MCT-242 : COMPUTER PROGRAMMING-I

using Python 3.9

Prepared By:

Mr. Muhammad Ahsan Naeem

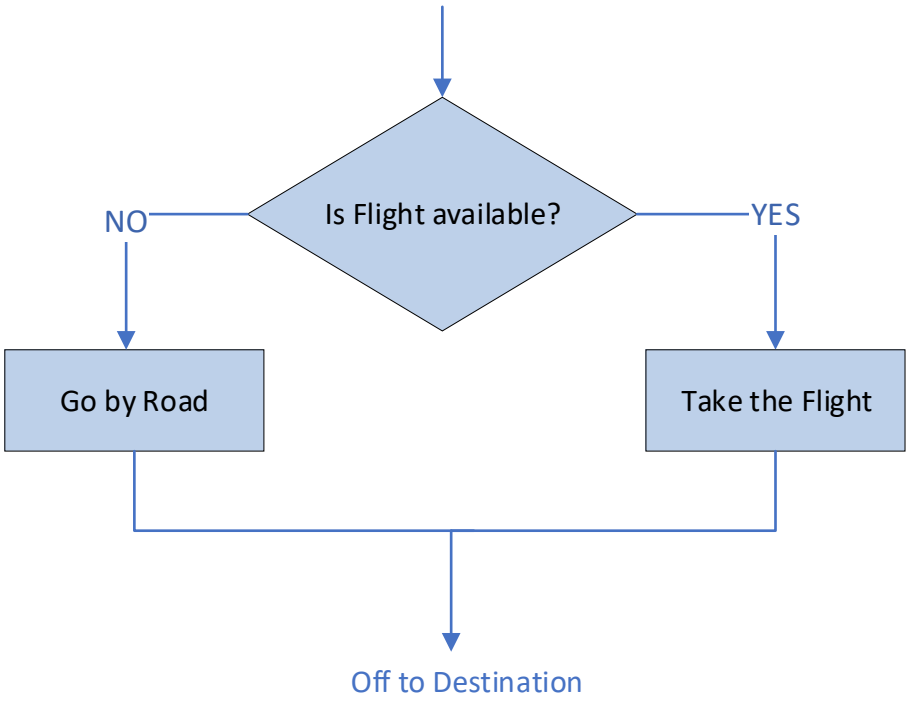
YouTube Playlist

https://www.youtube.com/playlist?list=PLWF9TXck7O_wMDB-VriREZ6EvwkWLNB7q

Lab 6: Nested if-else and elif: CLO 3

We practiced `if` statement and `if-else` statement. In both cases there is a block of code which gets executed based on condition being true or not. The blocks of `if` or `else` can contain single statement or multiple. Those blocks can even contain further `if` or `if-else` statements. It is known as nested `if-else` statements.

