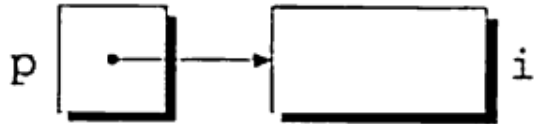


# İşaretçiler - Pointers

---

```
int *p;    p = &i;
```

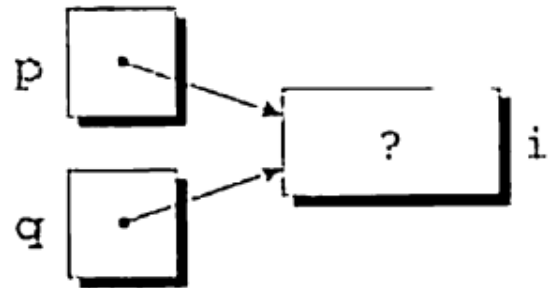


```
printf("%d\n", *p);
```

```
int i, j, *p, *q;
```

```
p = &i;
```

```
q = p;
```

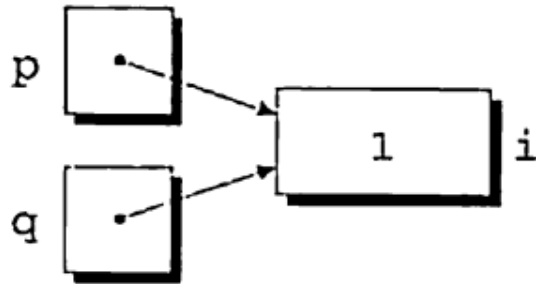


1/21

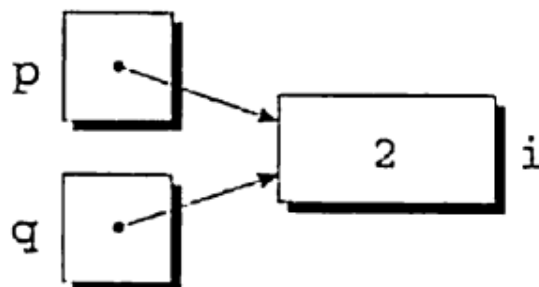
# İşaretçiler - Pointers

---

```
*p = 1;
```



```
*q = 2;
```



2/21

## İşaretçiler - Pointers

---

```
void decompose(double x, long *int_part, double *frac_part)
{
    *int_part = (long) x;
    *frac_part = x - *int_part;
}
```

```
decompose(3.14159, &i, &d);
```

3/21

---

## İşaretçiler - Pointers

---

```
int *max(int *a, int *b)
{
    if (*a > *b)
        return a;
    else
        return b;
}
```

```
int *p, i, j;
...
p = max(&i, &j);
```

