Module 4 Lab: Decisions and Boolean Logic

This lab accompanies Chapter 3 of *Intro to Python for Computer Science and Data Science*

Module 4 Lab Part 1 –Logical Operators and Dual Alternative Decisions

Critical Review

The logical AND operator and the logical OR operator allow you to connect multiple Boolean expressions to create a compound expression.

The logical NOT operator reverses the truth of a Boolean expression.

When using the AND operator, both conditions must be true in order for the statements within an if to process.

When using the OR operator, either condition must be true in order for the statements within an if to process.

A dual alternative decision structure will execute one group of statements if its Boolean expression is true, or another group if its Boolean expression is false.

The general structure of an `if-then-else` statement is

```
If condition Then
   Statement
   Statement
   Etc.
Else
   Statement
   Statement
   Etc.
End If
```

This lab requires you to think about possible true and false conditions using `if` statements.

Step 1: Consider the following values set to variables.

- `myAge = 32`
- `yourAge = 18`
- `myNumber = 81`
- `yourNumber = 17`
- `votingAge = 18`

Step 2: Based on the values to the variables in Step 1, what is the expected output? Hint: The output will be either what is printed to the screen, or nothing. (Reference: Logical Operators, page 189).

The condition	Expected Output
If myAge == 31 AND yourAge < myAge Then Display "My age is 31 and your age is less than that" End If	Nothing ▾
If myAge <= 35 AND myAge >= 32 Then Display "My age is between 32 and 35" End If	My age is between 32 and 3 ▾
If yourAge == votingAge OR yourAge > votingAge Then Display "You can vote" End If	You can vote ▾
If myNumber == 83 OR yourNumber == 83 Then Display "One of our numbers is 83" End If	Nothing ▾

Step 3: Based on the values to the variables in Step 1, what is the expected output? (Reference: Dual Alternative Decision Structures, page 167).

The condition	Expected Output
If myAge == 31 AND yourAge < myAge Then Display "My age is 31 and your age is less than that" Else Display "Our ages do not qualify" End If	Our ages do not qualify ▾
If myAge <= 35 AND myAge >= 32 Then Display "My age is between 32 and 35" Else Display "My age is not within that range" End If	My age is between 32 and 3 ▾
If yourAge == votingAge OR yourAge > votingAge Then Display "You can vote" Else Display "You cannot vote" End If	You can vote ▾
If myNumber == 83 OR yourNumber == 83 Then Display "One of our numbers is 83" Else Display "83 is not our number" End If	83 is not our number ▾

Module 4 Lab Part 2 – Pseudocode: Dual Alternative Decisions

Critical Review

A dual alternative decision structure will execute one group of statements if its Boolean expression is true, or another group if its Boolean expression is false.

The general structure of an `if-then-else` statement is:

```
If condition Then
     Statement
     Statement
     Etc.
Else
     Statement
     Statement
     Etc.
End If
```

This lab requires you to think about the steps that take place in a program but write actual python code by filling in the blanks.

Recall the retail company program from Lab 3.2. The company now wants to modify their bonus portion to include different levels and types and eliminate the day off program. The new program description is as follows:

```
A retail company assigns a $6,000 store bonus if monthly sales
are more than $110,000; else if monthly sales are greater than or
equal to $100,000 the store bonus is $5,000, else if monthly
sales are greater than or equal to $90,000 the store bonus is
$4,000, else if monthly sales are greater than or equal to
$80,000, the store bonus is $3,000 otherwise a $0 amount or no
store bonus is awarded.  They are using a percent of sales
increase to determine if employees get individual bonuses.  If
sales increased by an amount greater than or equal to 5% (0.05)
then all employees get $75, else if sales increased by an amount
greater than or equal to 4%, employees get $50, else if sales
increased by an amount greater than or equal to 3% employees get
$40 otherwise they get $0.
```

Step 1: To accommodate the changes to the program, create the additional variables needed.

- Create a variable named `storeAmount` to hold the store bonus amount.
- Create a variable named `empAmount` to hold the individual bonus amount.
- Create a variable named `salesIncrease` to hold the percent of increase.

