Programming Fundamentals (CS-302)

(Decisions & Loops)

Dr. Ihsan Ullah

Lecturer
Department of Computer Science & IT
University of Balochistan

Outline

- Decisions
  - The if statement
  - The else statement
  - Nested if else
  - The switch statement
  - Conditional operators
- Loops
  - The for loop
  - The while loop
  - The do while loop

The if statement

- The if statement enables to decide whether a statement or a block of statements will execute or not
- The general form is:

```
if (condition)
        Statement;
```

  - The condition part contains an expression that returns true or false
  - In case the condition is true, the statement is executed, otherwise the statement is skipped

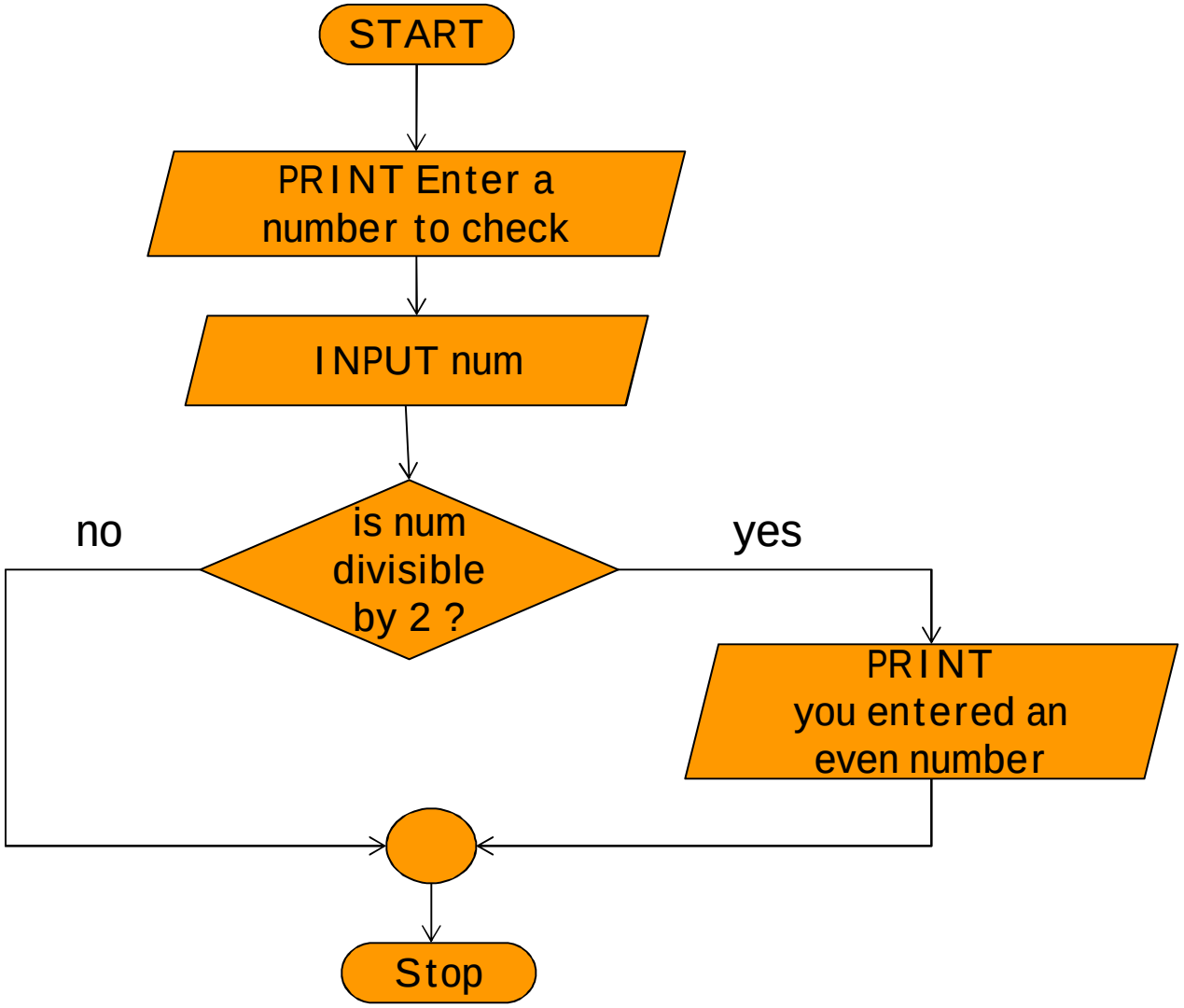
Example 2.1: checking the even number

```
# include <stdio.h>
   main()
   {
       int num;
       printf("Enter a number to check:");
       scanf("%d",&num);
       if(num%2==0)
         printf("You entered an even number");
   }
```

