MCAST
Institute of Information &
Communication Technology

Conditionals

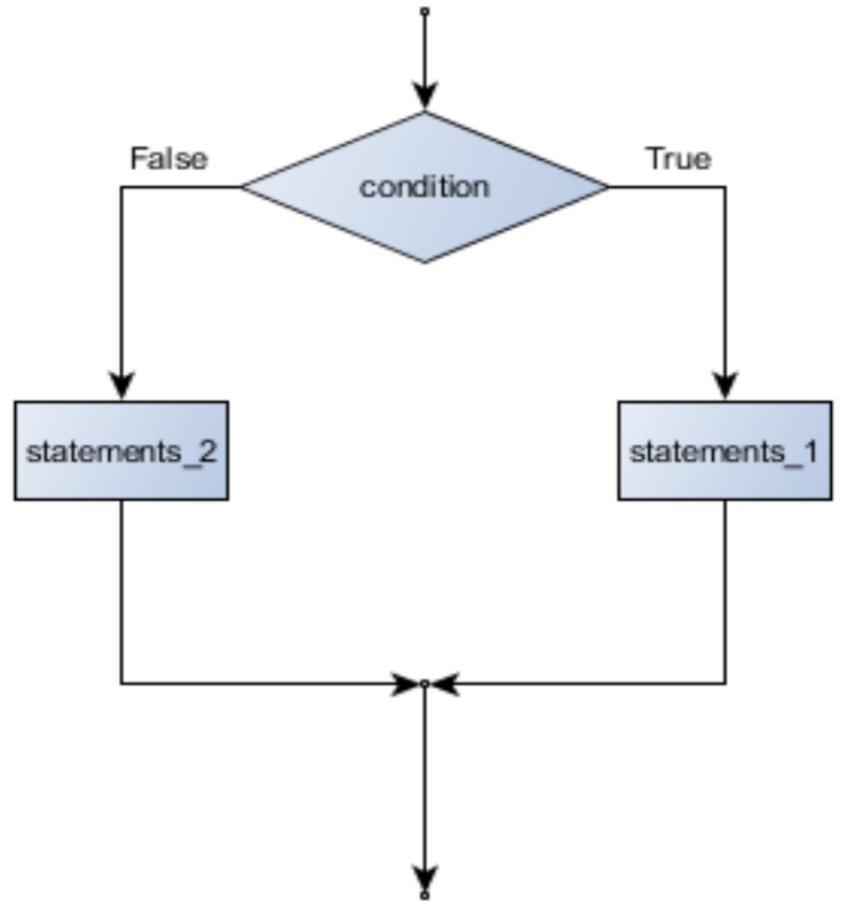
FUNDAMENTALS OF SCRIPTING

Decisions

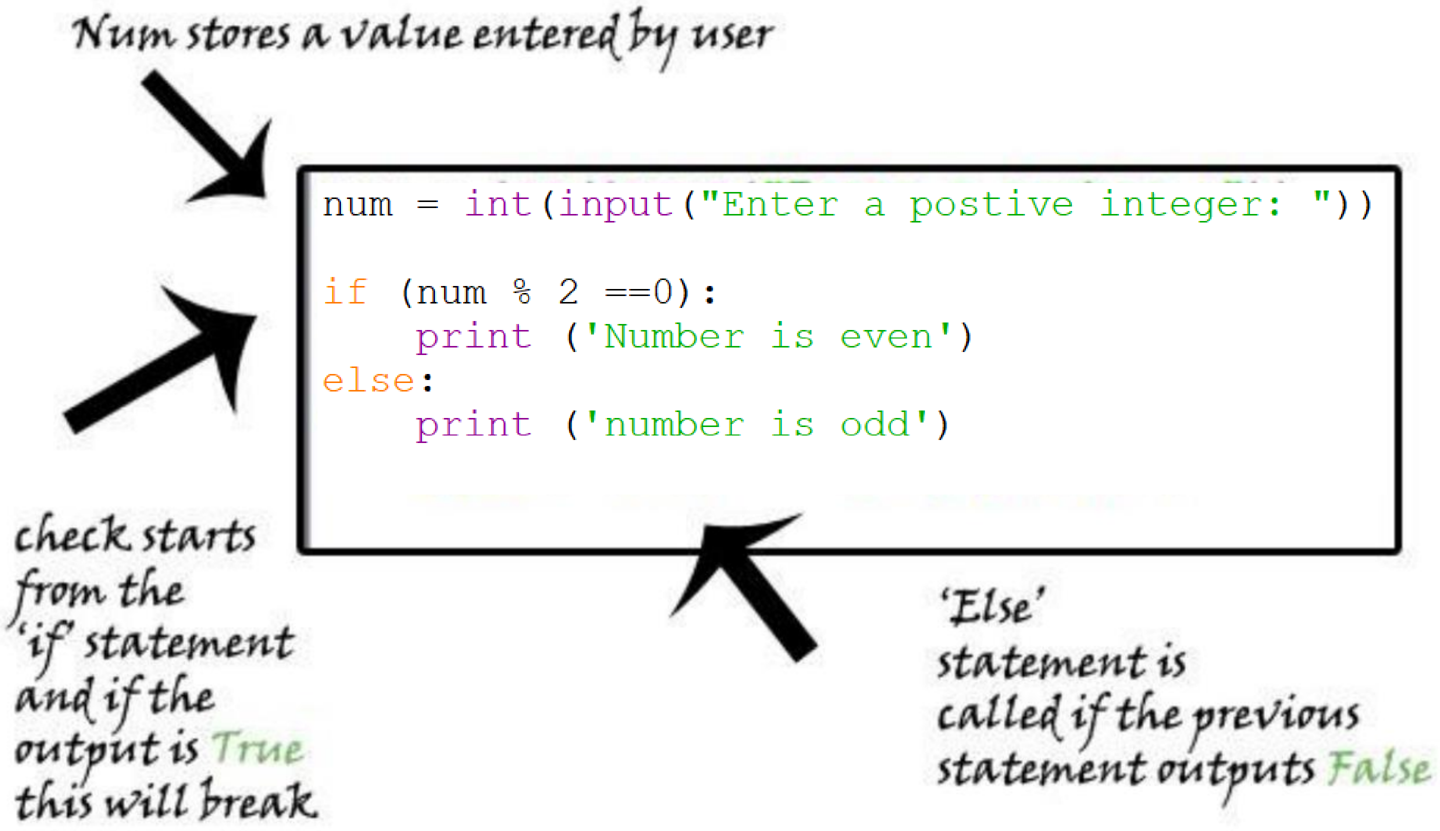
- In real life a decision occurs when there is more than one possible choice of actions depending on a variable's value
- E.g. If we're driving and we have two roads to choose from we must logically choose the road that will take us to our destination

Conditions

- A condition usually compares arithmetic expressions with variables
- These conditions result in a **Boolean** expressions whose value can be either *True* or *False*
- *if*....*else* statement is used for decision taking



Example



```
num = int(input("Enter a postive integer: "))

if (num % 2 ==0):
    print ('Number is even')
else:
    print ('number is odd')
```

Num stores a value entered by user
check starts from the 'if' statement and if the output is True this will break
'Else' statement is called if the previous statement outputs False

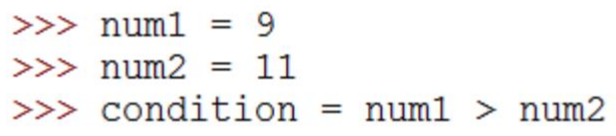
Comparison Operators

- **There are only two boolean values :** *True* **or** *False*
- **Important comparison operators:**

