# Control Statements

Structured Programming Language (CSE-1271)

Course Instructor : Mohammed Mamun Hossain
Assistant Professor, Dept. of CSE, BAUST

## Outline

1. Control Statements
2. if statements
3. if-else statements
4. Switch statements
5. Goto Statements
6. Conditional Operator (Statements)

## Control Statements

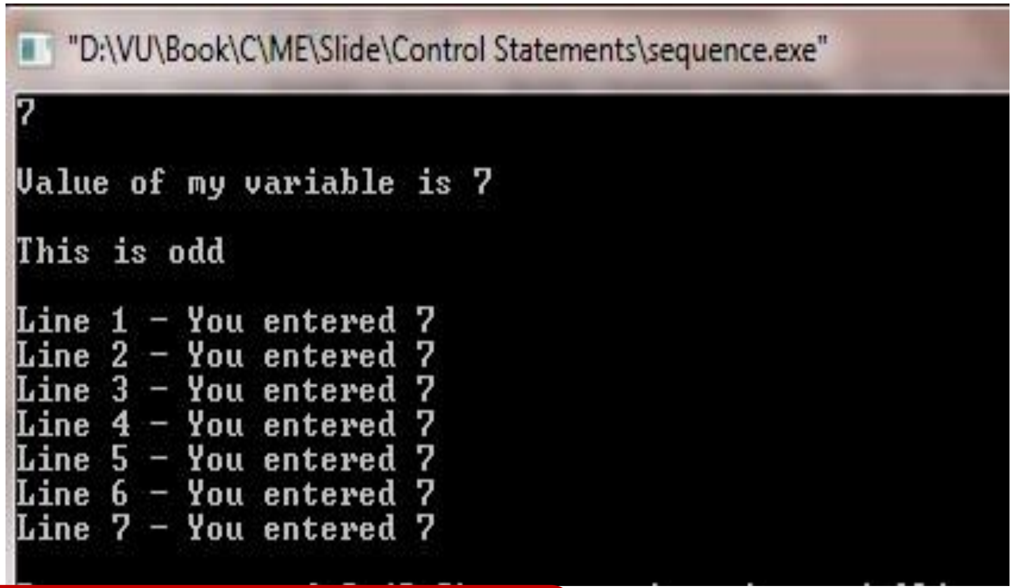
```c
#include<stdio.h>
int main()
{
    int i, myVariable;
    scanf("%d",&myVariable);
    printf("\nValue of my variable is %d\n\n",myVariable);

    if(myVariable%2==0)
    {
        printf("This is even\n\n");
    }
    else
    {
        printf("This is odd\n\n");
    }

    i=1;
    while(i<=myVariable)
    {
        printf("Line %d - You entered %d\n",i,myVariable);
        i=i+1;
    }

    return 0;
}
```