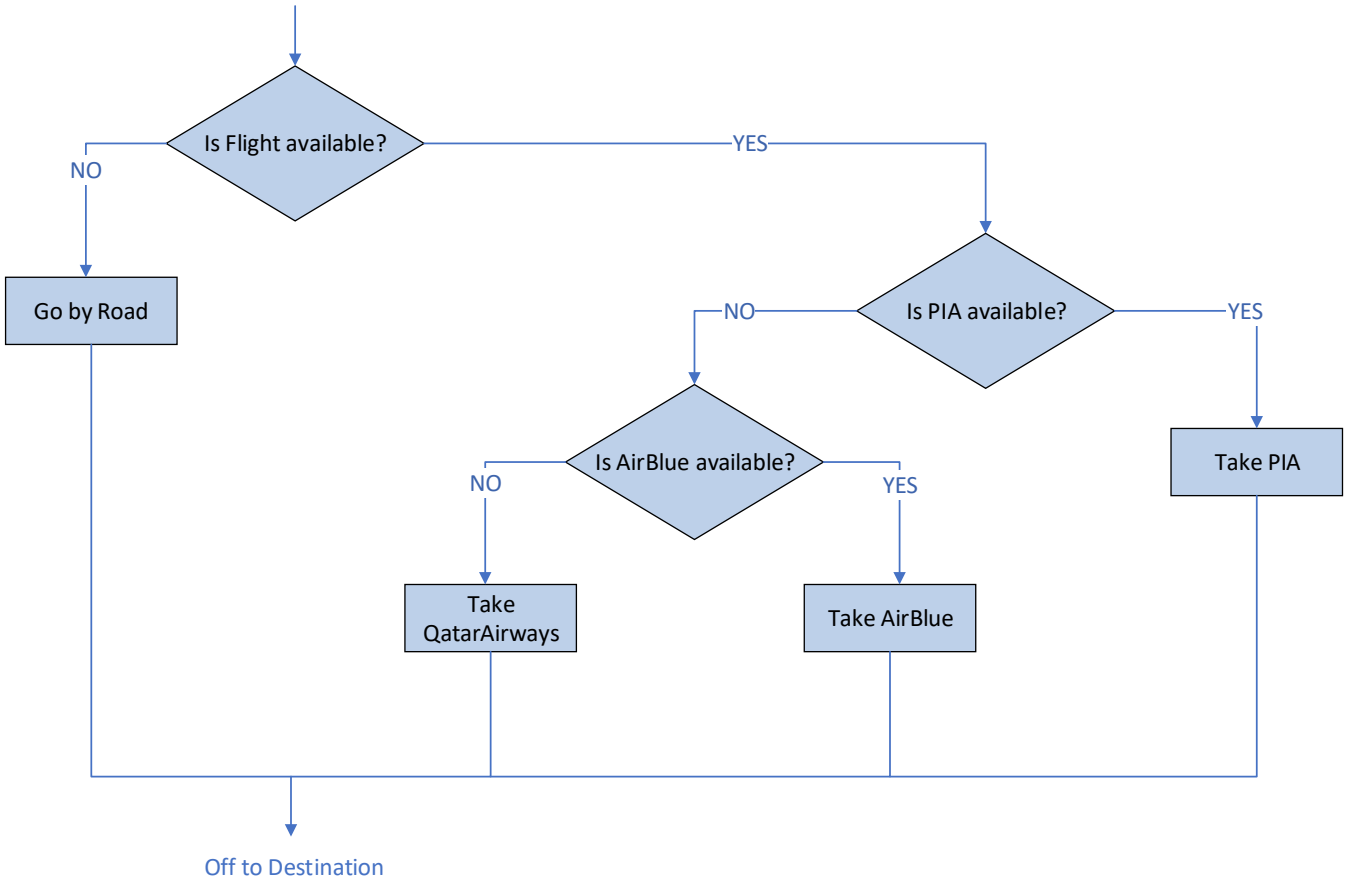
We use similar logic in our daily life decisions, knowingly or not-knowingly. Suppose you need to head to Islamabad on urgent basis. You will check **if** some flight is available and otherwise **(else)** you will go by road. This can be shown in flowchart as:



In Python programming style, this can be written as:

```
if (flight available):
    Take the Flight
else:
    Go by Road
```

In case flight is available, there can be multiple options and multiple decisions based on that. Suppose three airline services are available; **PIA, AirBlue, QatarAirways** *(I strongly wish someday we have situation like that)*. Considering your pocket, you prefer to go by PIA if available, and otherwise you will prefer AirBlue and lastly the QatarAirways. This scheme can be shown in flowchart as:

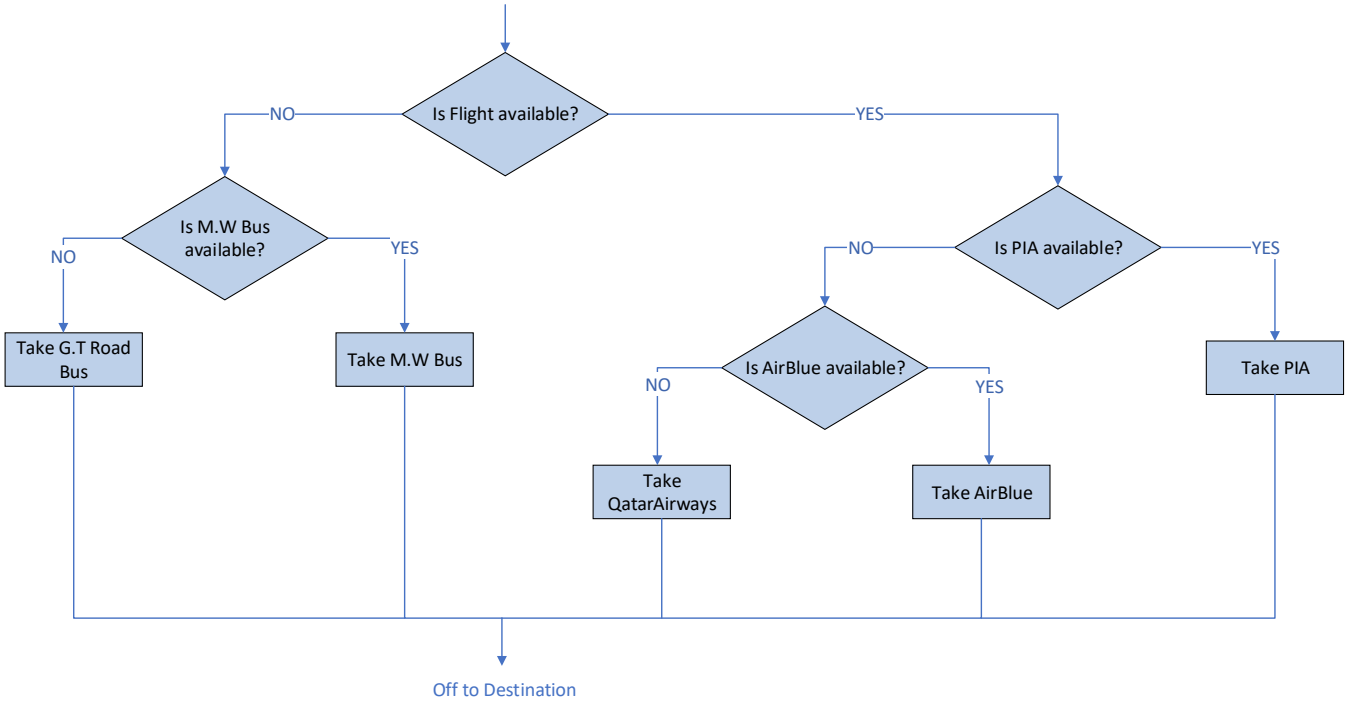


Notice carefully that if flight is not available, the complete right side of the first decision becomes void. In Python programming style, this can be shown as:

```
if (flight available):
    if(PIA available):
        Take PIA
    else:
        if(AirBule Available):
            Take Air Blue
        else:
            Take QatarAirways
else:
    Go by Road
```

See carefully the levels of inner conditions in flowchart and the extended indentation (double or triple tab/spaces) in programming equivalent.

Further, there can be two possibilities to go by road; via G.T Road or via Motor Way. If fight is not available and you prefer to go via Motor Way, the logic is extended as shown:




```
if (filght available):
    if(PIA available):
        Take PIA
    else:
        if(AirBule Available):
            Take Air Blue
        else:
            Take QatarAirways
else:
    if(M.W Bus available):
        Take M.W Bus
    else:
        Take G.T Road Bus
```

Again, see carefully the levels of inner conditions on flow chart and indentation in program.

Example: (Task 9 of last lab session using nested if-else)

Write a program that will take three numbers as input and will print the maximum of three numbers at output.

Suppose that the three numbers are stored in variables **a**, **b** and **c**. Now see this partial complete code:

```
if(a>=b):
    # 'a' is greater than or equal to 'b'
    # Hence 'b' cannot be greatest
    # So, we should compare 'a' and 'c' here
else:
    # 'b' is greater than 'a' here
```

```
    # Hence 'a' cannot be greatest
    # So, we should compare 'b' and 'c' here
```

The logic is simple and explained as comments. The complete logic is here:

```
if(a>=b):
    if(a>=c):
        greatest=a
    else:
        greatest=c
else:
    if(b>=c):
        greatest=b
    else:
        greatest=c
print(str(greatest)+' is the greatest.')
```

Tasks:

[1] Complete the code of above example by taking three numbers from user and printing the number value instead of **a**, **b** or **c**. Secondly, do it with the first comparison between **a** and **c** instead of **a** and **b**.