Symbol	Example	Description
==	x == y	x is equal to y
!=	x != y	x is not equal to y
<	x < y	x is smaller than y
>	x > y	x is greater than y
<=	x <= y	x is smaller or equal to y
>=	x >= y	x is greater or equal to y

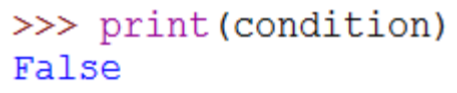
Example

- **What is the value of the variable condition?**



```
>>> num1 = 9
>>> num2 = 11
>>> condition = num1 > num2
```

- **Since num1 is 9 and num2 is 11, we know that 9 is not greater than 11, therefore condition is false**



```
>>> print(condition)
False
```

- **What is the value of the condition variables now?**
  - condition = num1 < num2
  - condition = num1 == num2
  - condition = num1 >= num2
  - condition = num1 <= num2
  - condition = num1 != num2

Logical Operators

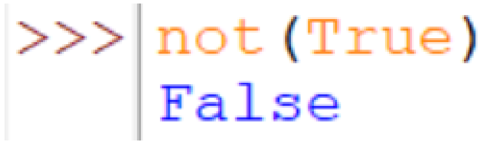
- **Logical operators combine two or more relational expression into a single expression. There are three logical operators:**

Operator	Expression	Description
and	a and b	If both a and b are True, then a and b return True, otherwise return False.
or	a or b	If both a and b are False, then a or b returns False, otherwise it returns True.
not	not a	If a is True, return False. If a is False, it returns True.