Selection

OR

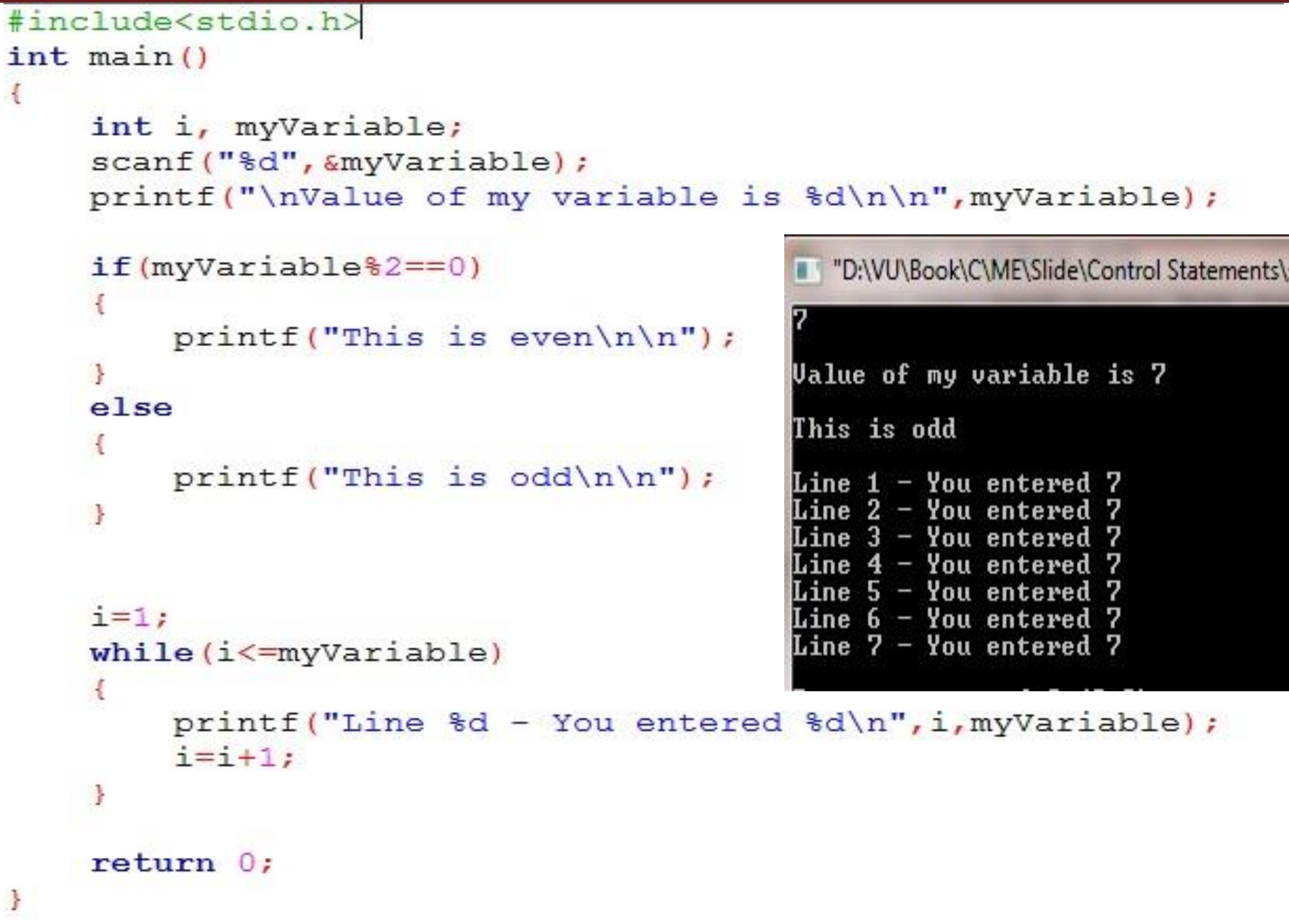
Repetition

Many time

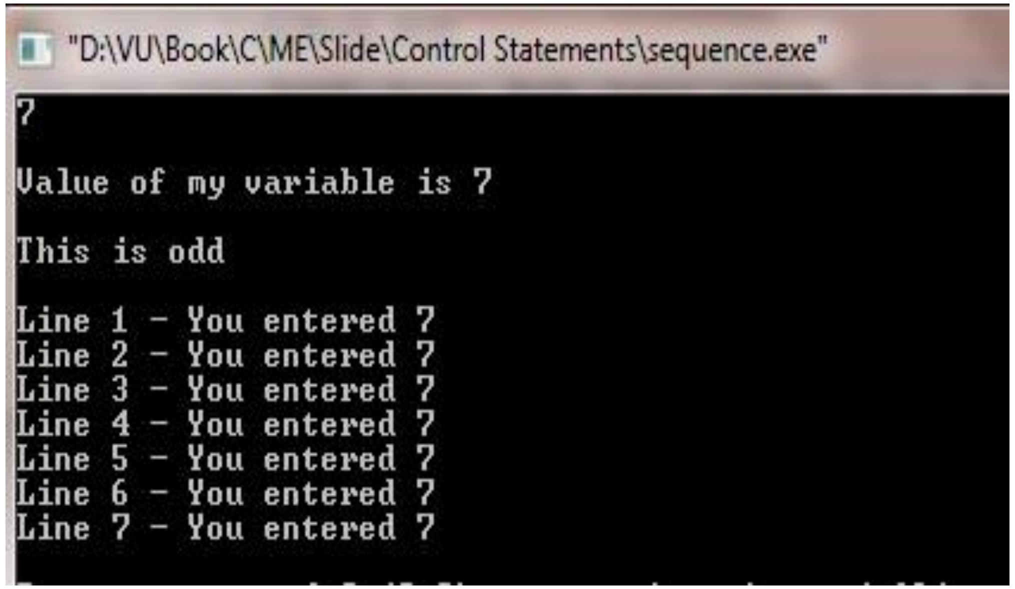


```
7

Value of my variable is 7

This is odd

Line 1 - You entered 7
Line 2 - You entered 7
Line 3 - You entered 7
Line 4 - You entered 7
Line 5 - You entered 7
Line 6 - You entered 7
Line 7 - You entered 7
```

# Control Statements



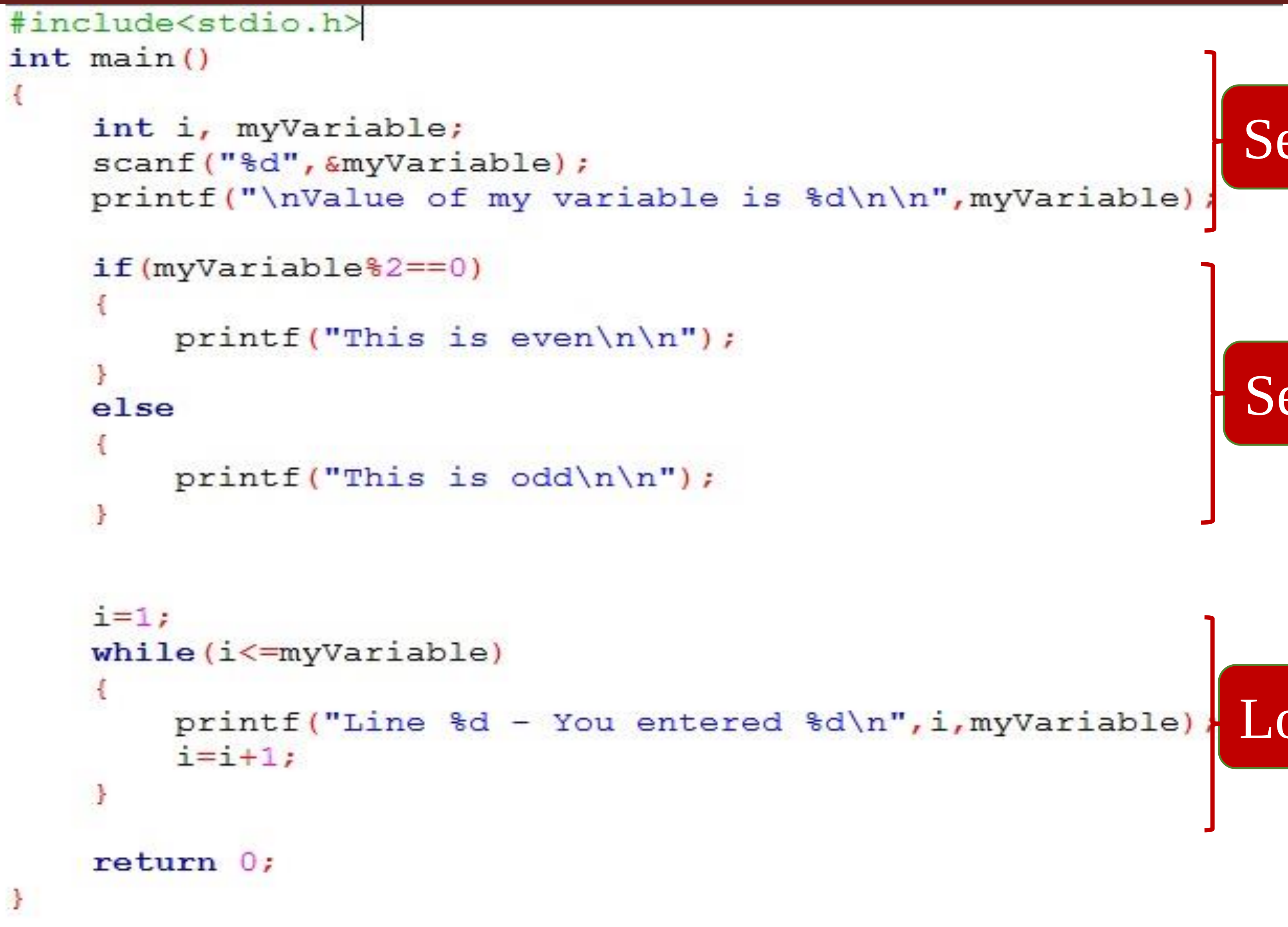
```c
#include<stdio.h>
int main()
{
    int i, myVariable;
    scanf("%d",&myVariable);
    printf("\nValue of my variable is %d\n\n",myVariable);

    if(myVariable%2==0)
    {
        printf("This is even\n\n");
    }
    else
    {
        printf("This is odd\n\n");
    }


    i=1;
    while(i<=myVariable)
    {
        printf("Line %d - You entered %d\n",i,myVariable);
        i=i+1;
    }

    return 0;
}
```



"D:\VU\Book\C\ME\Slide\Control Statements\sequence.exe"

```
7

Value of my variable is 7

This is odd

Line 1 - You entered 7
Line 2 - You entered 7
Line 3 - You entered 7
Line 4 - You entered 7
Line 5 - You entered 7
Line 6 - You entered 7
Line 7 - You entered 7
```

# Control Statements



```c
#include<stdio.h>
int main()
{
    int i, myVariable;
    scanf("%d",&myVariable);
    printf("\nValue of my variable is %d\n\n",myVariable);

    if(myVariable%2==0)
    {
        printf("This is even\n\n");
    }
    else
    {
        printf("This is odd\n\n");
    }


    i=1;
    while(i<=myVariable)
    {
        printf("Line %d - You entered %d\n",i,myVariable);
        i=i+1;
    }

    return 0;
}
```

Sequential statements

Selection statements

Loop statements

## Control Statements

Control the flow of execution in a program or function.

There are three kinds of execution flow:

- ❖ **Sequence**: The execution of the program is sequential.
- ❖ **Selection**: A control structure which chooses alternative to execute.
- ❖ **Repetition**: A control structure which repeats a group of statements.

## if Statements

- One of C’s **selection statement.**
- Sometimes called conditional statements
- Its operation is governed by the outcome of a conditional test evaluates to either true or false.
- In its simplest form, the if statement allows our program to conditionally execute a statement.

