Computer Programming & Problem Solving
CS100

Mrs Sanga G. Chaki

Department of Computer Science and Engineering

National Institute of Technology, Goa

November, 2022

What are Control Instructions in C

1. Enable us to specify the order in which the various instructions in a program are to be executed
   a) Sequence Control Instruction
   b) Selection or Decision Control Instruction
   c) Repetition or Loop Control Instruction
   d) Case Control Instruction

The Decision Control Structure – why needed?

1. Need to alter our actions due to changing circumstances
2. C allows us to implement this using the if statement
3. if (this condition is true)

    execute this statement ;

if statement

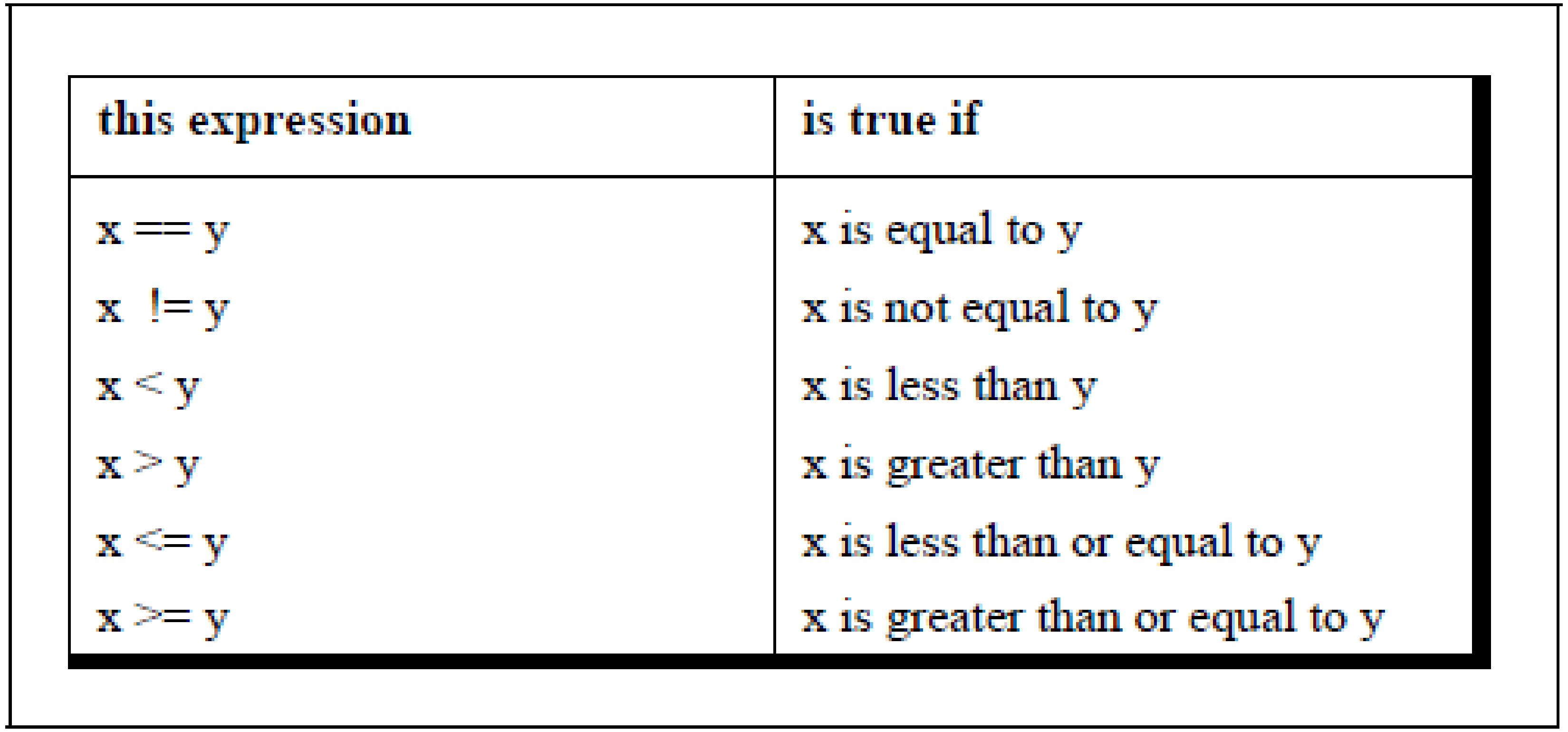
```
if ( this condition is true )
        execute this statement ;
```

1. keyword **if** tells the compiler that what follows is a decision control instruction.
2. The condition following the keyword if is always enclosed within a pair of parentheses.
3. If the condition is true, then the statement is executed.
4. If the condition is not true then the statement is not executed; instead the program skips past it.