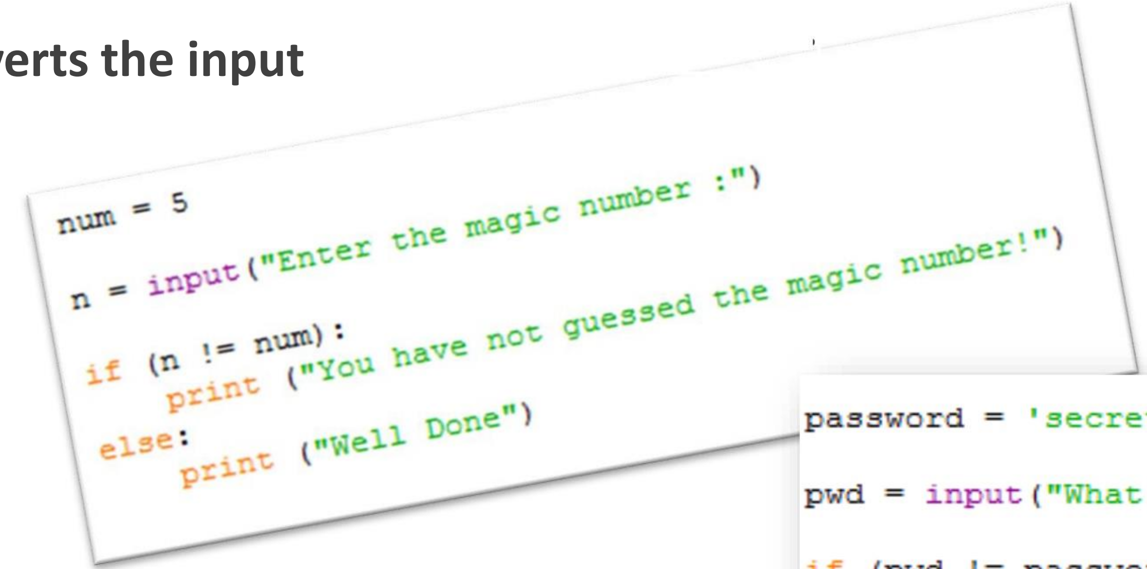
- **Expressions that use logical operators return a Boolean value –** ***True*** **or** ***False***

not Operator

- The *not* operator is a unary operator, i.e. it requires one input
- The input should be a **boolean** type
- The *not* operator inverts the input



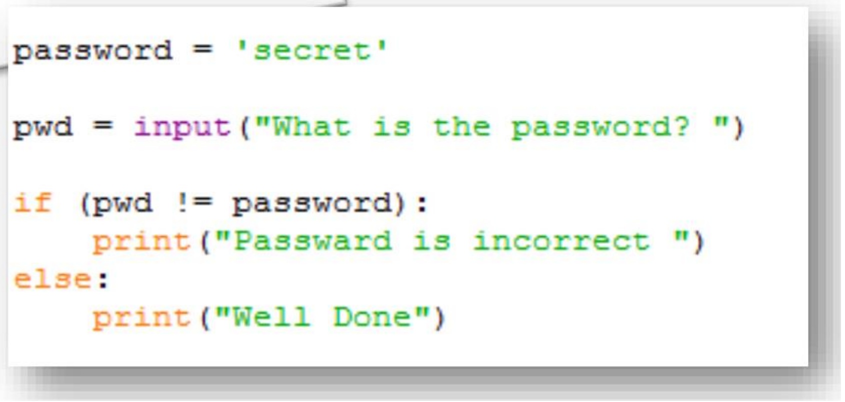
```
>>> not(True)
    False
```



```
num = 5

n = input("Enter the magic number :")

if (n != num):
    print ("You have not guessed the magic number!")
else:
    print ("Well Done")
```



```
password = 'secret'

pwd = input("What is the password? ")

if (pwd != password):
    print("Passward is incorrect ")
else:
    print("Well Done")
```