# if Statements

❖Simplest form of if statement:

```
if(expression)
        statement;
```

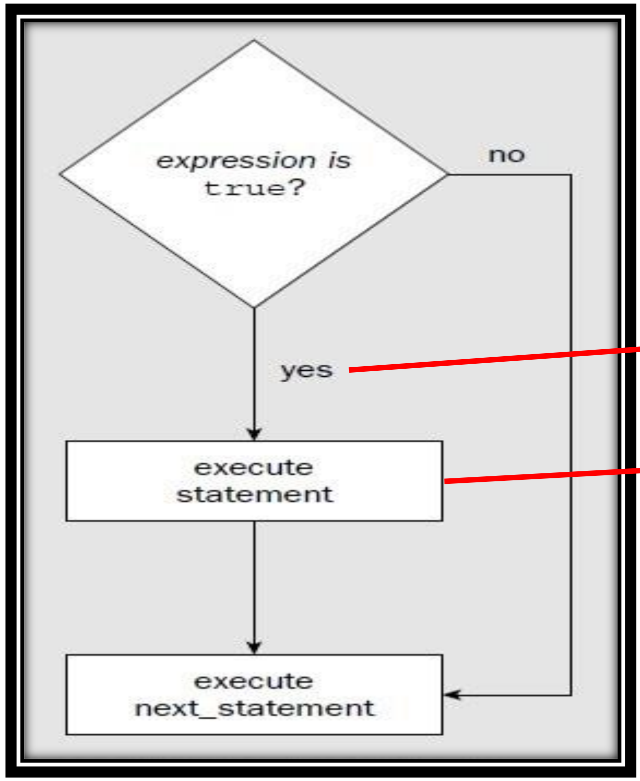
```
if(expression)
{
        statement;
}
```

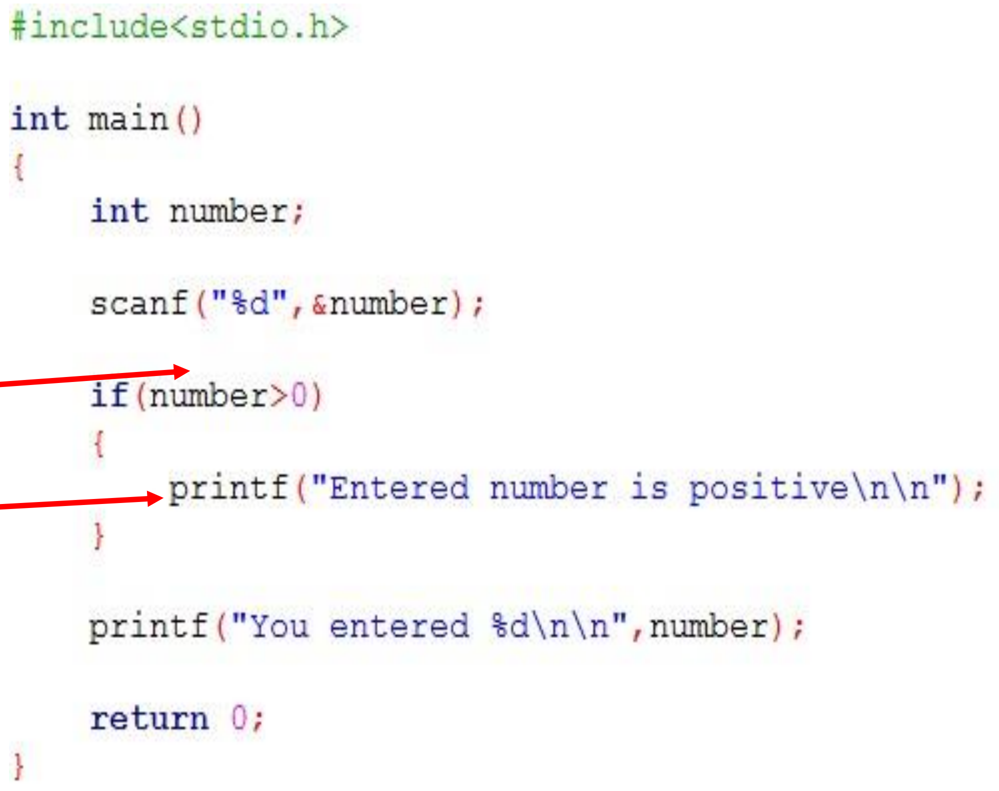
❖ For multiple statements:

```
if(expression)
{
     statement 1;
     statement 2;
     ……
     statement n;
}
```

# if Statements

- The expression may be any valid C expression.
- If the expression **evaluated as true**, the statement will be executed.
- If it **does not** - the statement is **bypassed** and the line of code following the if is executed.