Relational Operators

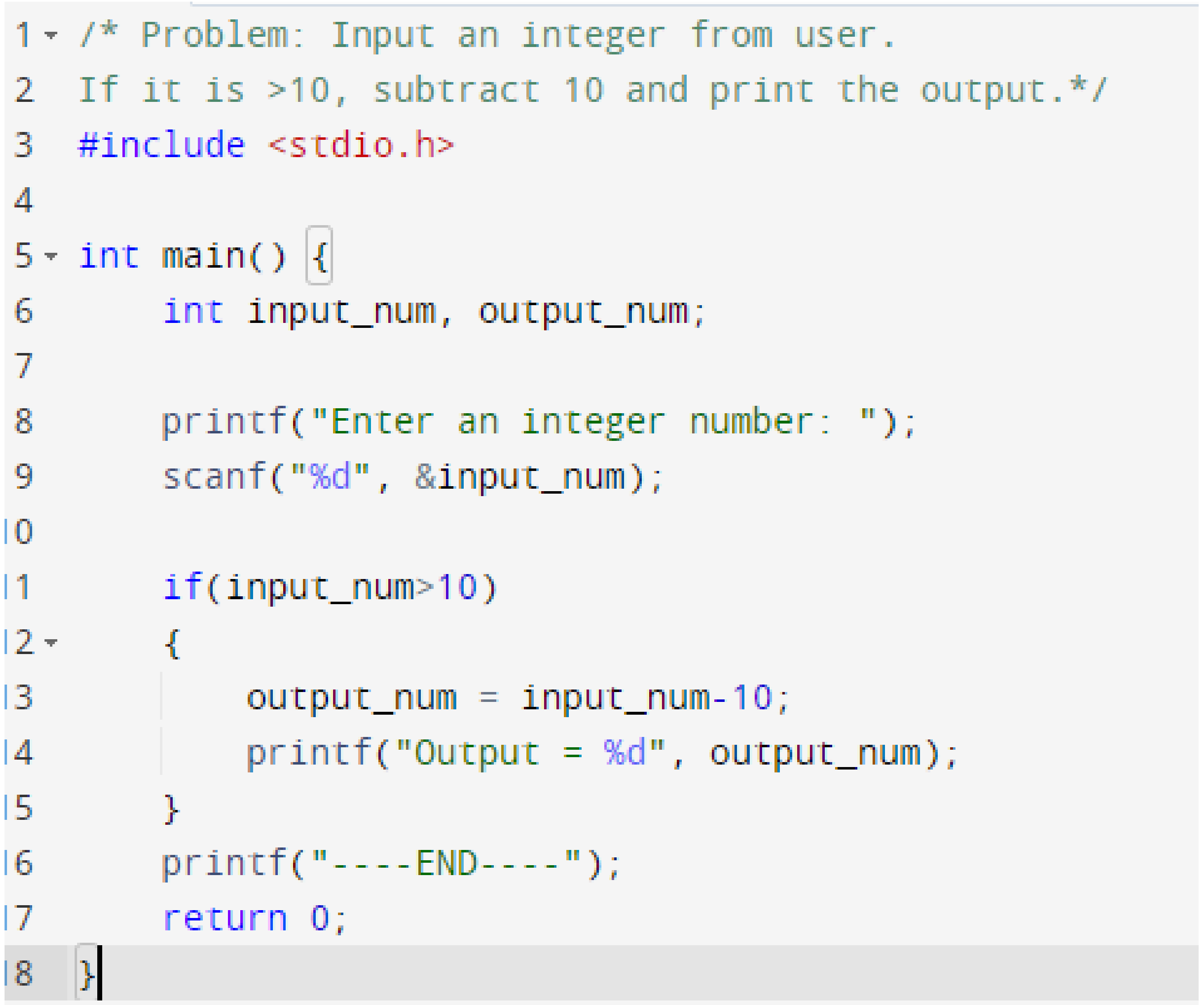
1. How do we express the condition in C? And how do we evaluate ?
2. Using C’s ‘relational’ operators.
3. The relational operators allow us to compare two values to see whether they are equal to each other, unequal, or whether one is greater than the other.

