
# UESTC 1005 – Introductory Programming

## Lecture 3 – Operators and Program Control

Dr Hasan T Abbas
Hasan.Abbas@glasgow.ac.uk

Fall 2019

Glasgow College – UESTC

# Constants

- When some values are repeatedly used, it is better to give them names and create a constant, through macro definition
  - PI is 3.14159

```
#define PI 3.14159
```

- Convention is we only use UPPERCASE letters to define constants
- `#define` is another preprocessor directive

# Constants - Example

```
#include<stdio.h>

// Another directive
#define INCH_CM 2.54

// A program to convert inches to centimeters
int main(){

    float measurement, new_units;

    measurement = 39.37;

    // Convert using constant
    new_units = measurement * INCH_CM;

    printf("%f inches are %f centimeters", measurement, new_units);

    return 0;
}
```

# Operators

- Arithmetic Operators
  - Binary ( +, - , *, /, %)
  - Unary ( ++ , --)
  - Mixed ( += , -=, *=, /= )

If `N=4; X=N++; Y=++N;`
then after execution
X is 4,
Y is 6
and N is 6.
We can used unary operators for `int` and `float`

# Operators - Example

## Modulo (Remainder Operator)

Get the individual digits in an integer number

```
int main(){

    int a;
    a = 435;
    printf("%d\n", a % 10);
    a /= 10;
    printf("%d \n", a % 10);
    a /= 10;
    printf("%d \n", a % 10);
    return 0;
}
```

# Relational Operators – Decision Making

| Standard algebraic equality operator or relational operator | C equality or relational operator | Example of C condition | Meaning of C condition |
|---|---|---|---|
| *Equality Operators* | | | |
| = | == | x == y | x is equal to y |
| ≠ | != | x != y | x is not equal to y |
| *Relational Operators* | | | |
| > | > | x > y | x is greater than y |
| < | < | x < y | x is less than y |
| >= | >= | x >= y | x is greater than or equal to y |
| <= | <= | x <= y | x is less than or equal to y |

# Care on Operators

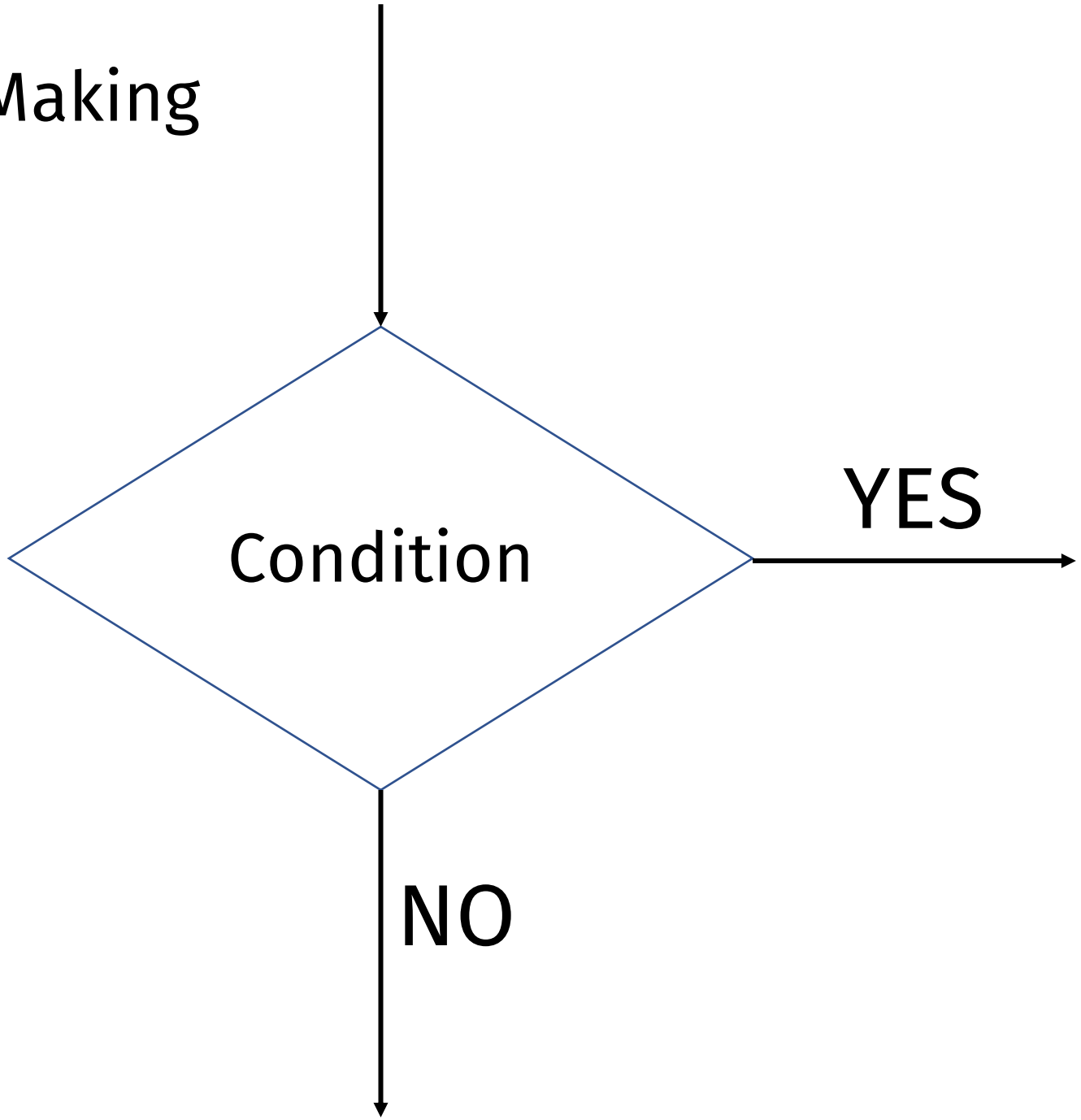
Some Expressions give us warnings

We should not change value of more than one variable in a single statement

```
int main(){

    int a, b, c;
    a = 10;
    b  = 5;
    c = 1;
    c = ( b  = a + 2) – (a = 1);
    printf(“Value of a: \n”, a);
    printf(“Value of b: \n”, b);
    printf(“Value of c: \n”, c);
    return 0;
}
```