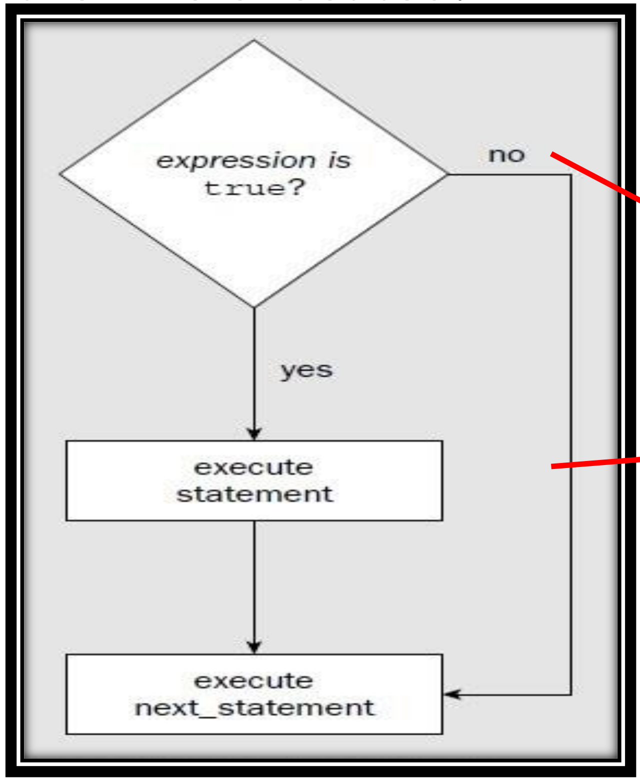


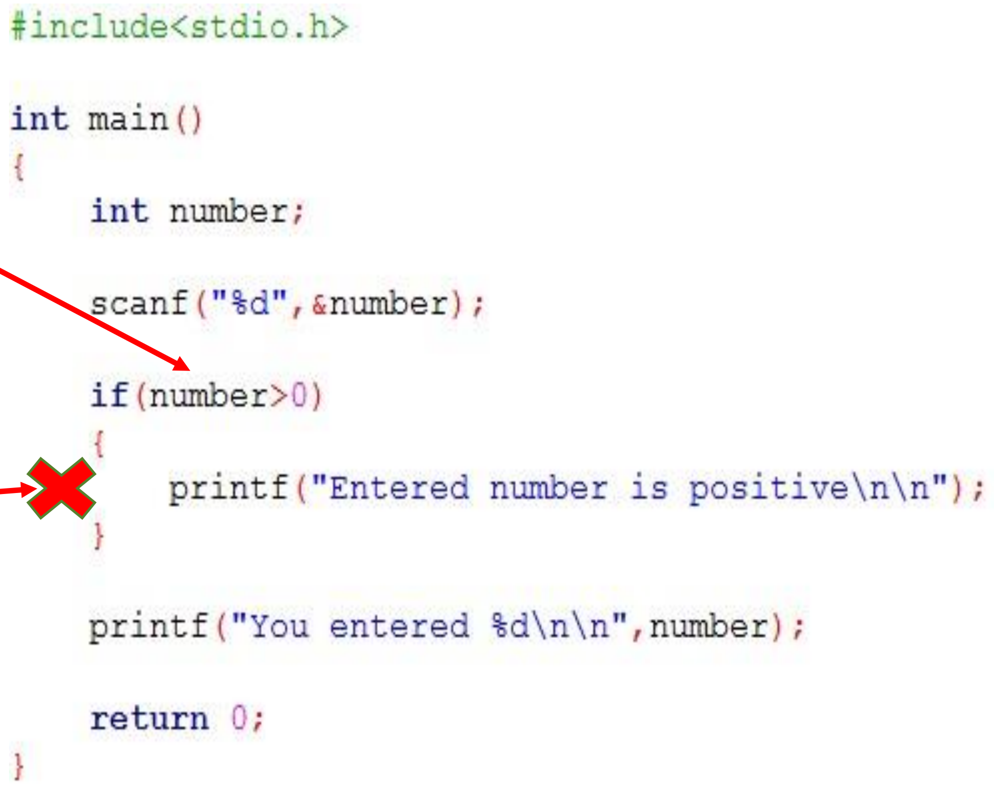


```
#include<stdio.h>

int main()
{
    int number;

    scanf("%d",&number);

    if(number>0)
    {
        printf("Entered number is positive\n\n");
    }

    printf("You entered %d\n\n",number);

    return 0;
}
```

# if Statements

- The expression may be any valid C expression.
- If the expression **evaluated as true**, the statement will be executed.
- If it **does not** - the statement is **bypassed** and the line of code following the if is executed.





```
#include<stdio.h>

int main()
{
    int number;

    scanf("%d",&number);

    if(number>0)
    {
        printf("Entered number is positive\n\n");
    }

    printf("You entered %d\n\n",number);

    return 0;
}
```

## if Statements

- In C, an expression is true if it evaluates to any nonzero values (5,9,-4,100 etc).

- If it evaluate to zero, it is false.

```
int a=10, b=20;
if(a<b)
  printf("This line will print.");
```

Output

This line will print.

```
int a=10, b=20;
if(a>b)
  printf("This line will print.");
```

Output

```
int a=10, b=20;
if(0)
  printf("This line will print.");
```

Output

## if Statements

```
int a=10, b=20;
printf("%d", a<=b);
```

```
Output

1
```

```
int a=10, b=20;
printf("%d", a>b);
```

```
Output

0
```

```
int a=10, b=20;
if(a)
 printf("This line will print.");
```

```
Output

This line will print.
```

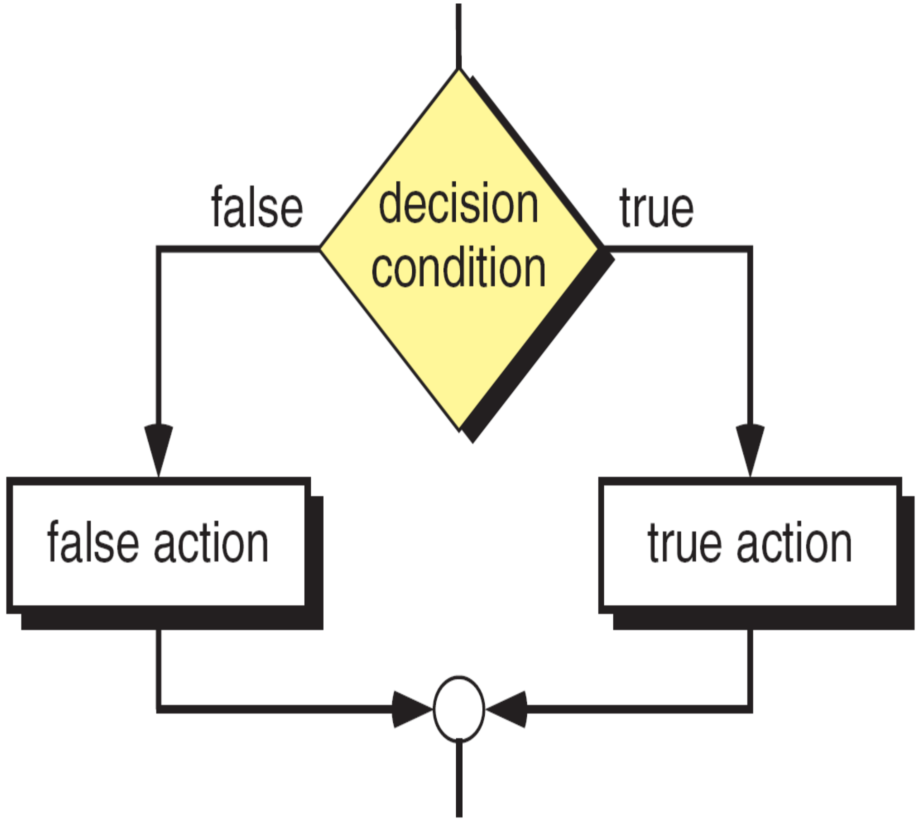
```
int a=-10, b=20;
if(a)
 printf("This line will print.");
```

```
Output

This line will print.
```

## If-else Statements

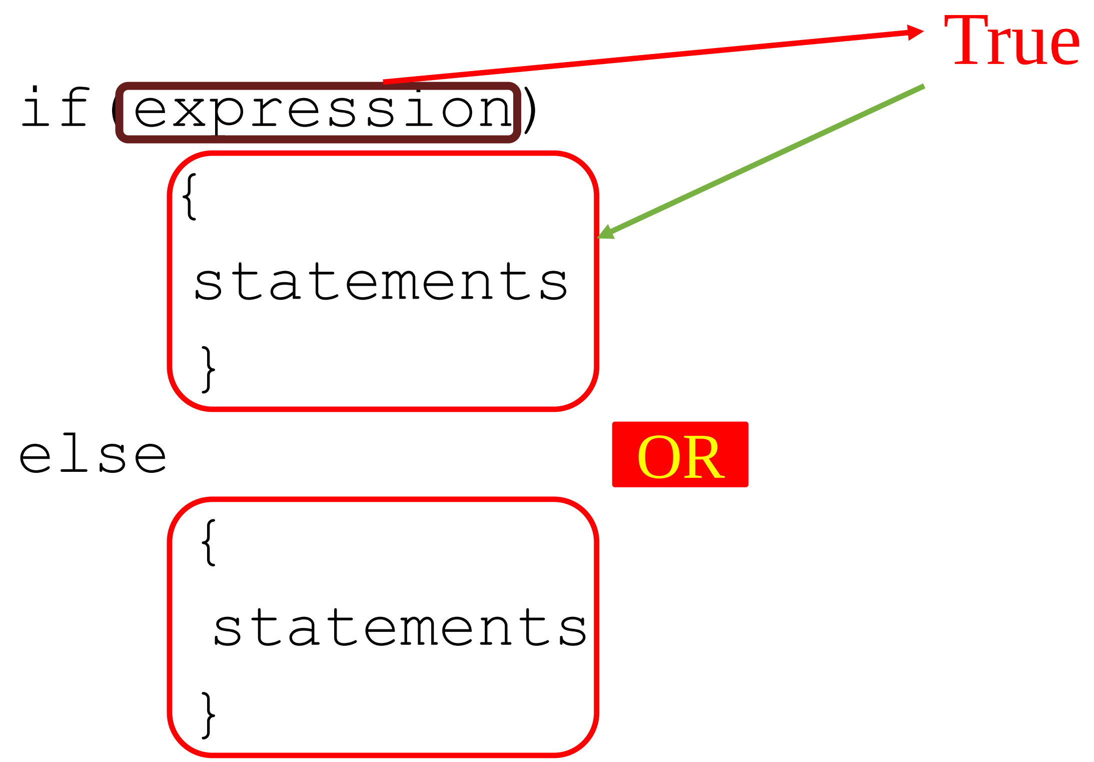
- We can add **else** statement to the if.
- Then the if statement looks like this:

```
if(expression)
     statement1;
else
     statement2;
```