Examples using integers and strings

and Operator

- The *and* operator is binary, i.e. it requires two inputs
- The inputs should be boolean types
- If both inputs are set to true then the *and* operator returns *True*, otherwise, the result is *False*

```
youngest = 18
oldest = 70
age = int(input("Enter your age:  "))
if (age >= youngest) and (age <= oldest):
   print("You are eligible to drive")
else:
   print("You are not eligible to drive")
```

Input1	Input2	Result
False	False	False
False	True	False
True	False	False
True	True	True

or Operator

- The *or* operator is binary, i.e. it requires two inputs
- The inputs should be boolean types
- If, at least, one of the inputs is set to *True* then the *or* operator returns *True*, otherwise, the result is *False*

```
if today=='Sunday' or today=='Saturday’:
    print('Today is off. Rest at home.’)
else:
    print('Sorry, but you need to go to school!')
```

Input1	Input2	Result
False	False	False
False	True	True
True	False	True
True	True	True

Chained Conditionals: *if-elif-else*

- **Sometimes there are more than two possibilities and we need more than two branches**
- **Example code using a chained conditional:**

```
x = 2
y = 3
if (x == y):
    print('x and y are equal')
elif (x > y):
    print('x is greater than y')
else:
    print('x is smaller than y')
```

- **Result:**

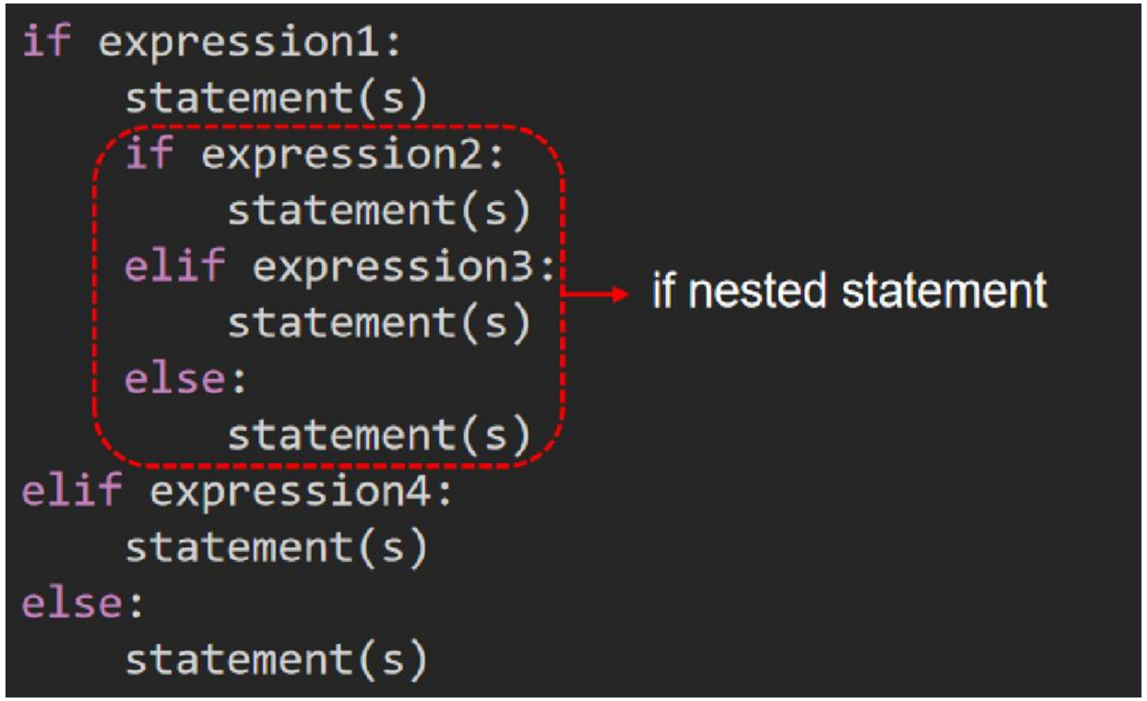
```
x is smaller than y
```

Chained Conditionals

- *elif* is an abbreviation of “else if.” In the previous example, exactly one branch will be executed
- Each condition is checked in order
- If the first is *False*, the next is checked, and so on
- If one of them is *True*, the corresponding branch executes, and the statement ends
- Even if more than one condition is *True*, only the first *True* branch executes

