Conditional Statement (if)

1. **If statements:** quite often in program we only want to do something provided something else is true. Python's if statement is what we need. Let's try a guess-a-number program. The computer picks a random number, the player tries to guess, and the program tells them if they are correct. To see if the player's guess is correct, we need something new, called an *if statement*.

```
from random import randint

num = randint(1,10)
guess = eval(input('Enter your guess: '))
if guess==num:
    print('You got it!')
```

The syntax of the if statement has a colon at the end of the if condition and the following line or lines are indented. The liens that are indented will be executed only if the condition is true. Once the indentation is done with, the if block is concluded.

The above program works, but it is pretty simple. If the player guesses wrong, nothing happens. We can add *else* to the if statement as follows.

```
if guess==num:
    print('You got it!')
else:
    print('Sorry.  The number is ', num)
```

The else statement is like an "otherwise".

2. **Conditional operators:** the comparison operators in python are explained in table below.

Expression	Description
`if x>3:`	if x is greater than 3
`if x>=3:`	if x is greater than or equal to 3
`if x==3:`	if x is 3
`if x!=3:`	if x is not 3

There are three additional operators used to construct more complicated conditions: and, or, and not. Here are some examples:

E-mail: hasanein.y.m.alhrabi@uomus.edu.iq

```
if grade>=80 and grade<90:
    print('Your grade is a B.')

if score>1000 or time>20:
    print('Game over.')

if not (score>1000 or time>20):
    print('Game continues.')
```

In terms of order of operations, *and* is done before *or*, so if you have a complicated condition that contains both, you may need parentheses around the *or* condition. Here is an example:

```
if (score<1000 or time>20) and turns_remaining==0:
    print('Game over.')
```

3. **Common Mistakes:**

Mistake 1: the operator for equality consists of two equals signs. It is a really common error to forget one of the equals signs.

Incorrect	Correct
`if x=1:`	`if x==1:`

Mistake 2: another very common mistake is to write something like below:

```
if grade>=80 and <90:
```

This will lead to a syntax error. We have to be explicit. The correct statement is:

```
if grade>=80 and grade<90:
```

On the other hand, there is a nice shortcut that does work in python:

```
if 80<=grade<90:
```

4. **elif:** a simple use of an if statement is to assign letter grades. Suppose that scores 90 and above are A's, scores in the 80s are B's, 70s are C's, 60s are D's, and anything below 60 is an F. here is one way to do this.

```
grade = eval(input('Enter your score: '))

if grade>=90:
    print('A')
if grade>=80 and grade<90:
    print('B')
if grade>=70 and grade<80:
    print('C')
if grade>=60 and grade<70:
    print('D')
if grade<60:
    print('F')
```

The code above is pretty straightforward and it works. However, a more elegant wat to do it is shown below.

```
grade = eval(input('Enter your score: '))

if grade>=90:
    print('A')
elif grade>=80:
    print('B')
elif grade>=70:
    print('C')
elif grade>=60:
    print('D')
else:
    print('F')
```

With the separate if statements, each condition is checked regardless of whether it really needs to be. That is, it the score is a 95, the first program will print an A but the continue on and check to see if the score is a B, C, etc. which is a bit of a waste. Using *elif*, as soon as we find where the score matches, we stop checking conditions and skip all the way to the end of the whole block of statements. An added benefit of this is that the conditions we use in the elfi statements are simpler than in their if counterparts. For instance, when using elif, the second part of the second if statement condition, grade < 90, becomes unnecessary because the corresponding elif does not have to worry about a score of 90 or above, as such a score would have already been caught by the first if statement.

EXERCISES

1. **Write a program that asks the user to enter a length in centimeters. If the user enters a negative length, the program should tell the user that the entry is invalid. Otherwise, the program should convert the length to inches and print out the result. There are 2.54 centimeters in an inch.**
2. **Ask the user to enter a temperature in Celsius. The program should print a message based on the temperature:**
   a. **If the temperature is less than-273.15, print that the temperature is invalid because it is below absolute zero.**
   b. **If it is exactly-273.15, print that the temperature is absolute 0.**
   c. **If the temperature is between-273.15 and 0, print that the temperature is below freezing.**
   d. **If it is 0, print that the temperature is at the freezing point.**
   e. **If it is between 0 and 100, print that the temperature is in the normal range.**
   f. **If it is 100, print that the temperature is at the boiling point.**
   g. **If it is above 100, print that the temperature is above the boiling point.**
3. **A store charges $12 per item if you buy less than 10 items. If you buy between 10 and 99 items, the cost is $10 per item. If you buy 100 or more items, the cost is $7 per item. Write a program that asks the user how many items they are buying and prints the total cost.**
4. **Write a program that asks the user for an hour between 1 and 12, asks them to enter am or pm, and asks them how many hours into the future they want to go. Print out what the hour will be that many hours into the future, printing am or pm as appropriate. An example is shown below.**

```
Enter hour: 8
am (1) or pm (2)? 1
How many hours ahead? 5
New hour: 1 pm
```

E-mail: hasanein.y.m.alhrabi@uomus.edu.iq