Relational Operators



this expression	is true if
x == y	x is equal to y
x != y	x is not equal to y
x < y	x is less than y
x > y	x is greater than y
x <= y	x is less than or equal to y
x >= y	x is greater than or equal to y

If statement example



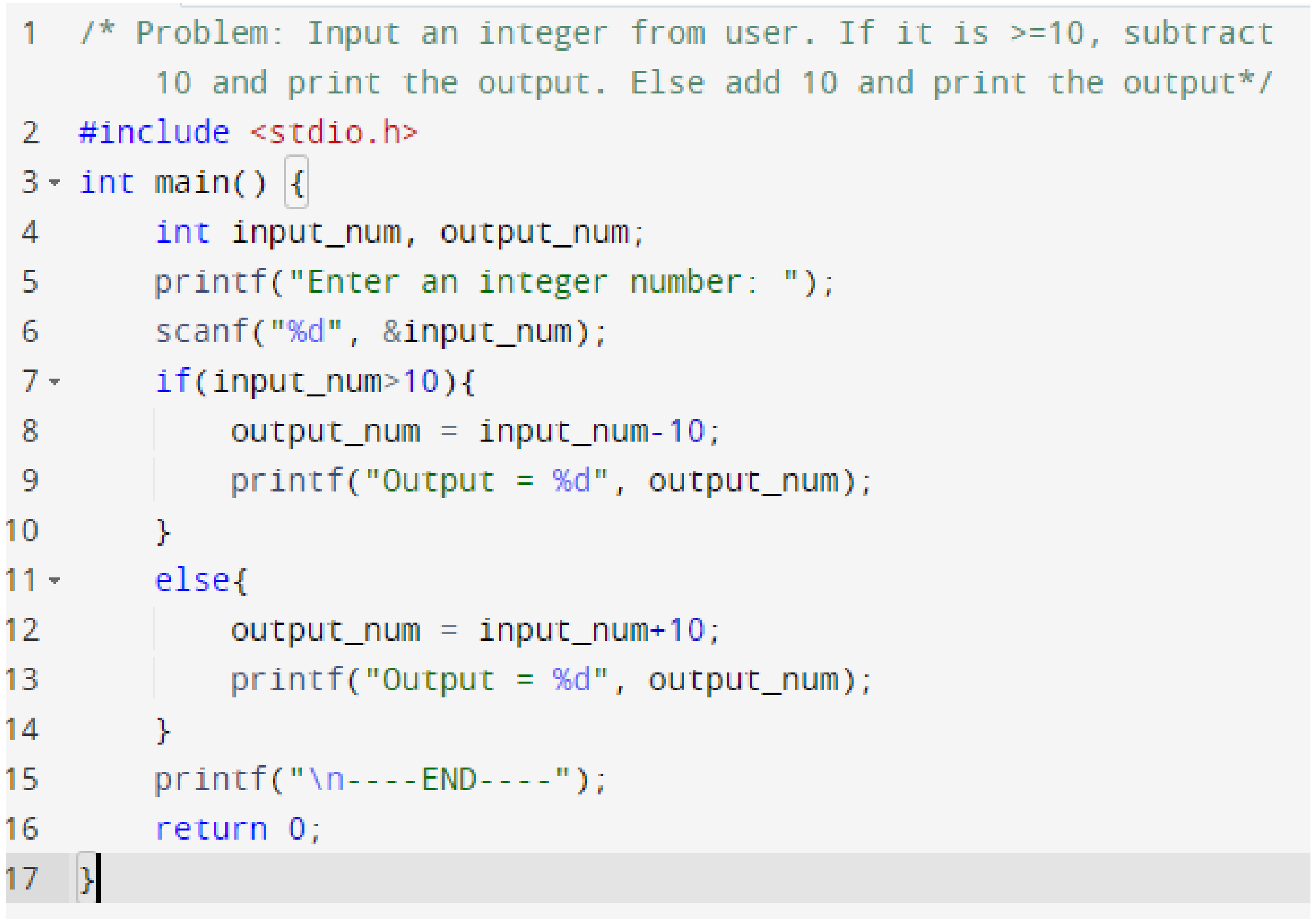
```c
/* Problem: Input an integer from user.
If it is >10, subtract 10 and print the output.*/
#include <stdio.h>

int main() {
    int input_num, output_num;

    printf("Enter an integer number: ");
    scanf("%d", &input_num);

    if(input_num>10)
    {
        output_num = input_num-10;
        printf("Output = %d", output_num);
    }
    printf("----END----");
    return 0;
}
```