Flowchart

- Graphical representation of an algorithm
- Symbols
  - terminator  start or end
  - Flow line  flow direction
  - Parallelogram  input/output operation
  - Rectangle  process
  - Diamond  Decision or branch
  - Connector  Junction

Example 2.1, Flowchart



The else statement

- In example 2.1, if an odd number is entered, nothing is displayed
- How we can display that the number was odd?
- The *else* statement
- Add before the ending brace of example 2.1

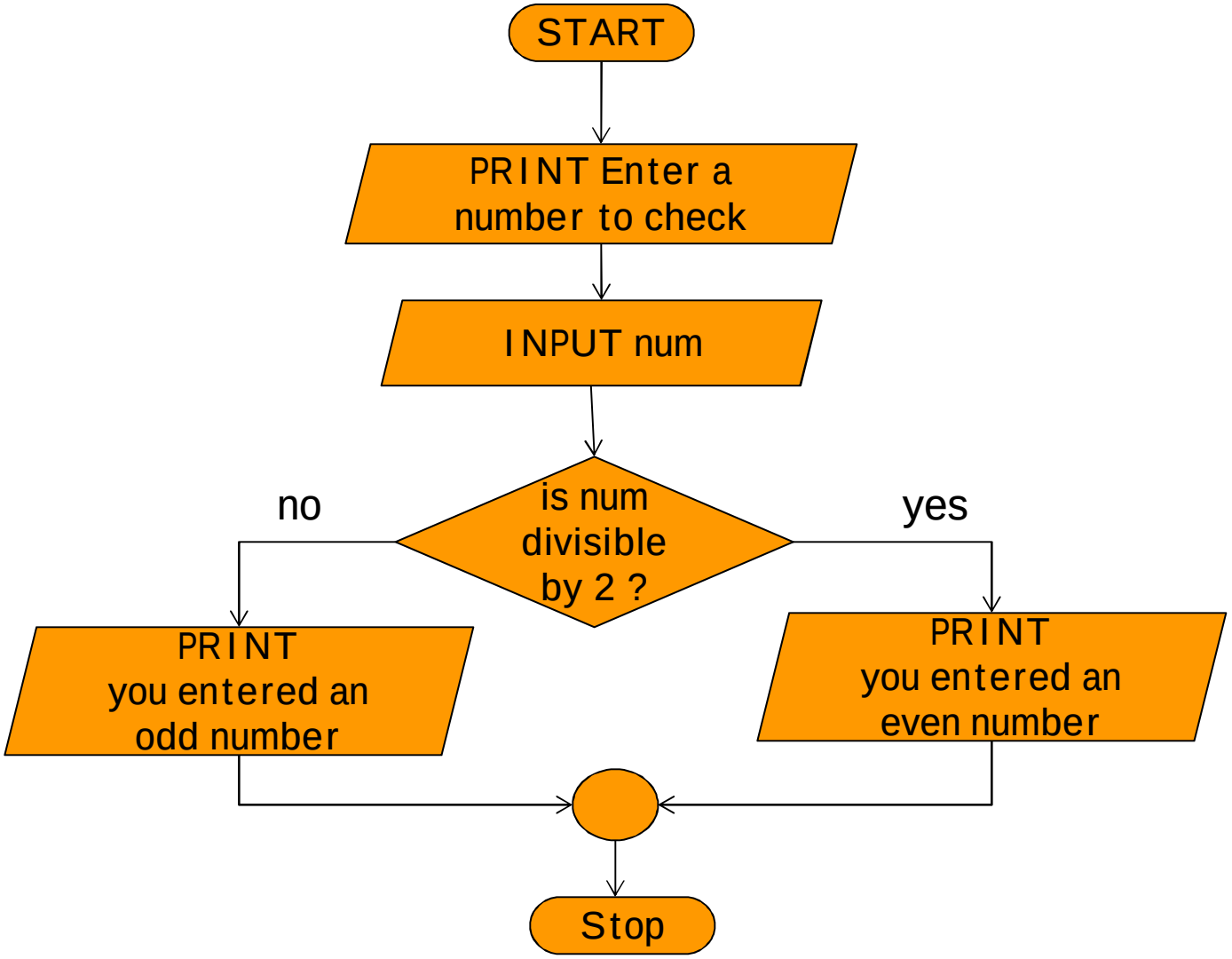
```
else
    printf("you entered an odd number");
```

- A statement in the else part is executed when the condition in if results into false

Example 2.2

```
# include <stdio.h>
main()
{
    int num;
    printf("Enter a number to check:");
    scanf("%d",&num);
    if(num%2==0)
       printf("You entered an even number");
    else
       printf("You entered an odd number");
}
```

Example 2.2, Flowchart



Multiple statements under if

- To allow executing more than one statement, depending on the condition in *if*

```
if(condition)
 {
      Statement1;
      Statement 2;
      …
      Statement n;
 }
```