if Nested Statement

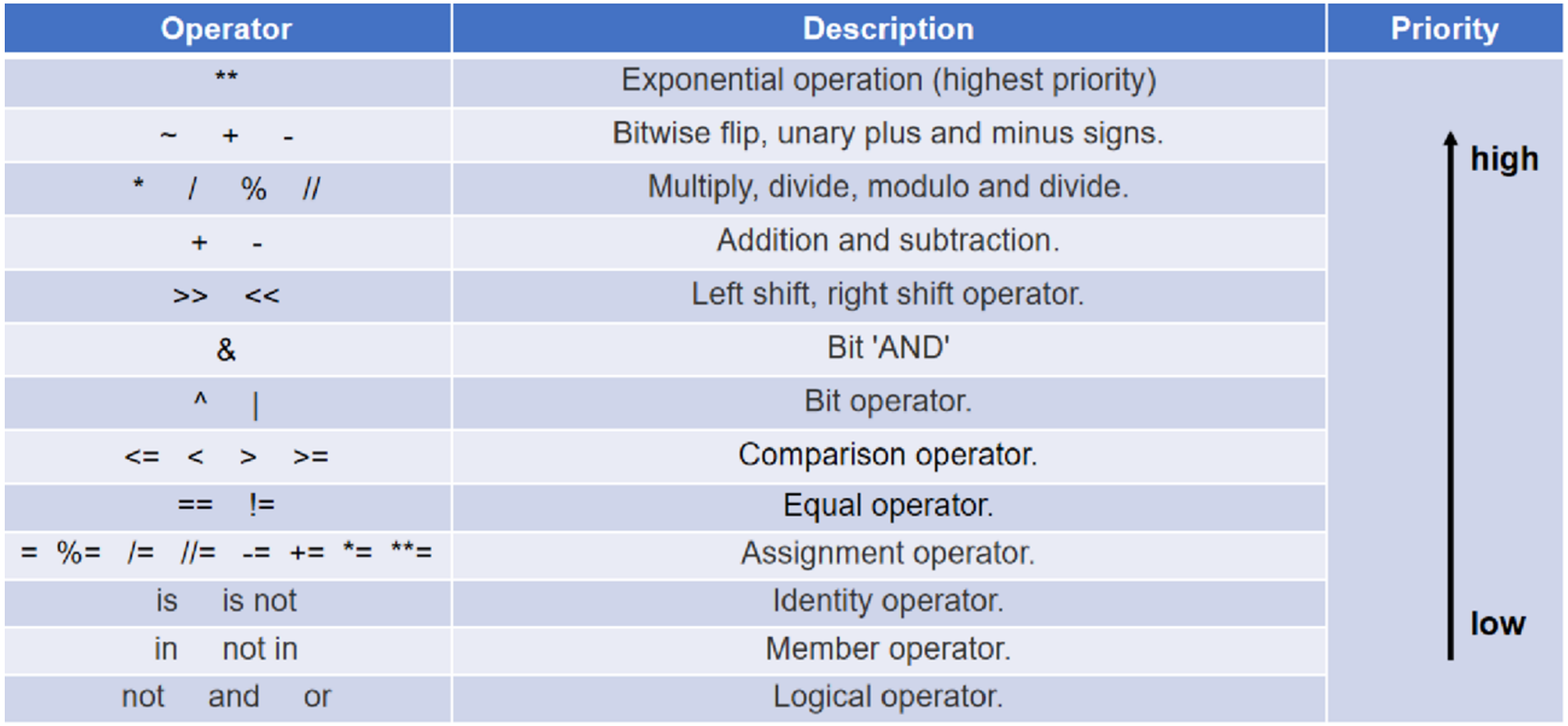
- In a nested *if* statement an *if-elif-else* structure can be placed within another *if-elif-else* structure
- Grammatical format:



```
if expression1:
    statement(s)
    if expression2:
        statement(s)
    elif expression3:
        statement(s)
    else:
        statement(s)
elif expression4:
    statement(s)
else:
    statement(s)
```

Priority

- **All operators have an order of precedence as shown below:**



Operator	Description	Priority
**	Exponential operation (highest priority)	
~ + -	Bitwise flip, unary plus and minus signs.	↑ high
* / % //	Multiply, divide, modulo and divide.	
+ -	Addition and subtraction.	
>> <<	Left shift, right shift operator.	
&	Bit 'AND'	
^ |	Bit operator.	
<= < > >=	Comparison operator.	
== !=	Equal operator.	
= %= /= //= -= += *= **=	Assignment operator.	
is is not	Identity operator.	low
in not in	Member operator.	
not and or	Logical operator.	

Try out worksheet 4