# Lesson 2 - Selection

|  |  |
| --- | --- |
| **Lesson Outcomes**  In this lesson you will learn:   * how to change the flow of a program; * the implementation of the IF… Else command; * comparison and logical operators; * and the more advanced multiple IF concept. | **C:\Users\Graham\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\X6CHINOH\MC900441498[1].png** |

## Changing the flow of a program

In programming one of the most powerful concepts is SELECTION. Selection allows us to change the natural flow of a program so that decision making can be included. You can represent this easily using program flowcharts:

Input Number1

Input Number2

No

Is Number 1 > Number 2?

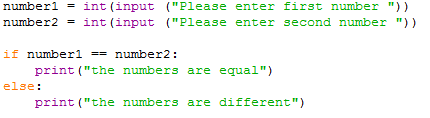
Output Number 2

Yes

Output Number1

The above program asks the user to enter two numbers which are stored in separate variables. These variables are then compared and the one with the largest value is outputted.

This is a fairly simple way of changing the natural sequence of commands in a program. In Python this is implemented using the **“IF”** command. It is used to test if a condition is met and then change the program flow accordingly. You normally write the expression in the following format:



Statement executed if condition is **TRUE.**

**Condition** e.g. is number1 equal to number2. This is followed by a colon **“:”.**

The **ELSE** command is used when the CONDITION above is **FALSE.**

In the above example the **CONDITION** is included after the “**if”** statement, with the colon representing the end. The print command is indented on the next line to show this is the part of the program executed if the condition is TRUE.

**NOTE: INDENTING** is very important in Python as it tells it how to **BLOCK** and structure the code**.**

The else command is used to tell Python the indented block code below is executed if the “IF” condition was false.

In the above program you will notice the “==” command has been used to check if the two numbers are equal. The reason why two equal signs have been used as opposed to one will be explained later. As well as equals you also have the following comparison operators:

|  |  |
| --- | --- |
| **Operator** | **Description** |
| == | Equals to |
| > | Greater than |
| < | Less than |
| != | Not equal to |
| >= | Greater than or equal to |
| <= | Less than or equal to |

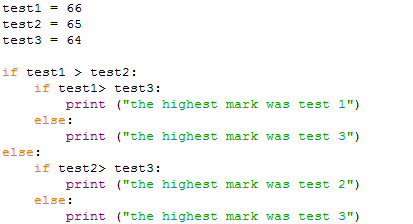
### Tasks

2.1 Write a program that asks for two numbers from the user and then displays a suitable message if the two numbers are the same (remember to convert user input to number using the int command)

2.2 Write a program that asks the user to enter 2 numbers and displays the larger of the two numbers.

## Multiple “IF” statements

So far you have learned how to create a simple **“IF..ELSE”** statement however there are times when more complex statements are needed. In Python, and indeed in most programming languages, it is possible to include **MULTIPLE IF** statements. For example, say we wanted to find the largest of three numbers:



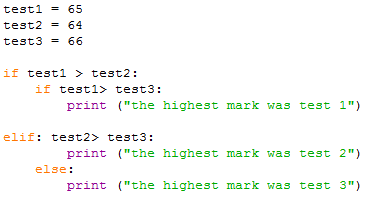
**INDENTING** print statement

ELSE statement if SECOND IF FALSE

2nd IF: if first condition TRUE

1st IF Condition

This program goes through and systematically compares each number until the largest one is found. You will notice that there is an IF within an IF from line 5 onwards – this is called a **Multiple IF statement.** As well as using the above structure you may also use the **elif** command for when there are lots of multiple IF statements.



elif statement

### Tasks

2.3 Write a program that asks the user to enter 3 numbers and displays the largest.

2.4 Write a program which asks their age and display whether they can drive or not.

2.5 Write a program to input 3 numbers and display which one is the largest.

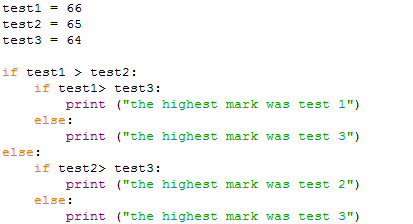
2.6 Write a program that lets the user enter a number between 1 and 12 and displays the month name for that month number. The input 3 would therefore display March.

2.7 Write a program that reads in the temperature of water in a container (in Centigrade) and displays a message stating whether the water is frozen, boiling or neither.

2.8 Write a program that asks the user for the number of hours worked this week and the hourly rate of pay. The program is to calculate the gross pay. If the number of hours worked is greater than 35, the extra hours are paid at 1.5 times the rate. The program should display an error message if the number of hours worked is not in the range 0 to 70.

## Logical Operators

In the above exercises you have written programs that use many **IF and ELSE** commands using a single **CONDITIONAL** statement. Using **multiple IF** statements works well however it isn’t always the most efficient way of constructing the statement. Take for example the largest number program shown earlier:

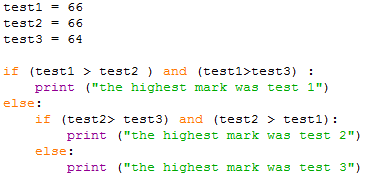


Although this program functions correctly there is a lot of duplication and unnecessary statements e.g. two print outputs are the same but copied twice for the different conditions. To tidy this we can use **LOGICAL OPERATORS.** Python supports the following operators:

|  |  |  |
| --- | --- | --- |
| **Operator** | **Description** | **Example** |
| and | Called Logical AND operator. If both the operands are true, then the condition becomes true. | (a and b) is true. |
| or | Called Logical OR Operator. If either or both true then the output is true. | (a or b) is true. |
| not | Called Logical NOT Operator. Use to reverse the logical state of its operand. If a condition is true, then Logical NOT operator will make it false. | Not (a and b) is false. |

So using these operators we can simply the above program:

Two CONDTIONS joined by an **AND** operator. Both have to be **true** for the condition to return true.



We have replaced the two **multiple IF** statements with one single **CONDTION** using the **AND** operator. Notice how each condition is put in brackets; although not required in Python it helps readability.

### Tasks

2.9 Using logical operators, write a program that asks the user to enter 3 numbers and displays the largest.

2.10 Write a program which given a percentage score displays the grade:

0 – 39 (E)

40-45 (D)

46-55 (C)

56-65 (B)

66-75 (A)

76- (A\*)

2.11 Write a program that asks the user for a month number and displays the number of days that month has.

2.12 Extend the program for task 2.11 to include leap years. The program should prompt for year and month. A year is a leap year if the year divides exactly by 4, but a century is not a leap year unless it is divisible by 400. For example the year2008 was a leap year, the year 1900 was not, and the year 2000 was a leap year.

2.13 Write a program that accepts a date as three separate integers such as 12, 5, 03. The program should display the date in the form 12th May 2003. No validation necessary e.g. the program won’t check if the number of days is allowed for the month specified.

2.14 Adapt your program to interpret a date such as 12 5 13 as 12th May 2013. Your program should interpret the range 1931 to 2030.