4/21

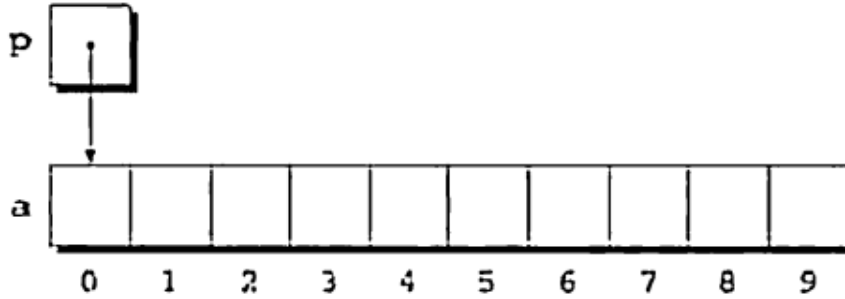
---

# İşaretçiler Diziler

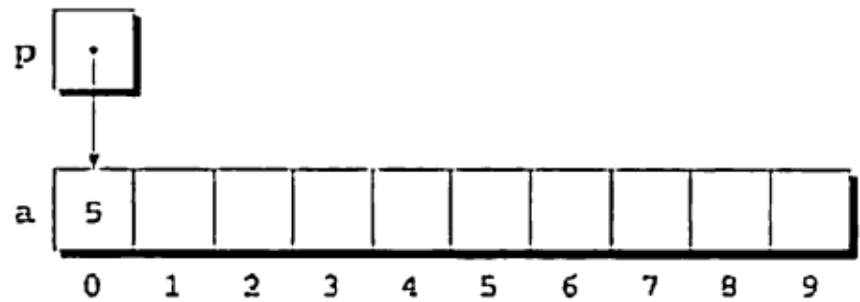
---

```
int a[10], *p;
```

```
p = &a[0];
```



```
*p = 5;
```

