

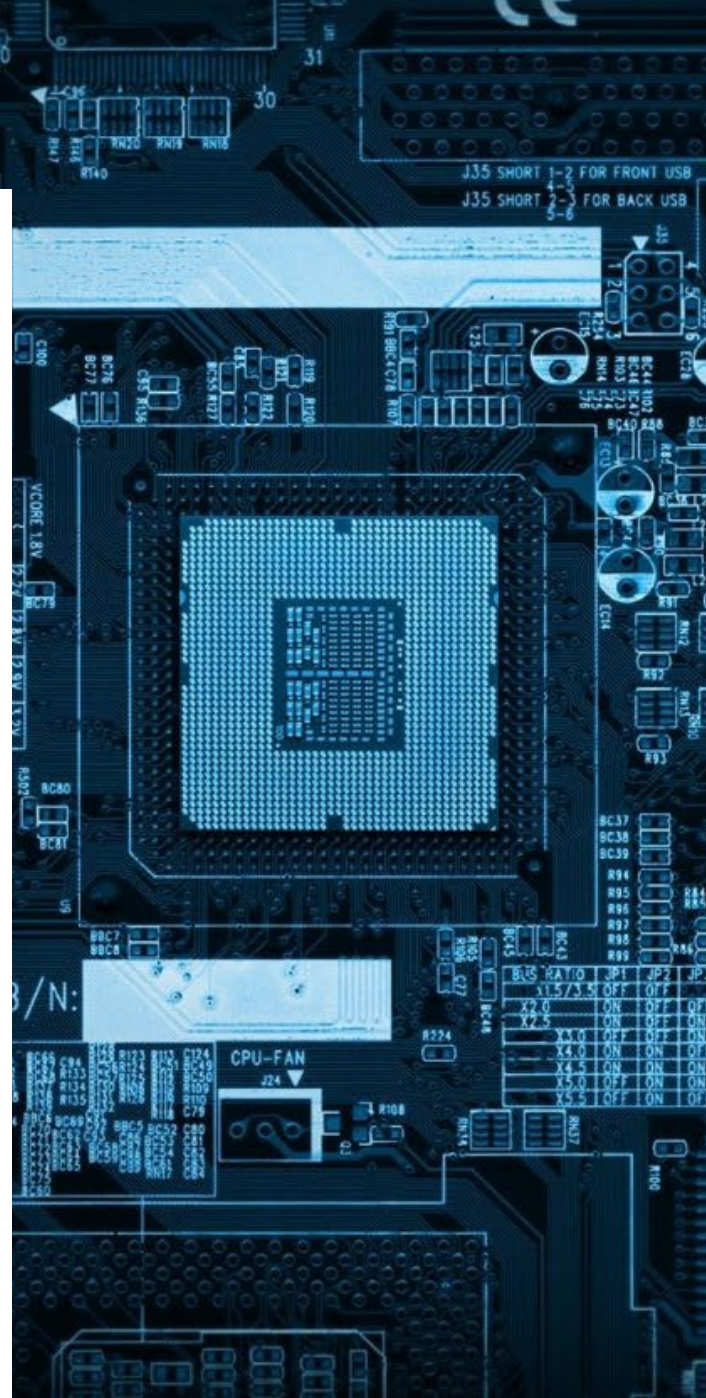
SCMA 249

# Computer Programming in Actuarial Science

---

1/2022

## LAB 2 Selection



# LAB 2

## Selection

### 2.1 Basic commands

#### 2.1.1 Getting input

The `input` function is used to get the input from the user via keyboard. The basic syntax is

```
variable_name = input(message to user)
```

##### Example 2.1

Write a simple code to read your name and keep it in the variable name "name".

```
name = input('Enter your name: ')
```

##### Example 2.2

Write Python code to get numbers from user to use in calculations.

```
num = eval(input('Enter a number: '))
sq = num*num
sq
```

**Note** Type `help(eval)` to see the description of `eval` function.

#### 2.1.2 Printing

The `print` function is used to print a message to the screen (or other output device).

##### Example 2.3

Recall Example 2.1, print out the name by typing `print(name)`.

##### Example 2.4

Recall Example 2.2, print out the number by typing `print('The number is: ', sq)`.

##### Example 2.5

Type the following codes and write down the result shown on the screen.