## If-else Statements

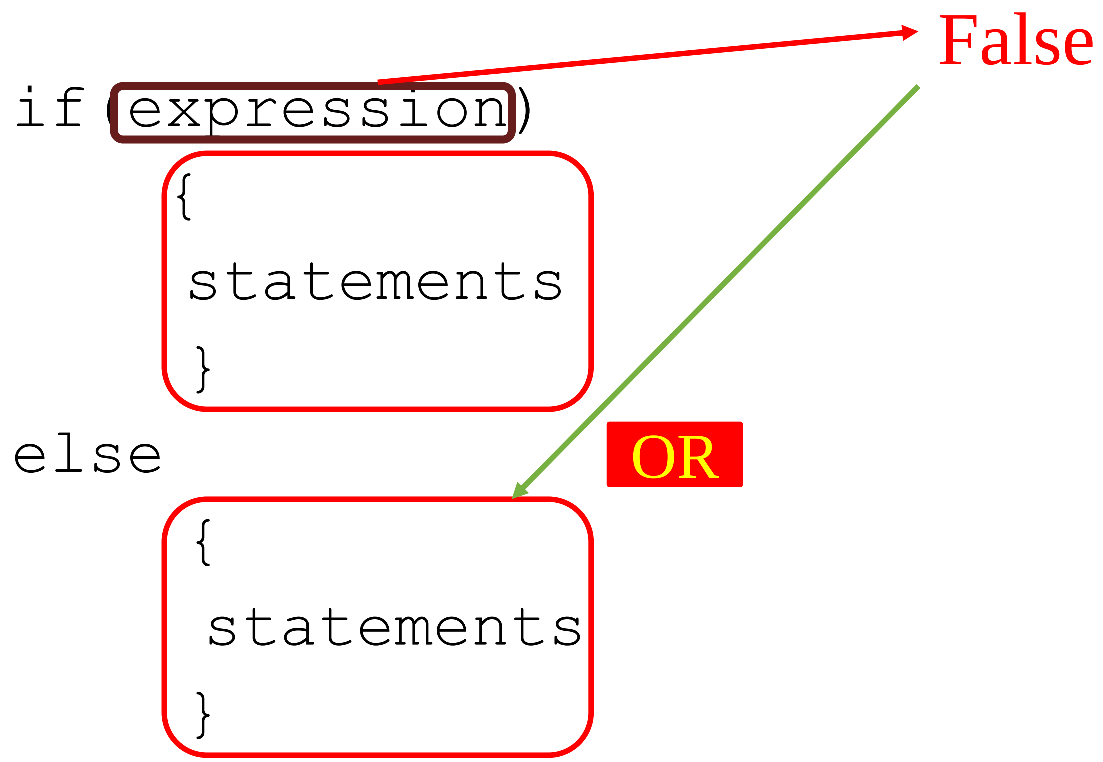
Structure of simple if-else is:



```
if (expression)
      {
       statements
      }
else
      {
       statements
      }
```

# If-else Statements

Structure of simple if-else is:



```
if (expression)
      {
       statements
      }
else
      {
       statements
      }
```

# If-else Statements

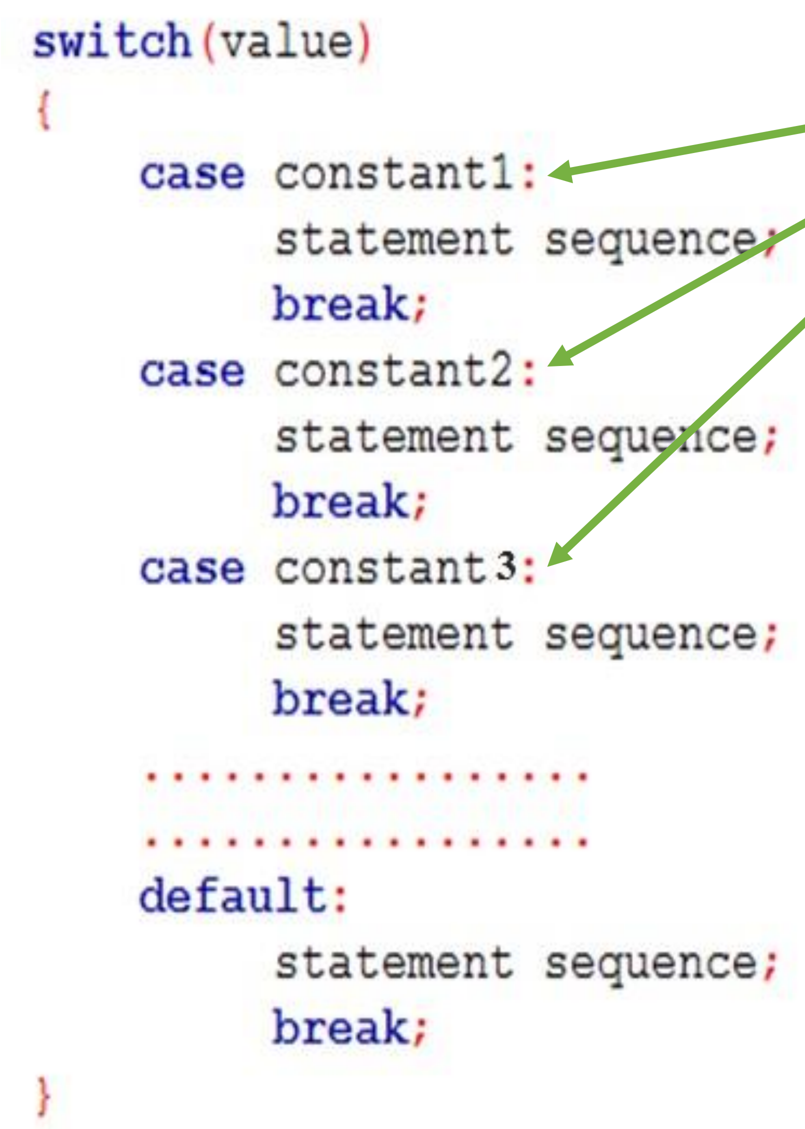
Structure of nested if-else is:

```
if(expression)
        {
         statements
         }
else if(expression)            OR
         {
          statements
         }
         ...
else                           OR
         {
         statements
         }
```