[2] Complete the task 11 of last lab session using nested **if-else** statement. A starting logic is given at the end:

A company insures its employees in the following cases:

- If the employee is married
- If the employee is unmarried, male & above 30 years of age
- If the employee is unmarried, female & above 25 years of age

Write a program which takes marital status, gender and age as input from user. After checking the given conditions, the output of the program will a message stating whether he/she is eligible for insurance or not.

Sample output is:

```
Enter your marital status (M/U): U
Enter your gender (M/F): F
Enter your age: 28
Congratulations!
You are eligible for the insurance.
```

Another Sample output is:

```
Enter your marital status (M/S): U
Enter your gender (M/F): M
Enter your age: 22
```

```
We are Sorry.
You are not eligible for the insurance.
```

The starting logic for the program is as under:

```
status = input("Enter your marital status (M/U): ")
gender = input("Enter your gender (M/F): ")
age=eval(input("Enter your age: "))
if(status=='M' or status=='m'):
    print('Congratulations! \nYou are eligible\
    for the insurance.')
else:
    if(gender=='M' or gender=='m'):
        # Here employee is Unmarried and Male
        # Write another if-else on his age and display
        # appropriate message
    else:
        # Here employee is Unmarried and Female
        # Write another if-else on his age and display
        # appropriate message
```

[3] Continuing the same previous problem, if the user is married, he/she is eligible for insurance regardless of the gender and age. So, if a person is married, the program should not ask for gender and age.

Sample output is:

```
Enter your marital status (M/U): M
Congratulations!
You are eligible for the insurance.
```

Another Sample output is:

```
Enter your marital status (M/S): U
Enter your gender (M/F): M
Enter your age: 22
We are Sorry.
You are not eligible for the insurance.
```

[4] Write a program that will ask user to enter day and month (e.g. day=7 and month=3). At output the program should display the next day. **We will assume that user will always enter a valid date.**

Sample output is:

```
Enter the day: 7
Enter the month: 3
The next day is : 8
```

Another Sample output is:

```
Enter the day: 30
Enter the month: 4
The next day is : 1
```

Another Sample output is:

```
Enter the day: 30
Enter the month: 3
The next day is : 31
```

Complete the logic given below for this task:

```
d=eval(input("Enter Day: "))
m=eval(input("Enter Month: "))
if(d<=27 or d==29):
    nextDay=d+1
else:
    if(d==28 and m==2):
        nextDay=1
    if(d==28 and m!=2):
        nextDay=29
    if(d==31):
        nextDay=1
    if() # if for d=30 and month with 30 days
    if() # if for d=30 and month with 31 days

print('The next day is : '+str(nextDay))
```

Following points must be noted in above logic:

- The next day generally is entered day plus one except for the cases when it is the last day of a month; then the next day will be 1.
- Last day of a month varies for different months. It can be 28 (For February), 30 (For April, June, September and November) or 31 (For January, March, May, July, August, October and December).
- Based on above rule, if day is less than or equal to 29 but not 28, then the next day is surely current day plus one. Therefore, we have the first **`if`** statement condition.
- The **`else`** of first **`if`** statement will be active when condition is false i.e. current day is 28, 30 or 31. It is handled as described below:
  - First two **`ifs'`** are for 28 days with month being February or other.
  - Third **`if`** is for 31 days and for that the next day is always 1st of next month.
  - Fourth and fifth **`ifs'`** are for 30 days; one for the months with 30 days having the next day as 1 and other for the months with 31 days having the next day as 31.
- After the whole calculations, final display statement is used. You must note that this statement is outside of the **`if`** and **`else`** block.
- Last point to be noted is that in the **`else`** block of the first **`if`** statement, the five **`ifs'`** are not nested within each other rather those five **`ifs'`** are independent. All those five

ifs' will be checked (given the code enters into **else** block) even if the first gets true. But the conditions of those **ifs'** are such that only one can be true.