This is the main function that you need to fill in the blanks with the variable names listed above:

```
def main():
     # declare local variables
     monthlySales    =   # monthly sales amount
     storeAmount     =   # store bonus amount
     empAmount       =   # employee bonus amount
     salesIncrease   =   # percent of sales increase

     # call to getSales(monthlySales)
     # call to getIncrease(salesIncrease)
     # call to calcStoreBonus(storeAmount)
     # call to calcEmpBonus(empAmount)
     # call to printBonus(storeAmount, empAmount)
```

Step 2: The first function in the program is `getSales()`. Since this is required, leave this function as is. This function should be written as follows:

```
# This function gets the monthly sales
def getSales(prompt):
   monthlySales = float(input(prompt))
   return monthlySales
```

Step 3: The next function in the program is `calcStoreBonus(monthlySales)`. Write an `if-then-else` statement within this module that will set the bonus amounts to those in the description at the beginning. Return the `storeAmount` to the calling statement. Complete the missing lines. (Reference: Dual Alternative Decision Structures, page 167).

```
# This function determines the storeAmount bonus
def calcStoreBonus(monthlySales):
     if monthlySales >= 110000:
          storeAmount = 6000
     elif monthlySales >= 100000:
          storeAmount = 5000
     elif monthlySales >= 90000:
          storeAmount = 4000
     elif monthlySales >= 80000:
          storeAmount = 3000
     else:storeAmount = 0

     return storeAmount
```

Step 4: Write a function that will ask the user to enter the percent of sales increase in decimal format. This module returns `salesIncrease` to the calling statement. Complete the missing lines.

```
# This function gets the percent of increase in sales
def getIncrease(prompt):
     salesIncrease = float(input("Enter the percent of sales increase in))
     salesIncrease = salesIncrease / 100
     return salesIncrease
```

Step 5: Write a function that will determine individual bonuses. Use the values given in the description at the beginning. This module should be called calc`EmpBonus` and has `salesIncrease` as a single parameter. Return `empAmount` to the calling statement. Fill in the blanks.

```
     # This function determines the empAmount bonus
     def calcEmpBonus(salesIncrease):
          if salesIncrease >= .05:
               empAmount = 75
          elif salesIncrease >= .04 :
               empAmount = 50
          elif salesIncrease >= .03 :
               empAmount = 40
          else:
               empAmount = 0
          return empAmount
```

Step 6: Write a function that will print the store bonus and the employee bonus amount. Name this function `printBonus()` and pass the two necessary variables.

```
     # This function prints the bonus information
     def printBonus(storeAmount ,empAmount ):
        print("The store bonus amount is $",storeAmouint )
        print("The employee bonus amount is $",empAmount)
        if (storeAmount == 6000 )AND (empAmount ==75 ):
          print('Congrats! You have reached the highest bonus amounts
possible! ')
```

Module 4 Lab Part 3 – Pseudocode: Nested Decision Structures

Critical Review

To test more than one condition, a decision structure can be nested inside another decision structure. This structure can become very complex, and often an `if-then-else-if` statement is used instead.

The general structure of the `if-then-else-if` statement is:

```
If condition_1 Then
     Statement
     Statement
     Etc.
Else If condition_2 Then
     Statement
     Statement
     Etc.
Insert as many Else If clauses as necessary
Else
```

```
            Statement
            Statement
            Etc.
        End If
```

A case structure lets the value of a variable or an expression determine which path of execution the program will take. This is often used as an alternative to a nested if else decision.

Module 4 Lab Part 4 – Python Code

Step 1: Start Visual Studio Code. Prior to entering code, save your file by clicking on File and then Save. Select your location and save this file as *Lab4.py*. Be sure to include the .py extension.

Step 2: Document the first few lines of your program to include your name, the date, and a brief description of what the program does.