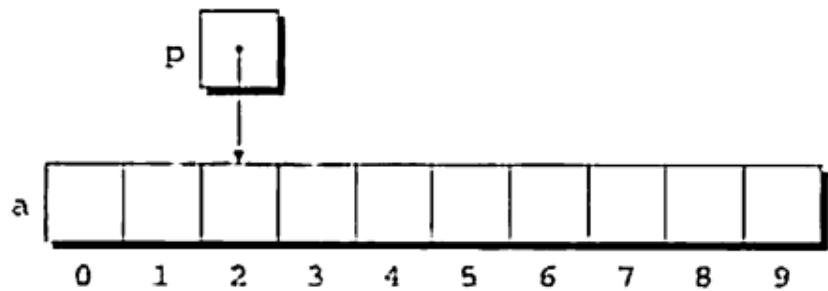


5/21

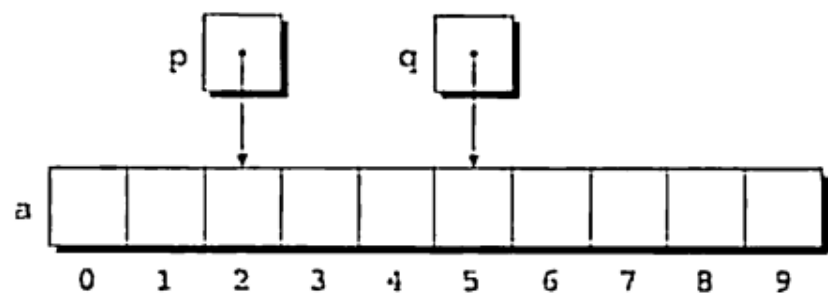
# İşaretçiler Diziler

---

```
p = &a[2];
```



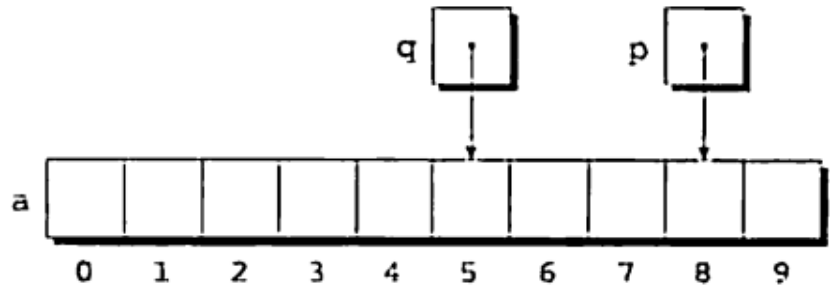
```
q = p + 3;
```



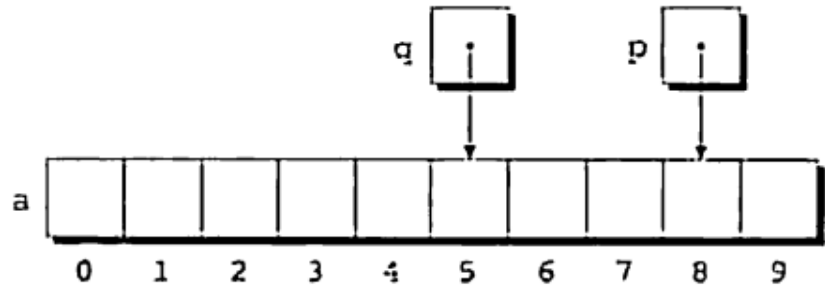
6/21

# İşaretçiler Diziler

```
p += 6;
```



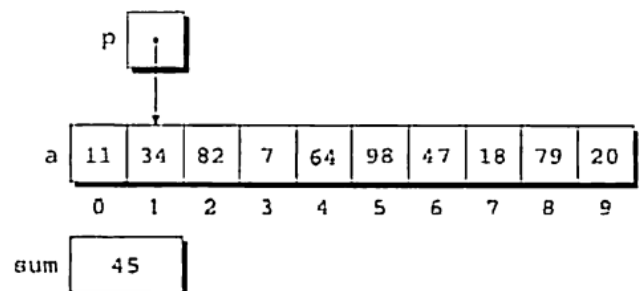
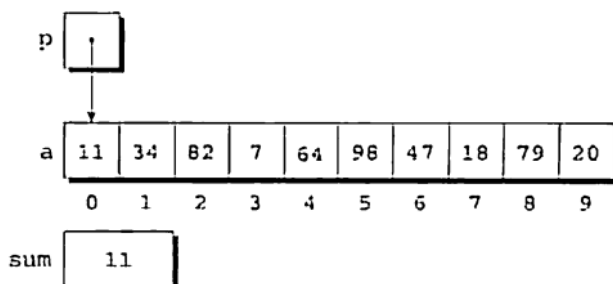
```
q = p - 3;
```



7/21

# İşaretçiler Diziler

```
#define N 10
...
int a[N], sum, *p;
...
sum = 0;
for (p = &a[0]; p < &a[N]; p++)
    sum += *p;
```

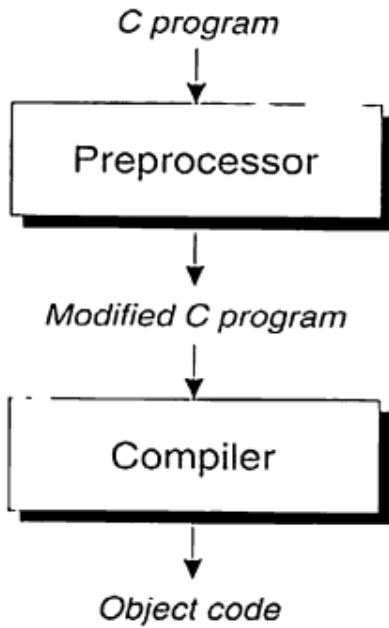


```
for (p = a; p < a + N; p++)
    sum += *p;
```

8/21

# Preprocessors

---



```
#define STR_LEN 80
#define TRUE 1
#define FALSE 0
#define PI 3.14159
```

```
#define BEGIN {
#define END }
```