## Switch Statements

- **If** is good for choosing between two alternatives
- When several alternatives are needed we should use **switch** statement**.**
- **switch** is C’s multiple selection statement.
- Use to select one of several alternative paths in program execution

## Switch Statements



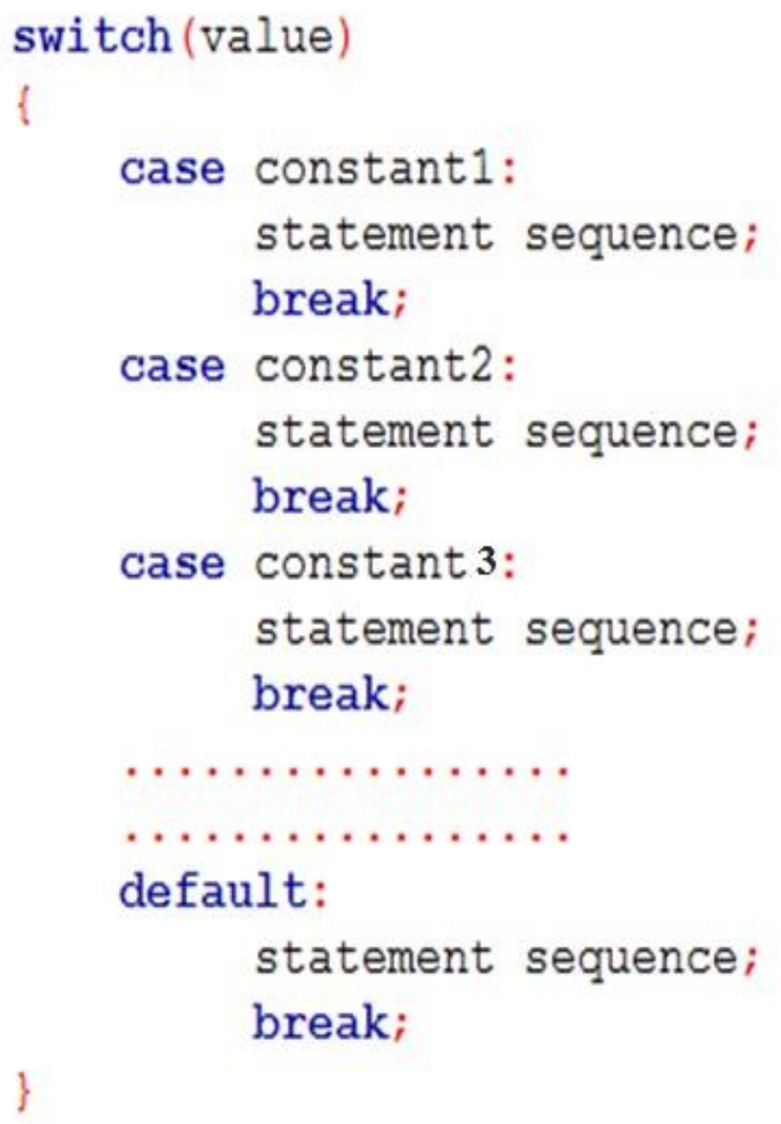
```
switch(value)
{
    case constant1:
        statement sequence;
        break;
    case constant2:
        statement sequence;
        break;
    case constant3:
        statement sequence;
        break;
    ..................
    ..................
    default:
        statement sequence;
        break;
}
```

A value is successively tested against a list of integer or character constants.

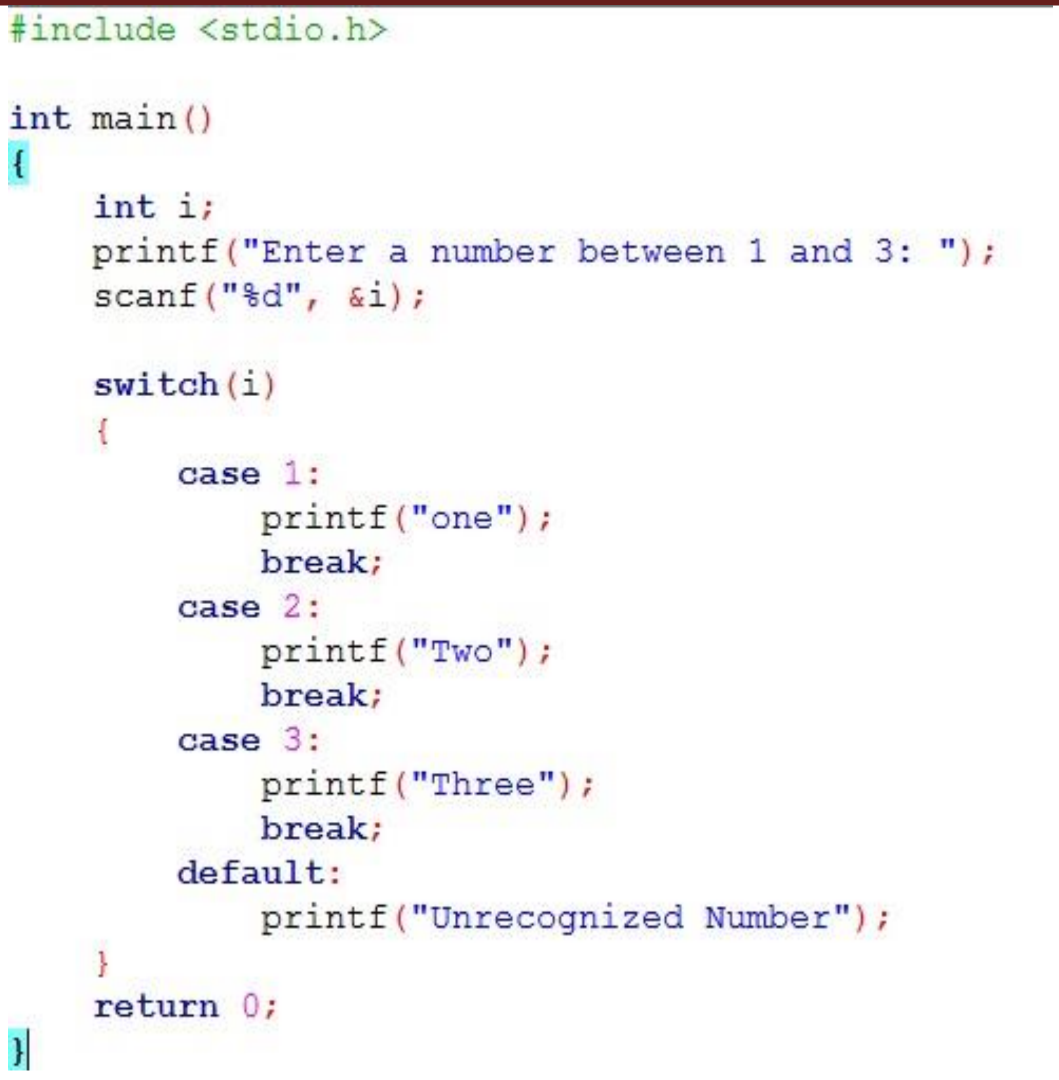
When the match is found, the statement sequence associated with that match is executed.