`print('2*4')` Result \_\_\_\_\_

`print(2*4)` Result \_\_\_\_\_

`print('The value of 2*4 is', 2*4)`

Result \_\_\_\_\_

## 2.2 Conditional operators

A conditional/logical statement requires the conditional operators. There are several conditional operators. Some of them are shown below.

>	greater than
> =	greater than or equal
==	equal
!=	not equal
and	both simultaneously
or	one or other or both
not	logical not

**Note** When you use more than one operator in the complicated conditional statement, the order of operations is concerned. The and is done before or. You are recommended to use parentheses around the or condition.

### Order of an arithmetic operator

1	**	Exponentiation	right to left
2	*/ % //	Multiply, divide, modulo and floor division	left to right
3	+ -	Addition and subtraction	left to right

## 2.3 Selection statement

To illustrate the condition problem, consider the following problems.

### Example 2.6

Suntaree bought a t-shirt for 259 baht and sold the same for 289 baht. Explain how we can find if Suntaree has made a profit or a loss. **Draw a flowchart to find whether Suntaree makes a profit or a loss.** The cost and selling prices are known. To see whether Suntaree makes a profit or a loss, we should ask a yes-no question **“Is a selling price more than a cost price?”** Hence, we can draw a flowchart as shown below.

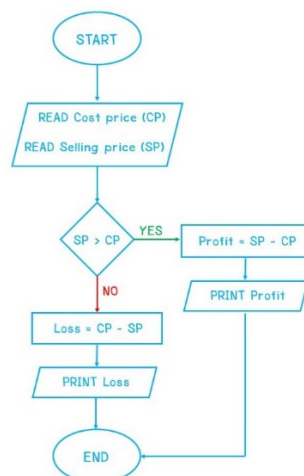


Figure 2.1 Flowchart of the above example

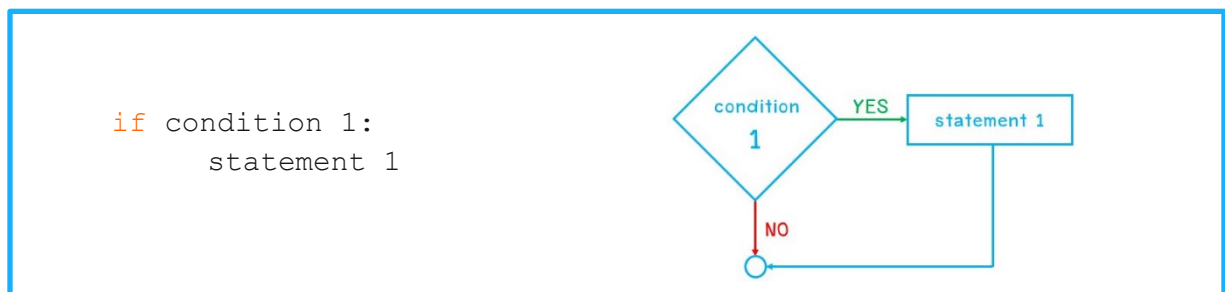
### Python code

```
#Read the value of cost price
cp = eval(input('Enter the cost price (baht):'))

#Read the value of selling price
sp = eval(input('Enter the selling price (baht):'))

if sp > cp:
    profit = sp - cp
    print('Somkid makes profit of ', profit)
```

The above example includes the simplest if statement. In general, the syntax is



### Example 2.7

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 is correct, we use an **if** statement.