- In case the condition is true, all the statements within the braces are executed, otherwise the whole block is skipped

Nested if else

- An if statement can be written inside the body of another if or else
- The condition of the inner if will only be evaluated if the body of the outer if or else is executed
- See example nestedIf

Outline

- Decisions
  - The if statement
  - The else statement
  - Nested if else
  - The switch statement
  - Conditional operators
- Loops
  - The for loop
  - The while loop
  - The do while loop

The switch case statement

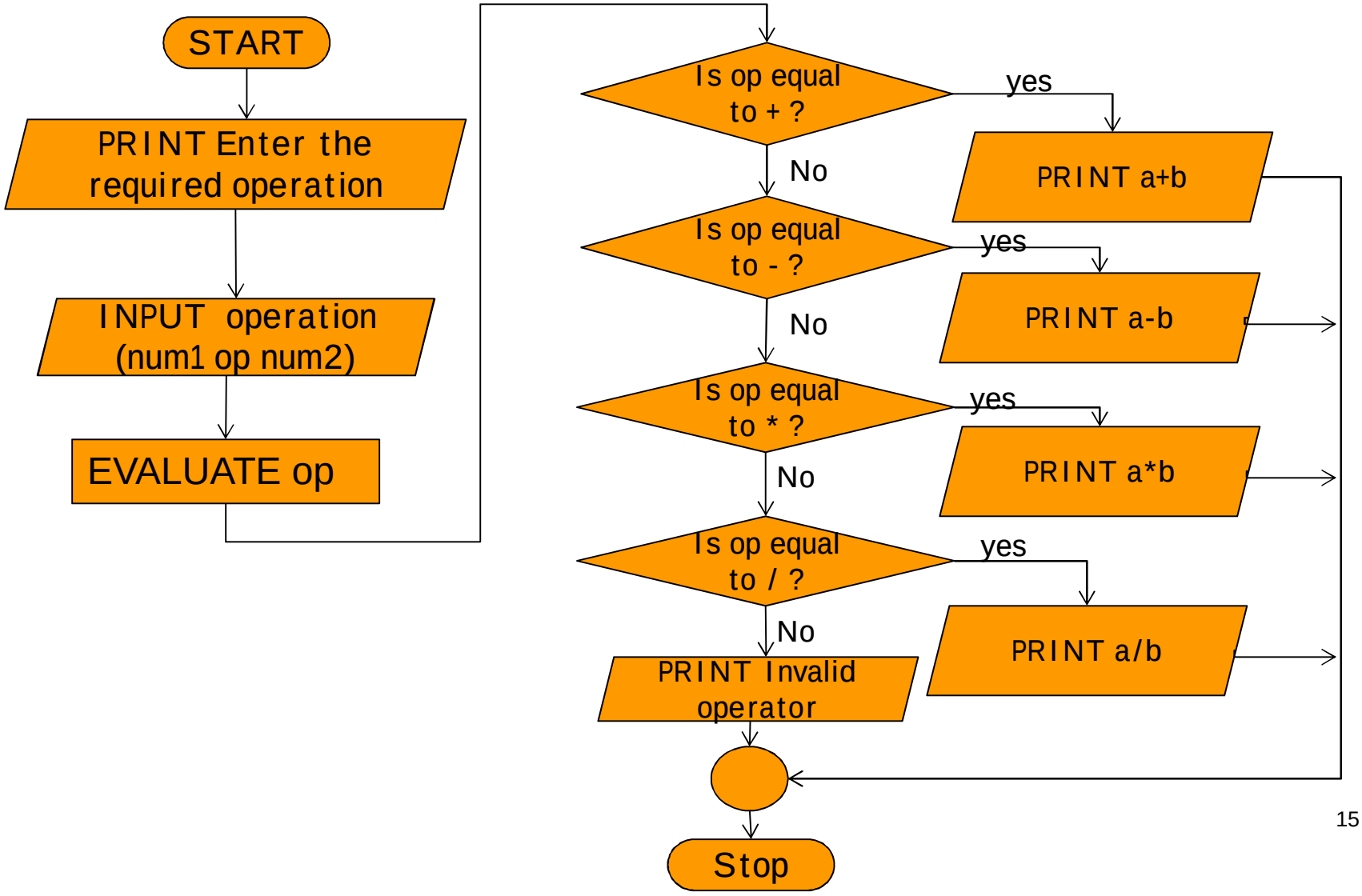
- Choosing one among several options, switch statement provide a better way of coding

```
switch(expression)   //the result of the expression is matched
      {              //with cases
      case expr1:
            statements;
            break;          //exits from the switch block
      case expr2:
            statements;
            break;
       default:     //optional, executed when none of the cases are matched
            statements;
      }
```

The switch case statement (contd.)

