LAB 5: Selection

Compound statements

Compound statements takes the following form:

```
<keyword>:          #This is the header and must always end with a colon (:)
    <statement>     # This is the block and must have consistent indentation
    …
    …
    …
```

<keyword> can be :, if, while, for, else, elif, class …

Compound statements should necessarily have

- colon (:)
- consistent indentation

Selection statement (or branching)

Structure of the selection statement are as follows:

Simple if statement:

```
if <condition>:          # must always end with a colon (:)
    <statement>          # can have one or many statements
    …                    # remember the indentation
```

if …else statement (Two way branching):

```
if <condition>:          # must always end with a colon (:)
    <statement>          # can have one or many statements
    …                    # remember the indentation
else:                    # must always end with a colon (:)
    <statement>          # can have one or many statements
    …                    # remember the indentation
```

if …elif statement:

```
if <condition>:          # must always end with a colon (:)
    <statement>          # can have one or many statements
    …                    # remember the indentation
elif <condition>:        # must always end with a colon (:)
    <statement>          # can have one or many statements
    …                    # remember the indentation
```

- **<condition>** is represented using comparison operators and logical expressions.
- The brackets around the conditions are optional.
- **<statement>** will be executed only if the condition is True.
- Indentation is the white space characters added in front of a statement line. In most programming languages, indentation is used mainly to make the code readable. However, *in python, indentation is used to determine if a particular statement belongs together for execution*.

Common errors:

- Missing the : after the condition
- Using a single = instead of == for the comparison
- Different indentation

Exercise 1:

Create a file called **Lab5Greeting.py** and type in the following:

```
gender = input('Please enter your gender: ')

if gender == 'male':
    print('Good Morning Gentleman')
```

This program will ask the user to enter their gender. If the gender is **male** then it will display a message "Good Morning Gentleman".

Run the program and check the results.

```
>>>
Please enter your gender: male
Good Morning Gentleman
>>>
```

Note: if the condition is not met (is False), i.e. if the gender is not **male**, then nothing happens. There are situations where it is desirable to have some actions even when the condition is False. For that, we can use if-else two-way branching.

Modify the **Lab5Greeting.py** file as shown below:

```
gender = input('Please enter your gender: ')

if gender == 'male':
    print('Good Morning Gentleman')
else
    print('Good Morning Gorgeous Lady')
```

In the program above, if the gender does not match male, then it will display “Good Morning Gorgeous Lady”. This may not always be suitable, as the user may have an incorrect spelling for male and gets a message “Good Morning Gorgeous Lady”.

In the program above, if the gender does not match **male**, then it will display “Good Morning Gorgeous Lady”. See sample output shown below:

```
Please enter your gender: male
Good Morning Gentleman
>>> ================================
>>>
Please enter your gender: female
Good Morning Gorgeous Lady
>>> ================================
>>>
Please enter your gender: mail
Good Morning Gorgeous Lady
>>>
```

So, it may be better to make the False condition more specific. Modify the **Lab5Greeting.py** file as shown below:

```
gender = input('Please enter your gender: ')

if gender == 'male':
    print('Good Morning Gentleman')
elif gender == 'female':
    print('Good Morning Gorgeous Lady')
```

Sample output:

```
Please enter your gender: male
Good Morning Gentleman
>>> ================================
>>>
Please enter your gender: female
Good Morning Gorgeous Lady
>>> ================================
>>>
Please enter your gender: mail
>>>
```

Exercise 2:

Create a script called **Lab5Pass.py** and type in the following:

```
mark = eval(input('Please enter the mark: '))

if mark>=50 and mark<=100:
    print('Pass')
    print('Congratulations!!')
else:
    print('Fail')
    print('Please try again')
```

This program will ask the user for a mark and based on the mark entered it will display the result statements. The user will get a message that they passed if the mark entered is between 50 and 100 otherwise, will get the message that they failed.

Note:

1. the if and else keywords must have the same level of indentation
2. the two print statements within if must have the same indentation
3. the two print statements within else must have the same indentation

Sample output:

```
Please enter the mark: 40
Fail
Please try again
>>> ================================
>>>
Please enter the mark: 50
Pass
Congratulations!!
>>> ================================
>>>
Please enter the mark: 60
Pass
Congratulations!!
>>> ================================
>>>
Please enter the mark: 100
Pass
Congratulations!!
>>> ================================
>>>
Please enter the mark: 110
Fail
Please try again
>>>
```

Task 1:

Calculate shipping charges. Ask the user to enter the total of their purchase. If the purchase is $50 or above, the shipping is free. Otherwise, the shipping cost is $10. Your program should display the final total (including shipping).
Please test your program using values under 50, 50, and above 50.