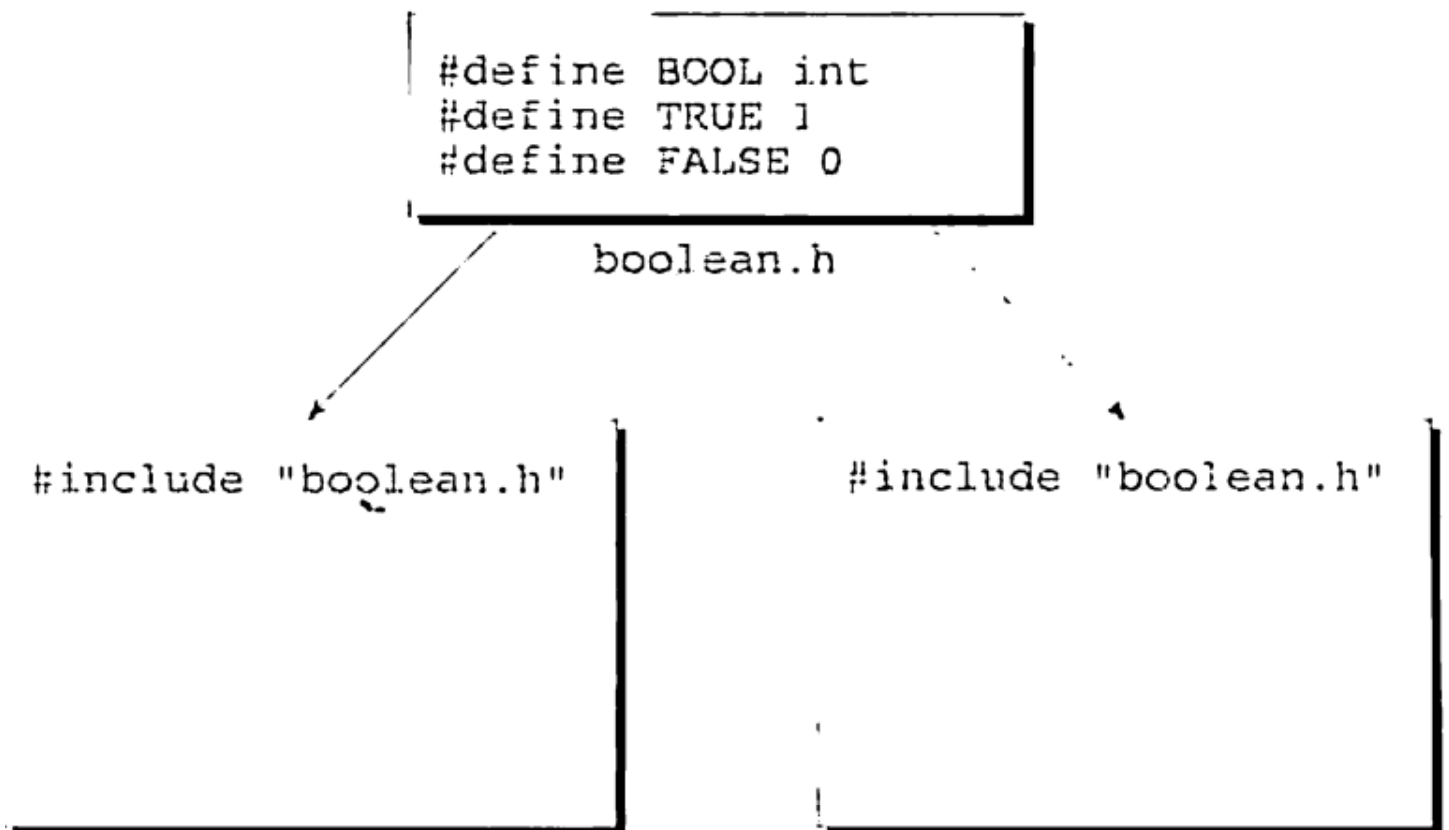
```
#define LOOP for (;;) 
```

9/21

---

# Preprocessors

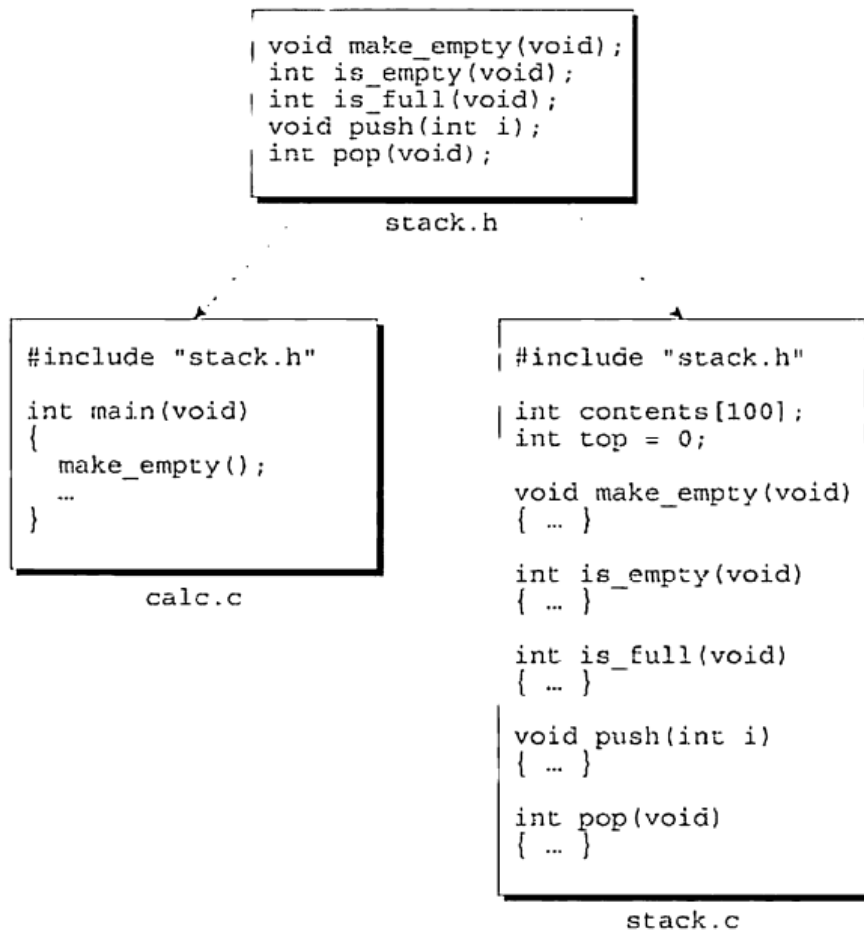
---



10/21

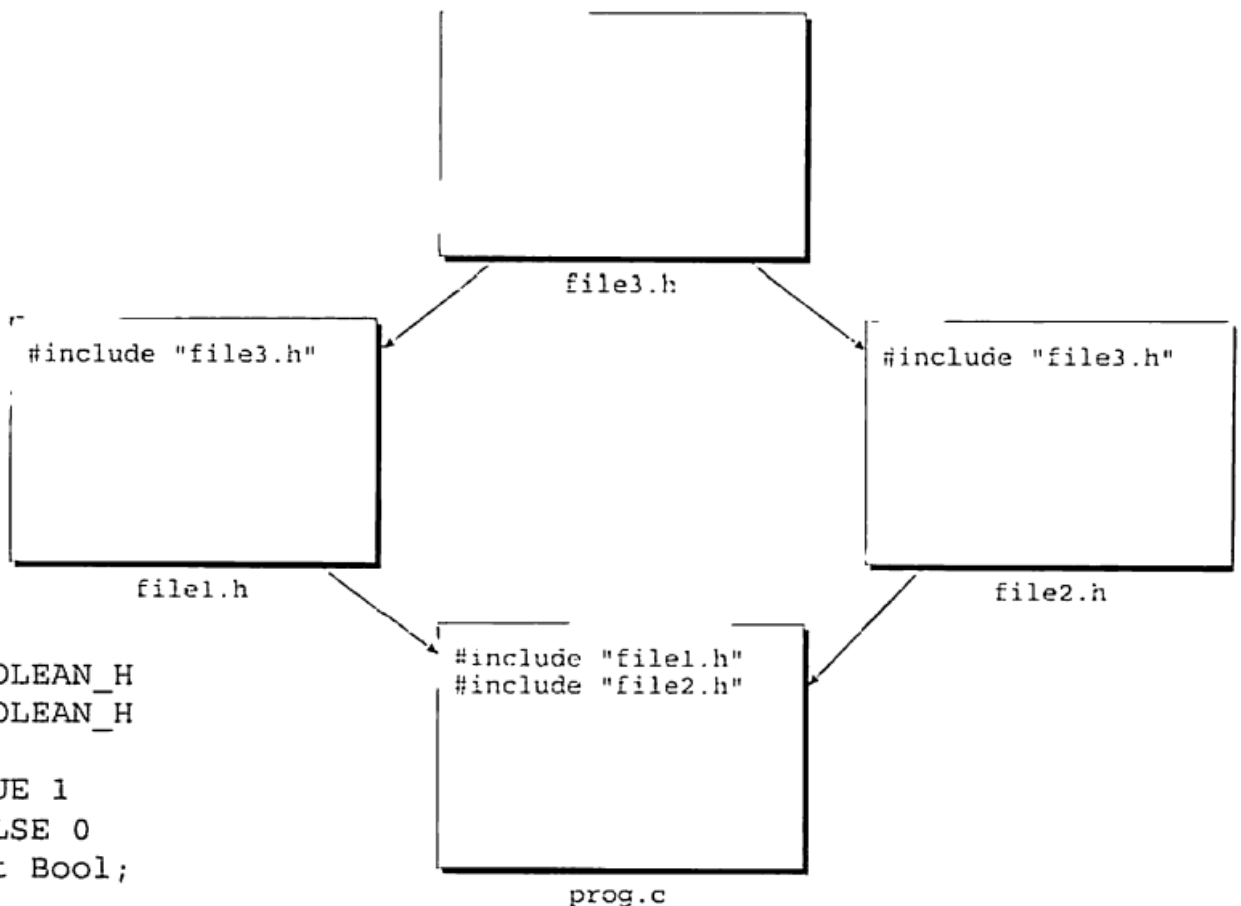
---

# Preprocessors



11/21

# Preprocessors



Örnek:

```
#ifndef BOOLEAN_H
#define BOOLEAN_H

#define TRUE 1
#define FALSE 0
typedef int Bool;

#endif
```

# Structures

---

```
struct {  
    int number;  
    char name[NAME_LEN+1];  
    int on_hand;  
} part1, part2;
```

```
struct {  
    int number;  
    char name[NAME_LEN+1];  
    int on_hand;  
} part1 = {528, "Disk drive", 10},  
   part2 = {914, "Printer cable", 5};
```

13/21

---

# Structures

---

```
struct {  
    int number;  
    char name[NAME_LEN+1];  
    int on_hand;  
} part1 = {528, "Disk drive", 10},  
   part2 = {914, "Printer cable", 5};
```

```
printf("Part number: %d\n", part1.number);  
