

MIB Coding Fundamentals – ITLB351

*- Individual Test -*

Carlos Andres Appelgren Valenzuela

**Autumn 2020/21 – Version A Student’s Name**

## **Time allowed: 90 minutes**

# Command-line exercise

Take a look at this directory structure:

computer-shop

|--desktop

| |--.git

| |--instock

| | └--summary.csv

| └--in-transit

| └--summary.csv

└--laptop

|--in-stock

└--in-transit

└--summary.csv

Your task is to write commands in the correct order from the directory given below.

Do it with fewer commands without chaining them together.

Assume that the following file is currently in the staging area:

* desktop/instock/summary.csv
* desktop/in-transit/summary.csv

Your current directory is computer-shop/

* Remove every file from the staging area
* Rename desktop/instock/ to desktop/in-stock/
* Commit the changes
* Change the laptop directory into a Git repository
* Get the first 5 lines from desktop/in-stock/summary.csv and redirect it to laptop/in-stock/summary.csv

Your commands:

git reset

mv desktop/instock/ desktop/in-stock/

git commit -a -m "rename instock to in-stock"

git init ./laptop/

head -5 desktop/in-stock/summary.csv > laptop/in-stock/summary.csv

# Question 1

What is the staging in Git?

The staging area is a step just before you commit your files to your local repository, this way git knows what are the changes that are happening between the current commit and the next one.

# Question 2

Explain what is the DRY principle in programming

The principal is for you to not repeat yourself, to reduce code duplication which could lead to people having difficulties reading your code and in worse of the cases poor maintenance or refactoring which could lead your program to bug.

# Question 3

What is the difference between syntax and semantic errors?

The difference between them is during the time that it occurs, for example syntax error occurs during the parsing of the input code and is cause by grammatically incorrect statements. While semantic errors occur during the execution of the code, after it has been interpreted as grammatically correct.

# Question 4

What is the purpose of the try-expect block, why do we use it?

Try-except, is a way to handle errors. The code in the try blocks will be executed and if an error is raised it will execute the code in the except block.

# Question 5

What is the output of the following program? Explain your answer!

1. a = True
2. b = False
3. c = False
5. **if** **not** a **or** b:
6. **print** 1
7. **elif** **not** a **or** **not** b **and** c:
8. **print** 2
9. **elif** **not** a **or** b **or** (**not** b **and** a):
10. **print** 3
11. **else**:
12. **print** 4

The output is 3, since the “elif not a or b or (not b and a):” condition is true.   
The first if condition is not true due to both 'not a' and b being false, the second condition is also not true since it contains and c, where c is false. The third condition is true and therefore is where the output comes from.

# Question 6

What is the output of the following program? Explain your answer!

1. counter = {}
3. **def** addToCounter(country):
4. **if** country **in** counter:
5. counter[country] += 1
6. **else**:
7. counter[country] = 1
9. addToCounter('China')
10. addToCounter('Japan')
11. addToCounter('china')
13. **print** len(counter)

The answer is 3, since there are 3 countries added to the counter variable. The addToCounter function is called 3 times with 3 different variables, as a result the length of the counter becomes 3 and is printed.

# Question 7

What is the output of the following program? Explain your answer!

1. numbers = [1, 2, 3, 4]
3. numbers.append([5,6,7,8])
4. **print** len(numbers)

The output is 5, since we append to the original numbers list one more list element, the total length becomes 5.

# Question 8

What is the output of the following program? Explain your answer!

1. **def** addToList(listcontainer):
2. listcontainer += [10]
4. mylistContainer = [10, 20, 30, 40]
5. addToList(mylistContainer)
6. **print** len(mylistContainer)

The output is 5 since we add one more element to the original list of 4 thus the length of the “mylistContainer” variable becomes 5

# Question 9

What is the output of the following program? Explain your answer!

1. list1 = ['green', 'fox', 2015, 2020]
2. list2 = [1, 2, 3, 4, 5, 6, 7]
4. **print** list1[-3]
5. **print** list2[1:]
6. **print** list1[1][1]

There are 3 "outputs" since there are 3 prints, the first will print the string "fox" since negative list indexes are counted from the ending of the list so the position list1[-3] represents the third element from the end of the list. The second output will print all from the list2 position 1 to the end of the list so the output is [2, 3, 4, 5, 6, 7]. The third output will print the letter o, since list1[1][1] represents the second element of the list and the second element of that resulting list(in our case a string) thus the letter o is printed

# Question 10

Which of these collections defines a LIST?

1. {"apple", "banana", "cherry"}
2. ("apple", "banana", "cherry")
3. ["apple", "banana", "cherry"]
4. {"name": "apple", "color": "green"}

The third one represents a list.

OVERVIEW OF MARKS AVAILABLE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Command-line exercise** | **Question 1** | **Question 2** | **Question 3** | **Question 4** |
| **30 points** | **10 points** | **10 points** | **10 points** | **10 points** |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Question 5** | **Question 6** | **Question 7** | **Question 8** | **Question 9** | **Question 9** | **TOTAL** |
| **5 points** | **5 points** | **5 points** | **5 points** | **5 points** | **5 points** | **100 points** |

|  |
| --- |
| ***Overall feedback comments***: |