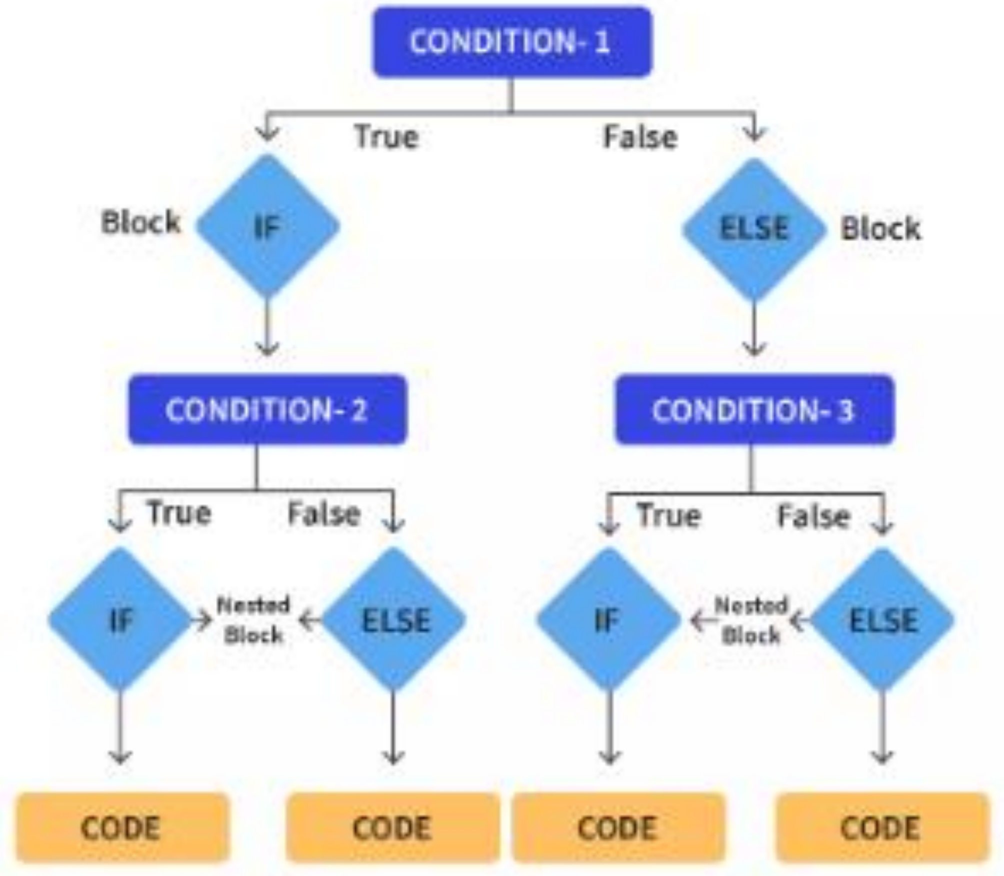
if-else statement



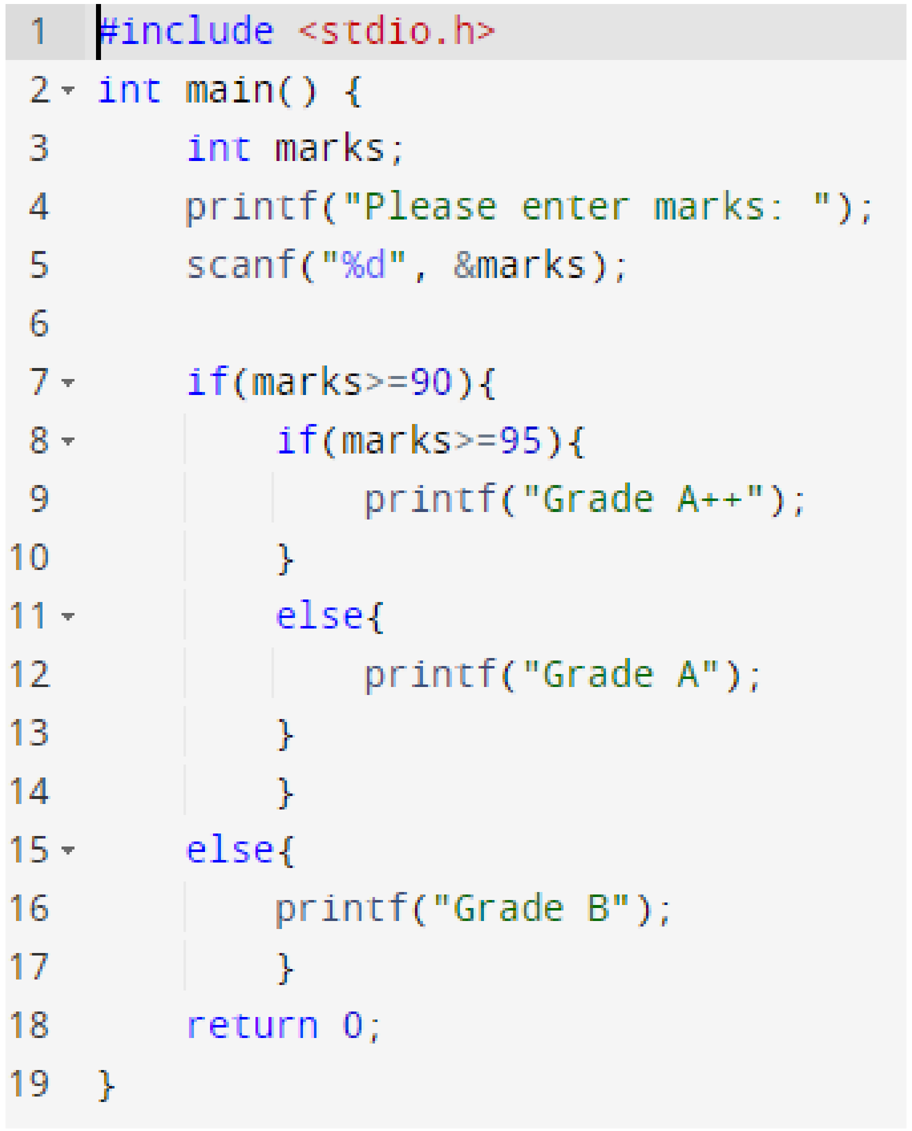
```c
/* Problem: Input an integer from user. If it is >=10, subtract
    10 and print the output. Else add 10 and print the output*/
#include <stdio.h>
int main() {
    int input_num, output_num;
    printf("Enter an integer number: ");
    scanf("%d", &input_num);
    if(input_num>10){
        output_num = input_num-10;
        printf("Output = %d", output_num);
    }
    else{
        output_num = input_num+10;
        printf("Output = %d", output_num);
    }
    printf("\n----END----");
    return 0;
}
```

Nested if-else statement

1. Check a condition within a condition



if-else statement



```c
#include <stdio.h>
int main() {
    int marks;
    printf("Please enter marks: ");
    scanf("%d", &marks);

    if(marks>=90){
        if(marks>=95){
            printf("Grade A++");
        }
        else{
            printf("Grade A");
        }
        }
    else{
        printf("Grade B");
        }
    return 0;
}
```

Forms of if statement

1. Forms of if statement
    a) Simple if
    b) If-else
    c) Nested if-else within either if, or else, or both if-else