Answer 1:

```
amount = eval(input(_____________________________________________ ))
shipping = 10

if ____________________:
        shipping =0

amount = amount + shipping

print('Total to pay(including shipping): $' + ____________________)
```

Answer 2:

```
amount = eval(input(                                                    ))

if ____________________:
        shipping =0
else:
        shipping =10

amount = amount + shipping

print('Total to pay(including shipping): $' + ____________________)
```

Create a script called **Lab5Task1.py** and enter the above code. Run the script and check the results.

Sample output:

```
Please enter the total amount: $100
Total to pay(including shipping): $100
>>> ================================ RESTART
>>>
Please enter the total amount: $40
Total to pay(including shipping): $50
>>> ================================ RESTART
>>>
Please enter the total amount: $50
Total to pay(including shipping): $50
>>> ================================ RESTART
>>>
Please enter the total amount: $49
Total to pay(including shipping): $59
>>>
```

Nested ifs

Besides two-way branching, we have multi-way branching we can use to deal with situations that are more complex. There are two approaches to writing multi-way branching, **nested ifs** and **elif** statement.

There are often times when an *if statement* inside another if statement is needed to accomplish the logic we want. We call them 'nested if statements'.

Nested if statement:

```
if <condition>:              # must always end with a colon (:)
    <statement>              # can have one or many statements
    …
else:
    if <condition>:          # remember the indentation
        <statement>
        ….
    else
        if <condition>:
            <statement>
            ….
```

Note: If the condition is True, then none of the statements in the *else* block will be executed.

One of the drawbacks of nested if statements is that, when the level of nesting gets deeper, so is the indentation, which makes it sometimes hard to manage.

Task 2:

Extend the temperature conversion program (Temperature.py, Lab 3 Task 6) to include user choice of conversion. Save that script as **Lab5T2.py**. The program will ask the user what they are converting and the temperature. The program should convert the entered temperature to the other unit (i.e. C is converted to F, and F is converted to C).

F = (9/5)*C + 32 C = (F-32) *(5/9)

Answer:

```
unit = input(_____)

temp = eval(input(_____)

if unit == 'C':
    newTemp =
else:
    if unit == 'F':
        newTemp =
print(_____)
```

elif header

An alternative form denoting multi-way selection in python is the **elif** header. This form is more concise and allows you to avoid deeply nested levels of indentation.

```
'''
Author:
Purpose: To convert the given temperature to the other unit
'''

# Get user input
unit=input('What unit are you entering in? (C or F):')
unit=unit.upper()        # will convert to uppercase
temp = eval(input('Please enter the temperature: '))

# Use if statement and convert the temperature
if unit =='C':
    newTemp = (9/5)*temp + 32
    newUnit = 'F'
elif unit == 'F':
    newTemp = (temp-32)*(5/9)
    newUnit = 'C'

# display output
print("")
print(temp, unit, 'is converted to', newTemp, newUnit)
```

Sample output:

```
>>>
What unit are you entering in? (C or F):c
Please enter the temperature: 100

100 C is converted to 212.0 F
>>> ================================ RESTART ============
>>>
What unit are you entering in? (C or F):F
Please enter the temperature: 212

212 F is converted to 100.0 C
>>>
```

Logically equivalent Boolean expressions

A Boolean expression can be written in more than one way. Consider the following:

num !=0	not(num == 0)	
(num !=0) and (num !=6)	not (num ==0 or num ==6)	
num >=0 and num <=6	(not num <0) and (not num >6)	not (num < 0 or num > 6)
num < 0 or num > 6	(not num >=0) and (not num <=6)	not (num <0 or num >6)

Task 3:

1. Which two of the following Boolean expressions are logically equivalent?

   a) not (num < 0 or num >10)
   b) num > 0 and num < 10
   c) num >=0 and num <= 10

Answer: a and c (option b misses out the boundary values 0 and 10)

2. Write the logically equivalent expressions of the following:

   a) mark >= 50 and mark <= 100 _____

   b) age >=18 or wintecStudent = 'Yes' _____

Task 4:

Create a new script called Lab5T4.py. This program will calculate the grade given a mark. Type the following:

```
# Get user input
mark=eval(input('Please enter the mark:'))

if mark >=90:
    grade = 'A'
elif mark >=80:
    grade = 'B'
elif mark>=70:
    grade = 'C'
elif mark>=60:
    grade = 'D'
else:
    grade = 'E'

print('Your grade is an', grade)
```

The same can also be done using if and else statements as follows:

```
# Get user input
mark=eval(input('Please enter the mark:'))

if mark >=90:
    grade = 'A'
else:
    if mark >=80:
        grade = 'B'
    else:
        if mark>=70:
            grade = 'C'
        else:
            if mark>=60:
                grade = 'D'
            else:
                grade = 'E'

print('Your grade is an', grade)
```

What's critical with this form is that, the conditions must be in order, either going up or coming down, but can't be interlacing or random. For example, the sequence should be either >=90, >=80, >=70, >=60 or it should be <60, < 70, < 80, < 90.

The following solution is problematic (This is because, for an **elif** statement, as soon as a condition is found true the rest are no longer checked)

```
if mark >=70:
    grade = 'C'
elif mark >=80:
    grade = 'B'
elif mark>=90:
    grade = 'A'
elif mark>=60:
    grade = 'D'
else:
    grade = 'E'
```

Task 5:

Write a program called Lab5T5.py to ask from user to enter the name of their city. Print a comment about the city: Auckland crowded, Wellington windy, Hamilton cosy, any other city mysterious.

```
# Get user input
city=input('Please enter the city:')

# will format as title no matter how the user enter the city
city = city.title()

if city == 'Auckland':
    message= 'Crowded'
elif city == 'Wellington':
    message= 'Windy'
elif city == 'Hamilton':
    message= 'Cozy'
else:
    message= 'Mysterious'

print(city, message)
```

Bonus Task: Attempt the same using nested if statements. Show it to your tutor when you finish.

Task 6:

Write a program that asks the user to enter a choice of 'C', 'R', or 'S' and display a message of 'Circle', 'Rectangle', or 'Square' respectively. For any other input, display 'Invalid choice'. Write two versions of your solution, one using nested if statements, and one using elif header.

Task 7:

Write a program that determines if an individual qualifies for a government first-time home buyer credit. The credit is only available to those who (1) buy a house cost less than 500000, (2) family combined income under 100000, (3) had not owned a primary residence in the last three years.

Sample output

```
>>>
Please enter the cost of the house:400000
Please enter your family income: 50000
Have you owned a house before? (Y or N):n
Congratulations!! You are eligible for the credit.
>>> ================================ RESTART ================
>>>
Please enter the cost of the house:550000
Please enter your family income: 90000
Have you owned a house before? (Y or N):n
Sorry ... You are not eligible for the credit.
>>>
```

Error Checks

Sometimes we expect certain values. For example, in the temperature program, we would expect the user to enter C or F. However, accidentally or intentionally, it is possible that the user will enter another value. To cope with such situations better, it is nice to have some code in the script that will let the user know that there is an unexpected value.

For example, the temperature program (Lab4T2.py) can be modified as follows:

```
# Get user input
unit=input('What unit are you entering in? (C or F):')
unit=unit.upper()        # will convert to uppercase

# Error check and convert the temperature
if unit != 'C' and unit != 'F':
    print('Invalid choice')
else:
    temp = eval(input('Please enter the temperature: '))
    if unit =='C':
        newTemp = (9/5)*temp + 32
        print(temp, 'is C is', newTemp, 'in F')
    else:
        newTemp = (temp-32)*(5/9)
        print(temp, 'is F is', newTemp, 'in C')
```

Sample output

```
>>>
What unit are you entering in? (C or F):d
Invalid choice
>>> ================================ RESTART ===
>>>
What unit are you entering in? (C or F):c
Please enter the temperature: 100
100 is C is 212.0 in F
>>> ================================ RESTART ===
>>>
What unit are you entering in? (C or F):f
Please enter the temperature: 212
212 is F is 100.0 in C
>>>
```

The above nested selection statements can be re-written by using if-elif-else statement, as follows:

```
if unit =='C':
   newTemp = (9/5)*temp + 32
   print(temp, 'is C is', newTemp, 'in F')
elif unit == 'F':
   newTemp = (temp-32)*(5/9)
   print(temp, 'is F is', newTemp, 'in C')
else:
   print('Invalid Choice')
```