Statement sequence are not blocks, not use curly braces

# Switch Statements



```
switch(value)
{
    case constant1:
         statement sequence;
         break;
    case constant2:
         statement sequence;
         break;
    case constant3:
         statement sequence;
         break;
    .................
    .................
    default:
         statement sequence;
         break;
}
```



```
#include <stdio.h>

int main()
{
    int i;
    printf("Enter a number between 1 and 3: ");
    scanf("%d", &i);

    switch(i)
    {
        case 1:
            printf("one");
            break;
        case 2:
            printf("Two");
            break;
        case 3:
            printf("Three");
            break;
        default:
            printf("Unrecognized Number");
    }
    return 0;
}
```

## Switch Statements

**Nested switch:**

```
switch(i)
{
   case 1:
        switch(j)
        {
            case A:
                printf("Fist letter.");
                break;
            case B:
                printf("Second letter.");
        }
        break;
   case 2:

   .................
   .................
   default:
        statement sequence;
        break;
}
```

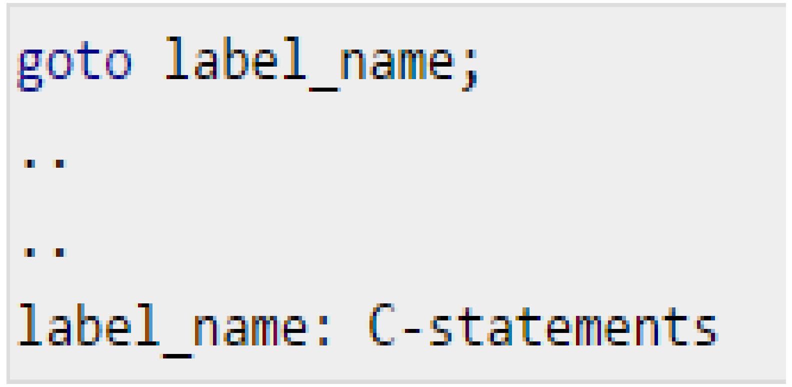
## If vs Switch

- **switch** can only test for equality, where the **if** conditional expression can be of any type
- **switch** will work with only **int** or **char** types. We can’t use **float** or others.

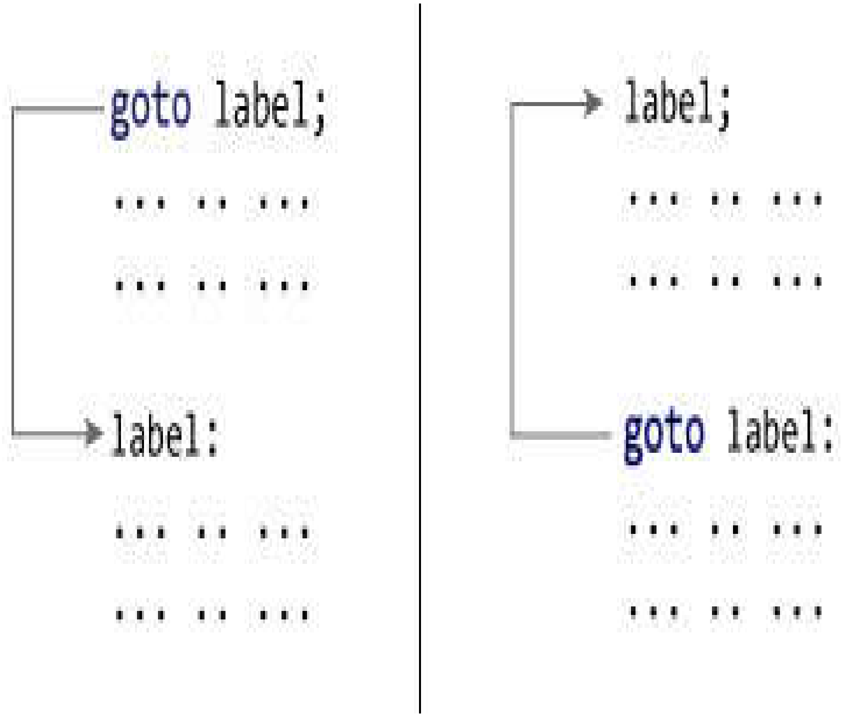
## Goto Statement

When compiler encounters goto statement in a C program, the control jumps to the corresponding label mentioned along with

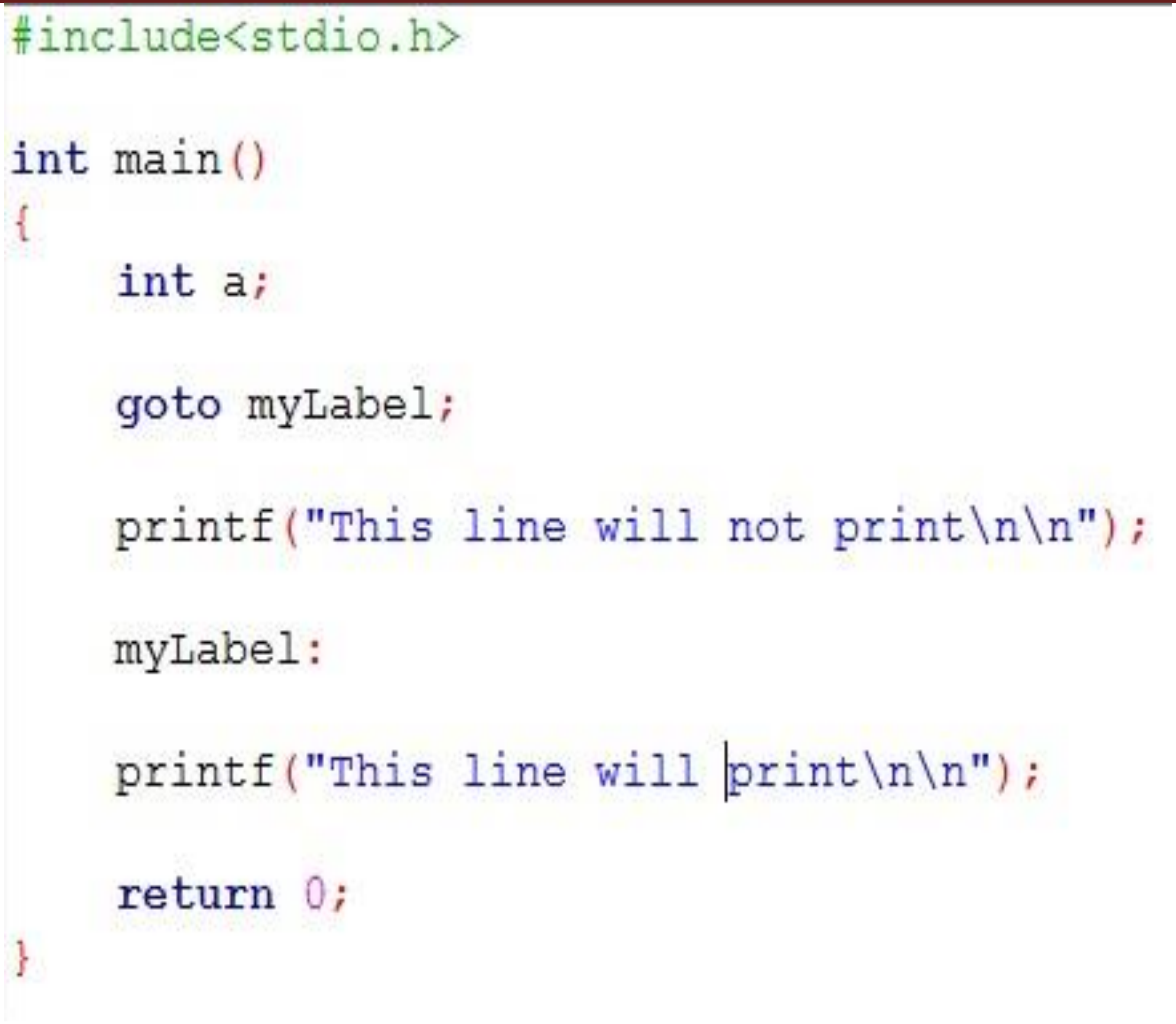
**Syntax of goto statement in C**



```
goto label_name;
..
..
label_name: C-statements
```