Step 3: Start your program with the following code and fill in the ??? and replace the various functions listed here with the functions you completed in the previous steps.

```
# Module 5 Lab-5
# Add your name here
# Add the date here
# Describe what the program does here

# The main function
def main():
     # declare local variables
     monthlySales    =   # monthly sales amount
     storeAmount     =   # store bonus amount
     empAmount       =   # employee bonus amount
     salesIncrease   =   # percent of sales increase

     # call to getSales(monthlySales)
     # call to getIncrease(salesIncrease)
     # call to calcStoreBonus(storeAmount)
     # call to calcEmpBonus(empAmount)
     # call to printBonus(storeAmount,empAmount)
# This function gets the monthly sales
def getSales(prompt):
    monthlySales = float(input(prompt))
    return monthlySales

# This function gets the percent of increase in sales
def getIncrease(prompt):
     salesIncrease = float(input("Enter the percent of sales increase in ))
     salesIncrease = salesIncrease / 100
     return # This function determines the storeAmount bonus
            salesIncrease
def calcStoreBonus(monthlySales):
     if monthlySales >= 110000:
           storeAmount = 6000
     elif monthlySales >= 100000 :
           storeAmount = 5000
     elif monthlySales >= 90000 :
           storeAmount = 4000
     elif monthlySales >= 80000 :
           storeAmount = 3000
     else:
           storeAmount = 0
     return storeAmount

     # This function determines the empAmount bonus
     def calcEmpBonus(salesIncrease):
           if salesIncrease >= .05:
                empAmount = 75
           elif salesIncrease >= .04:
                empAmount = 50
           elif salesIncrease >= .03:
                empAmount = 40
           else:
                empAmount = 0
           return empAmount

# This function prints the bonus information
def printBonus(storeAmount ,empAmount ):
     print("The store bonus amount is $",storeAmouint)
     print("The employee bonus amount is $",empAmount)
     if (storeAmount == 6000 )AND (empAmount ==75 ):
           print('Congrats! You have reached the highest bonus amounts
possible! ')

# calls main
main()
```

Step 4: Click Run to see how your program processes. Test the following values to verify the expected output.

Input Values	**Expected Output**

monthlySales = 120500
salesIncrease = 5
	The store bonus amount is $ 6000
	The employee bonus amount is $ 75
	Congrats! You have reached the highest bonus amounts possible!

monthlySales = 93400
salesIncrease = 5
	The store bonus amount is $4000
	The employee bonus amount is $75

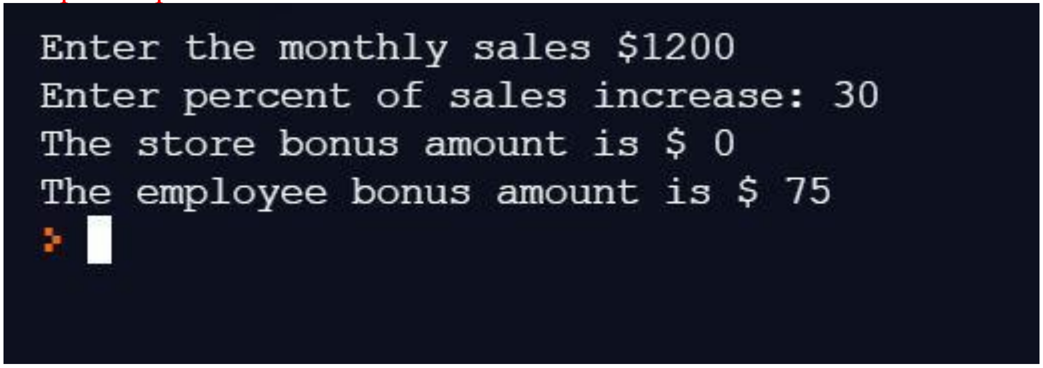
monthlySales = 75000
salesIncrease = 1.5
	The store bonus amount is $0
	The employee bonus amount is $0

monthlySales = 82000
salesIncrease = 3.6
	The store bonus amount is $3000
	The employee bonus amount is $40

monthlySales = 125000
salesIncrease = 4.5
	The store bonus amount is $6000
	The employee bonus amount is $50

Step 5: **Submit this completed word document and .py source code file to D2L.**

Sample Output/result should look like:



```
Enter the monthly sales $1200
Enter percent of sales increase: 30
The store bonus amount is $ 0
The employee bonus amount is $ 75
>
```