Syrian Arab Republic

Lattakia

Tishreen University

Department of Communication and electrical engineering

5 th , Network Programming : Homework No1

الجمهورية العربية السورية

وزارة التعليم العالي والبحث العلمي

جامعة تشرين

كلية الهمك

قسم هندسة الاتصالات والالكترونيات

وظيفة برمجة الشبكات

إعداد الطالبة:

دعاء محمد جمعه زينب

2499

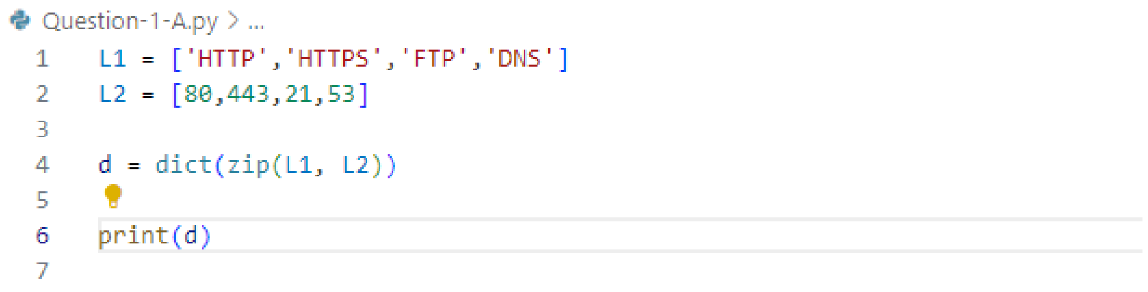
إشراف:

الدكتور المهندس:

مهند عيسى

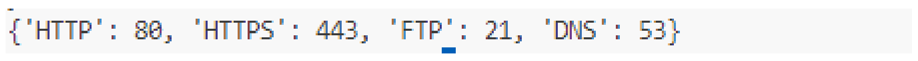
Question 1: Python Basics?

A-If you have two lists, L1=[‘HTTP’,’HTTPS’,’FTP’,’DNS’] L2=[80,443,21,53], convert it to generate this dictionary d={‘HTTP’:80,’HTTPS’:443,’FTP’:21,’DNS’:53 }



Question-1-A.py > ...

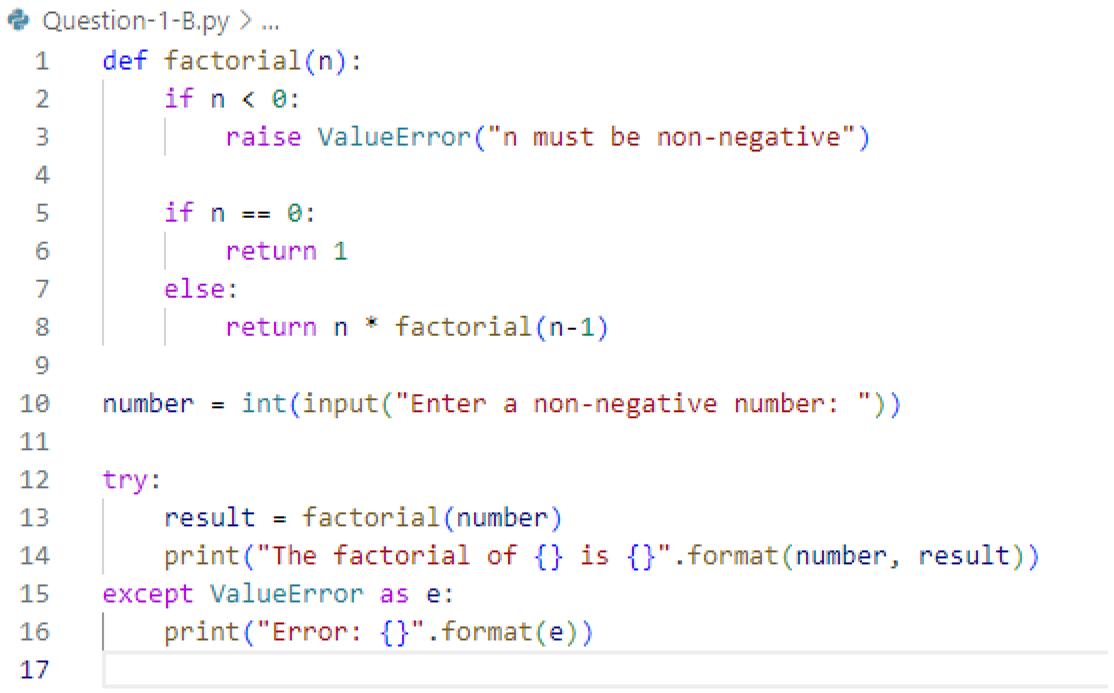
```python
L1 = ['HTTP','HTTPS','FTP','DNS']
L2 = [80,443,21,53]

d = dict(zip(L1, L2))

print(d)

```



```
{'HTTP': 80, 'HTTPS': 443, 'FTP': 21, 'DNS': 53}
```

استخدمت التابع dict() لبناء القاموس d تم تمرير التابع zip() كبارمتر داخل التابع وذلك لمقابلة قيم القائمتين حسب الفهارس، ومن ثم عن طريق الباني dict() حولنا النتيجة لقاموس.

B- Write a Python program that calculates the factorial of a given number entered by user.