# Decision Making



# Applying Conditions

Applying logic is one of the most important features of programming languages

What happens to my grades IF I complete all the labs.

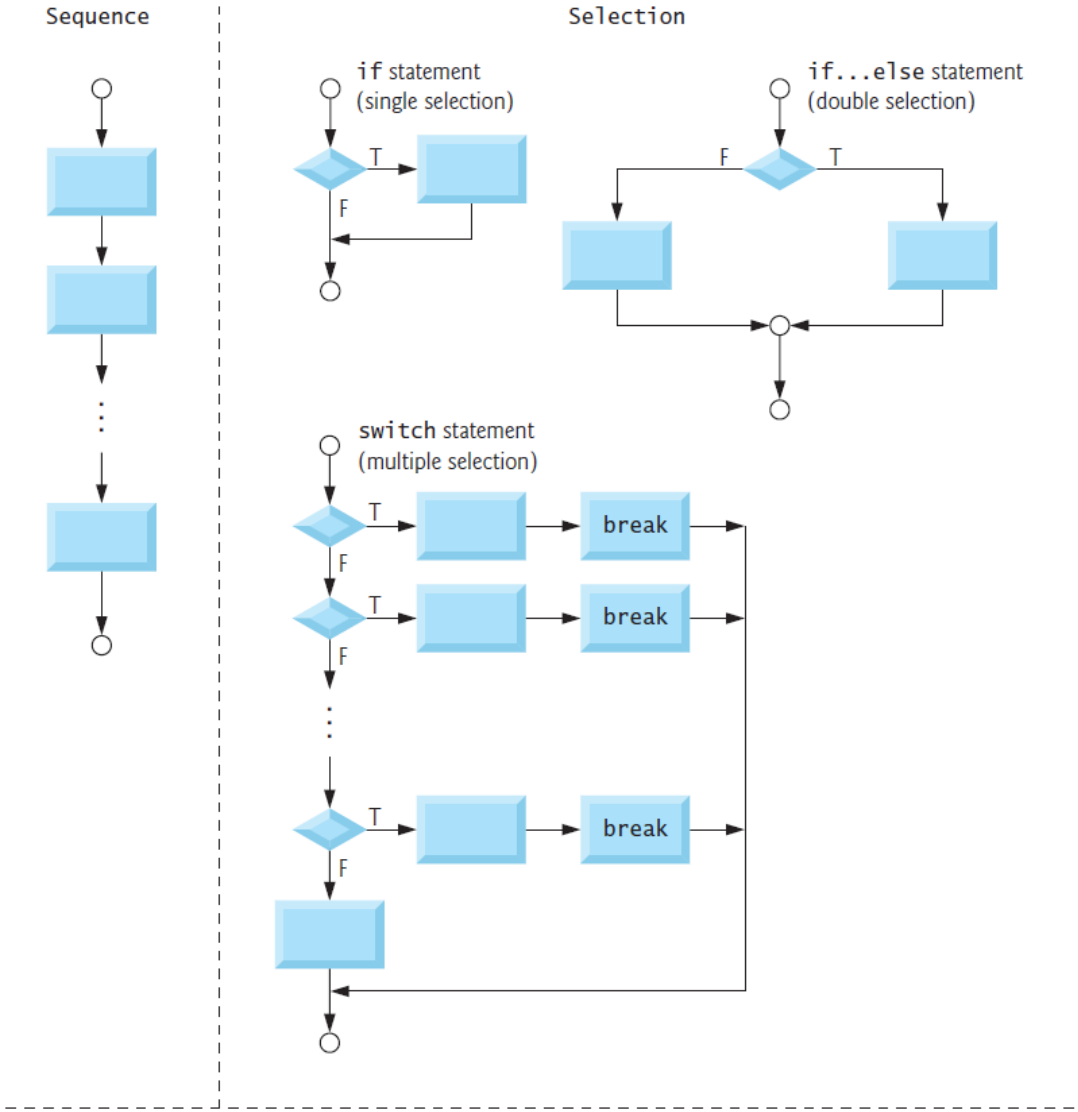
Erin will only go to party IF Qin is going.

IF it is warm outside, THEN turn on the AC, ELSE keep it off.

```
int main(){

    int thermostat = 25; // warm temperature ☹
    int room_temperature = 30;
    int ac_switch; // ON -> 1; OFF -> 0
    if (room_temperature > thermostat)
    {
            ac_switch = 1; // Turn ON the air conditioner
            printf("It is HOT! ;-0 \n\n TURNING ON THE AC");
    }
    return 0;
}
```

# Applying Conditions



# Next Lecture ...

Continue conditional statements

- `if() then elseif(), else`

Loop Environments

- `while() loop`
- `for() loop`