## Goto Statement



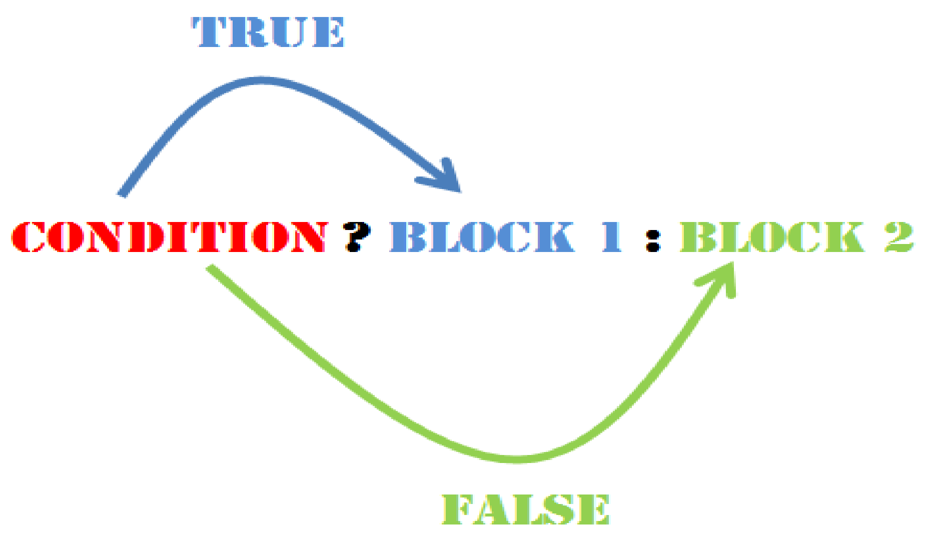
```
#include<stdio.h>

int main()
{
    int a;

    goto myLabel;

    printf("This line will not print\n\n");

    myLabel:

    printf("This line will print\n\n");

    return 0;
}
```

## Conditional Operator (Statements)

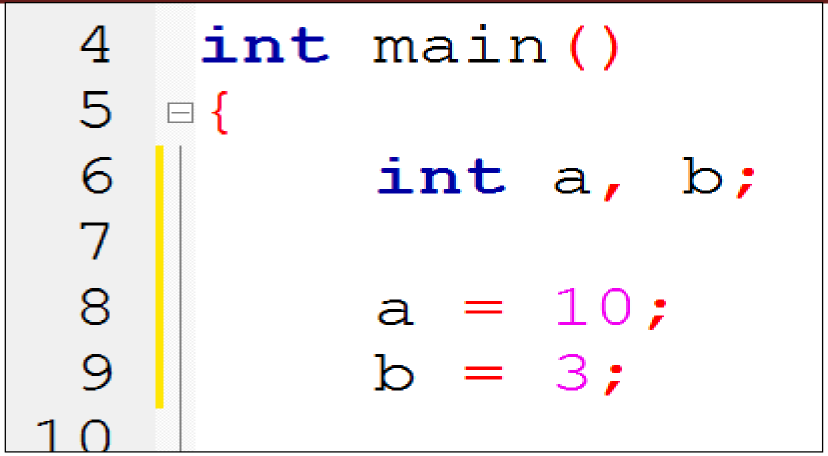
- The conditional operator `? :`
- A conditional expression is written in the form

expression 1 ? expression 2 : expression 3

| True or False? | True | False |
|---|---|---|



# Conditional Operator (Statements)



```
int main()
{
        int a, b;

        a = 10;
        b = 3;
```

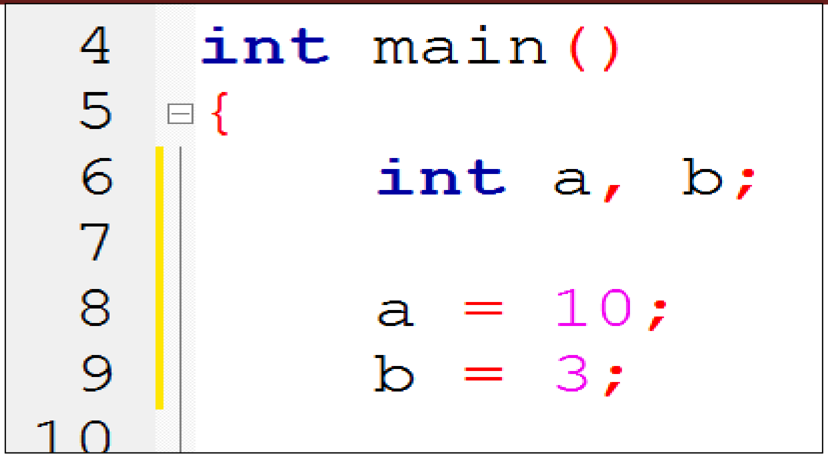
(a+b)>=13 ? a = 100 : a = 1000

Now a is 100

True or False?

True

# Conditional Operator (Statements)



```
int main()
{
        int a, b;

        a = 10;
        b = 3;
```

i = (a+b)<13 ? 100 : 1000

True or False?

False

Now i is 1000

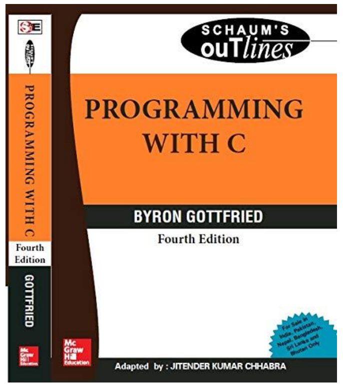
Thank You.

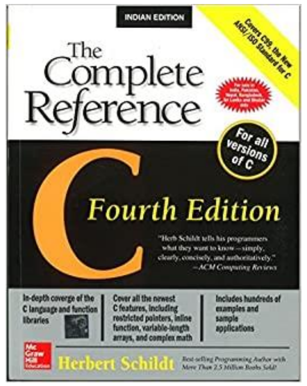
Questions and Answer

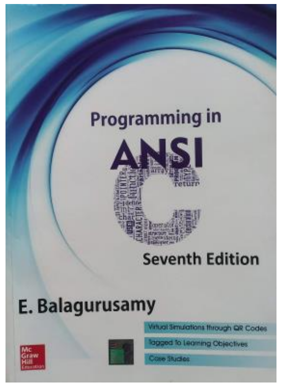
# References

Books:

1. Programming With C. *By Byron Gottfried*
2. The Complete Reference C. *By Herbert Shield*
3. Programming in ANSI C *By E. Balagurusamy*
4. Teach yourself C. *By Herbert Shield*







Web:

1. www.wikbooks.org

and other slide, books and web search.