### Python code

```
from random import randint

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

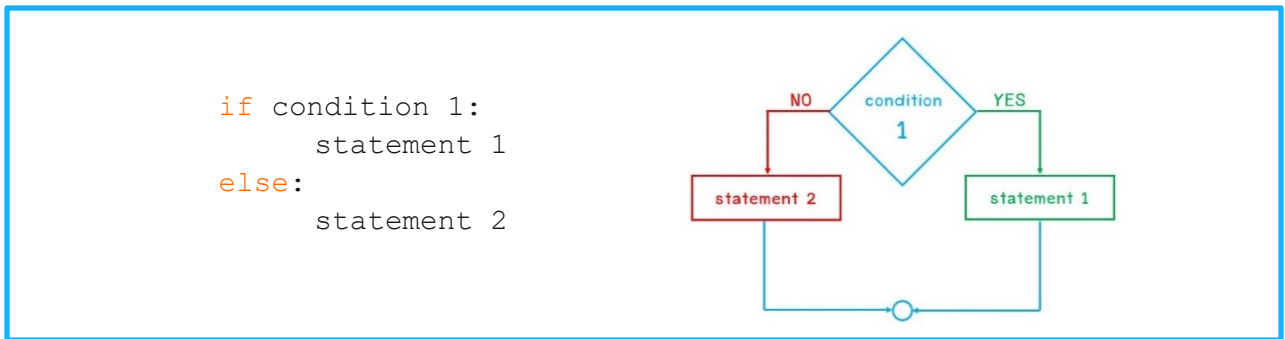
Unfortunately, the above program doesn't show anything if the player guesses wrong. We can adjust the code by adding a few lines.

### Python code

```
from random import randint

num = randint(1,10)
guess = eval(input('Enter your guess: '))
if guess==num:
    print('You got it!')
else:
    print('Sorry. The number is ', num)
```

The corresponding syntax and flowchart of the adjusted code is given below.



### Common Mistakes

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

**Incorrect syntax:** `if x = 1:`

**Correct syntax:** `if x == 1:`

2. A common mistake is to use `and` where `or` is needed or vice-versa.
3. Another very common mistake is to write something as

**Incorrect syntax:** `if grade >= 80 and < 90:`

**Correct syntax:** `if grade >= 80 and grade < 90:`  
`if 80 <= grade < 90:`

### Example 2.8

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.

#### Python code

```

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')
            
```

With the separate `if` statements, each condition is checked regardless of whether it really needs to be. That is, if the score is a 95, the first program will print an A but then continue on and check to see if the score is a B, C, etc., which is a bit of a waste.

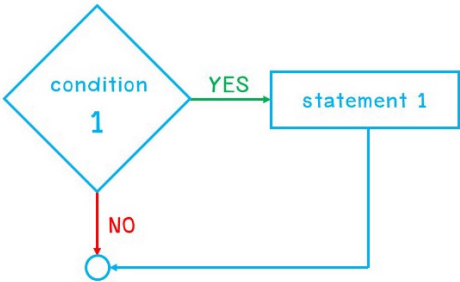
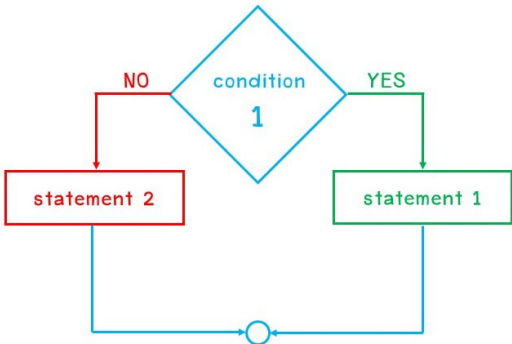
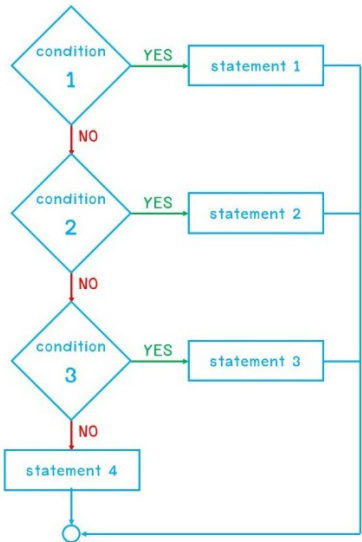
```

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')
            
