannya).
8. Dengan menggunakan pointer, buatlah array untuk menyimpan dan menampilkan data berikut :

Tahun	Jumlah
2009	1750
2008	1345
2007	950
2006	657