[5] There are always multiple ways to implement a logic. In above task we started on the basis of days first, followed by the month conditions. There can be another logic for the above program that starts with the month first and then checks for the day. Complete the program given below for the same task:

```
d=eval(input("Enter Day: "))
m=eval(input("Enter Month: "))
if(m==2):
    #Complete the logic for month February
if(m==4 or m==6 or m==9 or m==11):
    #Complete the logic for months with 30 days
if(m==1 or m==3 or m==5 or m==7 or m==8 or m==10 or m==12):
    #Complete the logic for months with 3 days
```

elif statement

We did Task 8 of previous lab session where a certain message was displayed based on average score of a student. The code is written here:

```
s1,s2,s3,s4,s5=eval(input("Enter five subjects' marks: "))
avg=(s1+s2+s3+s4+s5)/5
print('Average of five subject marks is: '+str(avg))
if(avg>=80):
    print('You are an outstanding student.')
if(avg>=70 and avg<80):
    print('You are a good student.')
if(avg>=60 and avg<70):
    print('You are an average student.')
if(avg>=50 and avg<60):
    print('You are a below-average student.')
if(avg>=40 and avg<50):
    print('You are a poor student.')
if(avg<40):
    print('You need extra ordinary efforts.')
```

This can be done in a better way using nested **if-else** statement as shown here:

```
s1,s2,s3,s4,s5=eval(input("Enter five subjects' marks: "))
avg=(s1+s2+s3+s4+s5)/5
print('Average of five subject marks is: '+str(avg))
if(avg>=80):
    print('You are an outstanding student.')
else:
    if(avg>=70):
        print('You are a good student.')
    else:
```

```
        if(avg>=60):
            print('You are an average student.')
        else:
            if(avg>=50):
                print('You are a below-average student.')
            else:
                if(avg>=40):
                    print('You are a poor student.')
                else:
                    print('You need extra ordinary efforts.')
```

See carefully how we don't need to check double conditions as the next conditions are in nested form till the last **else** that will be active only when average is less than 40.

The above code is perfect but you can see that the blocks of the most of the **else** statements starts with another **if** and requires extra indentation. For such cases i.e. where the **else** block starts with an **if**, we can use the **elif** statement as shown here:

```
s1,s2,s3,s4,s5=eval(input("Enter five subjects' marks: "))
avg=(s1+s2+s3+s4+s5)/5
print('Average of five subject marks is: '+str(avg))
if(avg>=80):
    print('You are an outstanding student.')
elif(avg>=70):
    print('You are a good student.')
elif(avg>=60):
    print('You are an average student.')
elif(avg>=50):
    print('You are a below-average student.')
elif(avg>=40):
    print('You are a poor student.')
else:
    print('You need extra ordinary efforts.')
```

Tasks:

[6] Electricity Bill Calculator: Electricity Bills are calculated on the basis of Units (KWhr) used. The LESCO tariffs for domestic users are given as:

Description	Rates
For first 100 Units	13.85 Rs/Unit
101-200 Units	15.86 Rs/Unit
201-300 Units	16.83 Rs/Unit

301-700 Units	18.54 Rs/Unit
Above 700 Units	20.94 Rs/Unit
Neelum Jhelum Surcharge NJ-Sur	0.1 Rs/Unit
Financial Cost Surcharge FC-Sur	0.43 Rs/Unit
TV Charges (Fix)	35 Rs
GST	12% of Calculated amount

Write a program that will ask user to enter number of electricity units consumed and will show the amount of bill at output. LESCO charges extra 8.34% of total amount if bill is not paid within due date. The program should display the amount of bill payable within due date and after due date. Use `elif` instead of `else if`.

Sample output is:

```
Enter Electricity Units consumed: 400
Amount Payable Within Due-Date: 7565.6
Amount Payable After Due-Date: 8196.57
```