- See example 2.3 for a simple calculator
  - Problem?
- Example 2.4 also handles the invalid input

Flowchart example 2.4



Conditional operators

- Also called ternary operators (?, :)
- expression 1 ? expression 2 : expression 3
  - If expression 1 is true then the returned value will be expression 2, otherwise it will be expression 3
- Works like one statement if else constructs

```
if (a<b)
    c=b;
else
    c=a;
```

- Can be written as

```
C=(a<b ? b:a);
```

Example 2.5

```
# include <stdio.h>
main()
{
    int num;
    printf("Enter a number to check:");
    scanf("%d",&num);
    (num%2==0)?printf("Even"):printf("Odd");
}
```

Outline

- Decisions
  - The if statement
  - The else statement
  - Nested if else
  - The switch statement
  - Conditional operators
- Loops
  - The for loop
  - The while loop
  - The do while loop

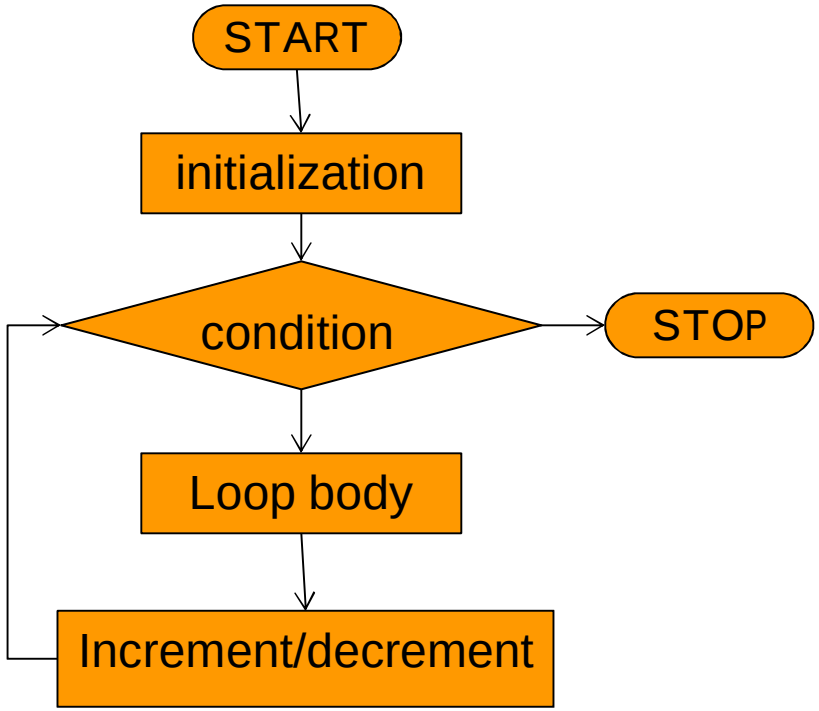
Loop structure

- Loops enable to execute a statement or a block of statements more than one time
- Executing the loop body once is called an iteration
- Three types of loops in C
  - for
  - while
  - do while

The for loop

- Syntax

```
for(initialization; condition; increment/decrement)
{
   Statements;
}
```



Number from 1 to 10

```
# include<stdio.h>
void main()
{
   int i;
   for(i=1;i<=10;i++)
      printf("%d\n",i);
}
```

Generating odd numbers

```
 # include <stdio.h>
void main()
{
    int a;
    for (a=1; a<20; a+=2)
    {
        printf("%d \n",a);
    }
}
```

The while loop

- Syntax

```
while(condition)
  {
    statements;
  }
```

- If the condition evaluates true, the body executes and the process is repeated unless the condition becomes false

While loop contd.

- While loop can be used just like for loop
- Initialization if required, must be done before the loop
- Increment/decrement can be implemented in the body of the loop if required
- While loop is more suitable in scenarios where the number of iterations is not fixed
- See example charcount.c

Do while loop

- Syntax

```
do
   {
   statements;
   }
while(condition);
```

- In do while loop the body executes before the condition is checked
- Allows the body to be executed at least once
- Used in scenarios where a statement or more need to be executed at least once and the repeated depending on some condition
- See example dowhile-goto.c

Goto: labeled jump

- Goto statement is used to transfer control to a labeled point in the program
- Excessive use of goto is not recommended
- Increases the complexity and reduces the readability
- Makes debugging (finding & correcting errors) difficult
- See example dowhile-goto.c

Continue statement

- Continue statement is used to skip the remaining statements in the body of a loop
- Is used with an if statement
- See example continue.c

Summary

- Three types of loops
  - For
  - While
  - Do while
- For loop is suitable for already known fixed number of repetitions
- While loop is suitable when the number of repetitions are not fixed or known in advanc
- Both can be used interchangeably
- Do while loop is used when the body must execute at least once