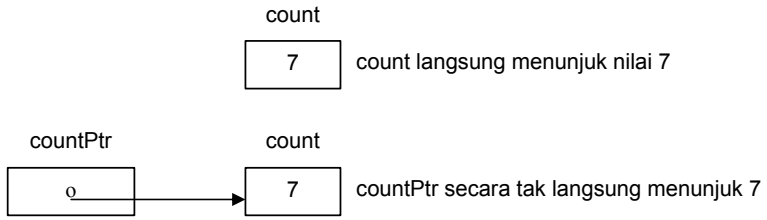
BAB VII
POINTER

Pointer adalah variabel yang berisi alamat memori sebagai nilainya, berbeda dengan variabel biasa yang berisi nilai tertentu. Dengan kata lain, pointer berisi alamat dari variabel yang mempunyai nilai tertentu. Dengan demikian, ada variabel yang secara langsung menunjuk ke suatu nilai tertentu, dan variabel yang secara tidak langsung (merupakan variabel pointer) menunjuk ke nilai. Hal ini dapat digambarkan melalui deklarasi berikut ini.

```
int *countPtr, count;
```

yang bila digambarkan dengan diagram menjadi:



OPERATOR POINTER

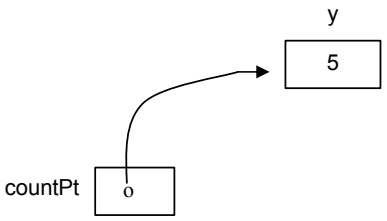
Operator alamat (dilambangkan sebagai **&**) adalah operator *unary* yang mengembalikan alamat dari operannya. Sebagai contoh, diasumsikan deklarasi sebagai berikut:

```
int y = 5;
int *yPtr;
```

maka pernyataan:

```
yPtr = &y;
```

menyatakan bahwa alamat dari variabel y ditujukan kepada variabel pointer yPtr, atau secara diagram dapat digambarkan sebagai berikut:



POINTER DAN ARRAY

Array dan pointer adalah dua struktur data yang saling berkaitan satu sama lain di dalam C, dan dapat saling dipertukarkan penggunaannya. Hal ini karena suatu array dapat didefinisikan sebagai pointer ke elemen pertama dari array tersebut. Misalnya didefinisikan dua variabel pointer dan array sebagai berikut:

```
int *bPtr, b[5];
bPtr  = b;
bPtr = &b[0];
```

yang berarti **bPtr** ditugaskan untuk menunjuk ke alamat elemen pertama dari array **b**, atau **b[0]**. Dengan demikian, elemen array b[3] dapat pula dituliskan dengan:

```
*(bPtr + 3)
```

dan alamat **&b[3]** dapat ditulis dengan:

```
bPtr + 3
```

Sedangkan deklarasi suatu variabel array **x[]** yang berisi nilai **int** dapat dituliskan sebagai:

```
iInt x[];       atau
int *x;

int y[][];      atau
int *y[];       atau
int *(*y);
```

Contoh berikut menunjukkan penggunaan pointer dan array untuk mendeklarasikan variabel array ganda untuk menyimpan empat buah elemen yang masing-masing bertipe string (array dari karakter):

```
Char *suit[4] = { "Hearts", "Diamonds", "Clubs", "Spades" };
```

LATIHAN 7

Untuk lebih memahami konsep pointer dan juga kaitannya dengan array, telaah masing-masing program berikut dan perkirakan apa yang dilakukannya.

1.
```
#include <stdio.h>

void misteri1(char *);

main() {
    char string[] = "characters";
    printf("String sebelum proses adalah %s", string);
    misteri1(string);
    printf("String setelah proses adalah %s", string);
    return 0;
}

void misteri1(char *s) {
    while ( *s != '\0' ) {
        if ( *s >= 'a' && *s <= 'z' )
            *s -= 32;
        ++s;
    }
}
```

2.
```
#include <stdio.h>

void misteri2(const char *);

main() {
    char *string = "ilmu komputer";
    misteri2(string);
    putchar('\n');
    return 0;
}

void misteri2(const char *s) {
    for ( ; *s != '\0' ; s++ )
        putchar(*s);
}
```

3.
```
#include <stdio.h>

int misteri3(const char *);

main() {
    char string[80];
    printf("Ketik sebuah string : ");  scanf("%s", string);
    printf("%d\n", misteri3(string));
    return 0;
}
```



```
int misteri3(const char *s) {
    int x = 0;

    for ( ; *s != '\0' ; s++)
        ++x;

    return x;
}
```

4.
```
#include <stdio.h>
#define SIZE 4

main() {
    int a[][SIZE] = { {0, 1, 2, 3}, {4, 5, 6}, {7, 8, 9, 10} };
    int *p = &a[0][0];
    int *q = a[0];
    int *r = a[1];
    int *s = a[2];

    printf("%d\n", *(p + SIZE + 1) );
    printf("%d\n", p[SIZE + 1] );
    printf("%d\n", p[2 * SIZE + 1] );
    printf("%d\n", *(q + 2 * SIZE + 2) );
    printf("%d\n", *r );
    printf("%d\n", *(r - 2) );
    printf("%d\n", s[3] );
    return 0;
}
```

5.
```
#include <stdio.h>

char *st = "KOMPUTER\n";
main() {
    int counter = 0;

    while ( *st++ ) ++counter;

    printf("%d : %s\n", counter, st) ;

    return 0;
}
```