```

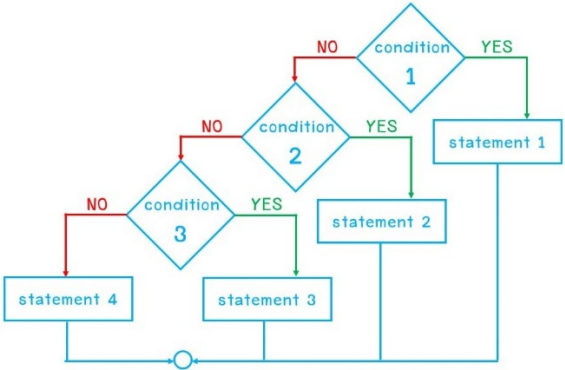
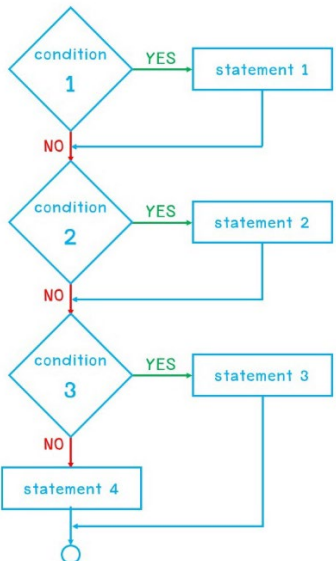
Using `elif`, as soon as we find where the score matches, we stop checking conditions and skip all the way to the end of the whole block of statements.

The following table is a summary of some possible cases of **if** statement.

**Table 2.1** Some possible cases of **if** statement

Flowchart	Syntax
	<pre>if condition 1:     statement 1</pre>
	<pre>if condition 1:     statement 1 else:     statement 2</pre>
	<pre>if condition 1:     statement 1 elif condition 2:     statement 2 elif condition 3:     statement 3 else:     statement 4</pre>

**Table 2.1** Some possible cases of **if** statement (cont)

Flowchart	Syntax
 <pre> graph TD     C1{condition 1} -- YES --&gt; S1[statement 1]     C1 -- NO --&gt; C2{condition 2}     C2 -- YES --&gt; S2[statement 2]     C2 -- NO --&gt; C3{condition 3}     C3 -- YES --&gt; S3[statement 3]     C3 -- NO --&gt; S4[statement 4]     S1 --&gt; Exit(( ))     S2 --&gt; Exit     S3 --&gt; Exit     S4 --&gt; Exit </pre>	<pre> if condition 1:     statement 1 elif condition 2:     statement 2 elif condition 3:     statement 3 else:     statement 4 </pre>
 <pre> graph TD     C1{condition 1} -- YES --&gt; S1[statement 1]     C1 -- NO --&gt; C2{condition 2}     C2 -- YES --&gt; S2[statement 2]     C2 -- NO --&gt; C3{condition 3}     C3 -- YES --&gt; S3[statement 3]     C3 -- NO --&gt; S4[statement 4]     S1 --&gt; Exit(( ))     S2 --&gt; Exit     S3 --&gt; Exit     S4 --&gt; Exit </pre>	<pre> if condition 1:     statement 1 if condition 2:     statement 2 if condition 3:     statement 3 else:     statement 4 </pre>



## 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:
  - If the temperature is less than -273.15, print that the temperature is invalid because it is below absolute zero.
  - If it is exactly -273.15, print that the temperature is absolute 0.
  - If the temperature is between -273.15 and 0, print that the temperature is below freezing.
  - If it is 0, print that the temperature is at the freezing point.
  - If it is between 0 and 100, print that the temperature is in the normal range.
  - If it is 100, print that the temperature is at the boiling point.
  - If it is above 100, print that the temperature is above the boiling point.
3. A year is a leap year if it is divisible by 4, except that years divisible by 100 are not leap years unless they are also divisible by 400. Write a program that asks the user for a year and prints out whether it is a leap year or not.
4. Write a multiplication game program for kids. The program should give the player ten randomly generated multiplication questions to do. After each, the program should tell them whether they got it right or wrong and what the correct answer is.

```
Question 1: 3 x 4 = 12
Right!
Question 2: 8 x 6 = 44
Wrong. The answer is 48.
...
...
Question 10: 7 x 7 = 49
Right.
```

5. Write a program that lets the user play Rock-Paper-Scissors against the computer. There should be five rounds, and after those five rounds, your program should print out who won and lost or that there is a tie.

## References

- <https://realpython.com/python-data-types/#integers>
- Kittipon P, Kittipob P, Somchai P, Sukree S. Python 101. V1.0.2., Chulalongkorn University Printing House; 2018.
- Andrew J. Python: The Ultimate Beginners Guide!., CreateSpace Independent Publishing Platform; 2016.
- Brian H. A Practical Introduction to Python Programming., Crative Commons Attribution-Noncommercial-Share Alike 3.0; 2015.