Conditional statements:

In order to write useful programs, we always need the ability to check *conditions* and change the behavior of the program accordingly. **Conditional statements** give us this ability. The simplest form is the *if statement*:

```
if x > 0:                       #notice the ":" at the end of the if line
    print 'x is positive'       #notice the indentation after if
```

The *boolean expression* (x>0) after if is called the **condition**. If it is true, then the indented statement (i.e. the print statement in this case) gets executed. If not, nothing happens.

Boolean expression:

A **boolean expression** is an expression that is either true or false. It usually uses the following operators (called relational operators):

Relational operators	Meaning
x == y	is equal to
x != y	# x is not equal to y
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

Enter the following instructions in interactive mode and write down python's response in the bracket

1. >>> 5 ==6 ( )
2. >>> 14 == 14 ( )
3. >>> 15 <= - 11 ( )
4. >>> 33 != 24 ( )

The following example is a program to check whether a mark entered passes or fails in the test.

Programs

```
# an example to show the use of if ..... then....

x = input("please enter a mark")
mark = int(x)
if mark >= 50:
  print (mark, "passes the test")      # notice that if mark less than 50, nothing will happen
```

```
#an example to show the use of if …. then ….. else…..
x = input("please enter a mark ")
mark = int(x)
if mark >= 50:                          #pay attention to if … and :
      print (mark, "passes the test")   #pay attention to indentation
else:                                   #pay attention to else: (with a :)
     print (mark, "fails!")             #pay attention to indentation
```

Notice that here is no limit on the number of statements that can appear in the body (after if), but there has to be at least one. Occasionally, it is useful to have a body with no statements. In that case, you can use the **pass** statement, which does nothing. For example

```
if x <0:
    pass          #no need to handle negative number now. But maybe later.
```

Program practice:

1. Write a program to check whether a number entered is even or odd. An sample output message is shown as follows with 35 and 44 are user's input:

```
What is your number?  35
35 is an odd number.
What is your number? 44
44 is a even number.
```

2. Modify (1) so that the program is used to check whether a number entered is divisible by 6.

Chained conditionals:

Modify program (1) so that it produces output as follows:

```
What is your number?  35
35 is an odd number.
What is your number? 44
44 is a even number.
What is your number? 0
0 is neither even or odd
```

First attempt (with undesirable output):

```
num = int(input("What is your number? "))
remainder = num%2
if remainder == 0:
   print (num, "is a even number")
elif remainder != 0 :
   print (num, "is an odd number")
else:
   print ("0 is neither even nor odd")
```

Programs	

#Sample workable program 1
#the use of elif (else if)
num = int(input("What is your number? "))
remainder = num%2
if remainder != 0:
  print (num, "is an odd number")
elif num == 0 :
  print ("0 is neither even or odd")
else:
  print (num, "is a even number")
	# Sample workable program2
	num = int(input("What is your number? "))
	if num == 0:
	  print ("0 is neither a even or odd number")
	elif num%2 == 0:
	  print (num, " is an even number")
	else:
	  print (num, "is an odd number")

The above programs can further be modified using *nested if; ie..* if statement within a if-statement.

Program

```
num = int(input("What is your number? "))
if num==0:
   print ("0 is neither even nor odd")
remainder = num%2
if remainder == 0:
  if num != 0:
      print (num, "is a even number ")
else:
      print (num, "is an odd number")
```

In using nested-if statement, pay attention to the indentation.

Logical operators:

There are three **logical operators**, *and, or*, and *not* which are often used in conditional statements.
The above program can be further modified using the logical operator "and".

For example: the program segment from the left can be changed to the right by using "**and"** operator.

Program

```
# the use of "and"
num = int(input("What is your number? "))
if num == 0:
   print ("0 is neither even nor odd")
else:
   remainder = num%2
   if remainder == 0 and num != 0:
      print (num, "is a even number ")
   else:
      print (num, "is an odd number")
```

Program practice (for submission):

Choose either one of the following programs:

1. Write a program that requests a user to enter a mark. Your program should then
   i. allocate the grade to the mark according to the table on the right hand side.
   ii. be able to check errors from the user. For example, if the user enters a negative mark or textual mark, the program should display an error message.

Marks	Grade
0 – 39	Fail
40 – 59	Pass
60 – 79	Credit
80 – 100	Distinction

2. Write a program that requests a user to enter three numbers. Your program should then
   i. print out the highest and lowest numbers among the three. If the user enters 6, 17 and -15, the output should be the highest number is 17 and the lowest number is -15.
   ii. be able to handle ties properly too. If the user enters 6, 6, 3, the program should report 6 is the highest and 3 is the lowest.
   iii. report "there is no highest or lowest number", if the user enters 6, 6, 6.