printf("Part name: %s\n", part1.name);  
printf("Quantity on hand: %d\n", part1.on_hand);
```

14/21

---

# Structures

---

```
struct part {  
    int number;  
    char name[NAME_LEN+1];  
    int on_hand;  
};
```

```
struct part part1, part2;
```

```
part part1, part2;    /*** WRONG ***/
```

---

15/21

# Structures

---

```
typedef struct {  
    int number;  
    char name[NAME_LEN+1];  
    int on_hand;  
} Part;
```

```
Part part1, part2;
```

---

16/21



# Structures

---

```
void print_part(struct part p)
{
    printf("Part number: %d\n", p.number);
    printf("Part name: %s\n", p.name);
    printf("Quantity on hand: %d\n", p.on_hand);
}

print_part(part1);
```

17/21

---

# Enumerations

---

```
enum week{Mon, Tue, Wed, Thur, Fri, Sat, Sun};

int main()
{
    enum week day;
    day = Wed;
    printf("%d",day);
    return 0;
}
```

Output:

2

18/21

---

# Enumerations

---

```
#include <stdio.h>
enum day {sunday = 1, monday, tuesday = 5,
          wednesday, thursday = 10, friday, saturday};

int main()
{
    printf("%d %d %d %d %d %d %d", sunday, monday, tuesday,
          wednesday, thursday, friday, saturday);
    return 0;
}
```

Output:

```
1 2 5 6 10 11 12
```

19/21

---

# Enumerations

---

```
#include <stdio.h>
enum day {sunday = 1, tuesday, wednesday, thursday, friday, saturday};

int main()
{
    enum day d = thursday;
    printf("The day number stored in d is %d", d);
    return 0;
}
```

```
The day number stored in d is 4
```

20/21

---

```
struct node {  
    int value;  
    struct node *next;  
};  
  
struct node *new_node;  
  
new_node = malloc(sizeof(struct node));  
  
(*new_node).value = 10;  
new_node->value = 10;
```