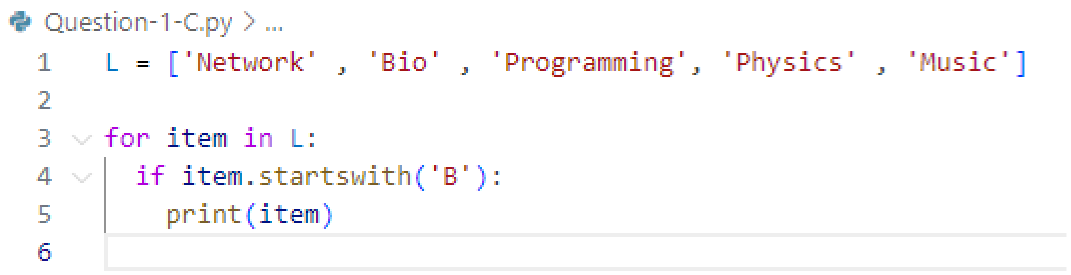
Question-1-B.py > ...

```python
def factorial(n):
    if n < 0:
        raise ValueError("n must be non-negative")

    if n == 0:
        return 1
    else:
        return n * factorial(n-1)

number = int(input("Enter a non-negative number: "))

try:
    result = factorial(number)
    print("The factorial of {} is {}".format(number, result))
except ValueError as e:
    print("Error: {}".format(e))

```

```
Enter a non-negative number: 6
The factorial of 6 is 720
```

تم بناء التابع factorial() لحساب عاملي العدد المدخل ثم باستخدام مفهوم العودية تم حساب العاملي وباستخدام بنية try- except تم معالجة الاستثناء الحاصل في حال إدخال عدد سالب.

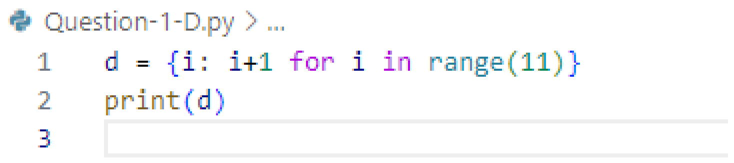
C- L=[‘Network’ , ’Bio’ , ’Programming’, ‘Physics’ , ‘Music’] In this exercise, you will implement a Python program that reads the items of the previous list and identifies the items that starts with ‘B’ letter, then print it on screen. Tips: using loop, ‘len ()’ , startswith() methods.


Question-1-C.py > ...
```python
L = ['Network' , 'Bio' , 'Programming', 'Physics' , 'Music']

for item in L:
  if item.startswith('B'):
    print(item)

```

```
Bio
```

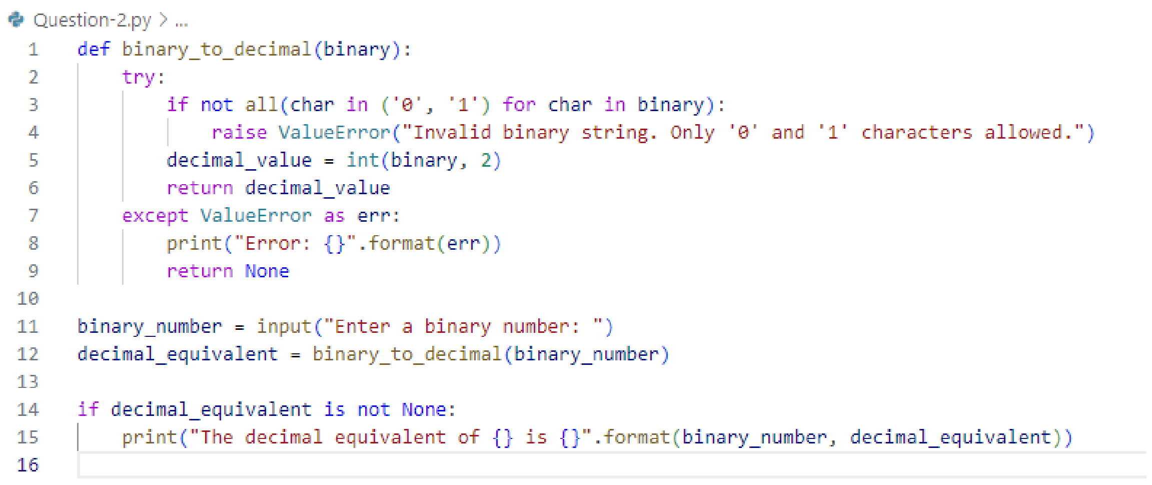
قمت بتعريف القائمة L وباستخدام الحلقة for قمنا بالمرور على عناصر القائمة واختبار بواسطة الميثود startswith() الكلمات التي تبدأ ب B.

D: Using Dictionary comprehension, Generate this dictionary d={0:1,1:2,2:3,3:4,4:5,5:6,6:7,7:8,8:9,9:10,10:11}


Question-1-D.py > ...
```python
d = {i: i+1 for i in range(11)}
print(d)

```

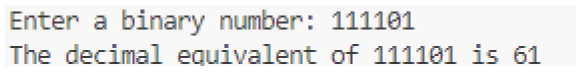
```
{0: 1, 1: 2, 2: 3, 3: 4, 4: 5, 5: 6, 6: 7, 7: 8, 8: 9, 9: 10, 10: 11}
```

<u>Question 2:</u> Convert from Binary to Decimal Write a Python program that converts a Binary number into its equivalent Decimal number. The program should start reading the binary number from the user. Then the decimal equivalent number must be calculated. Finally, the program must display the equivalent decimal number on the screen. Tips: solve input errors.



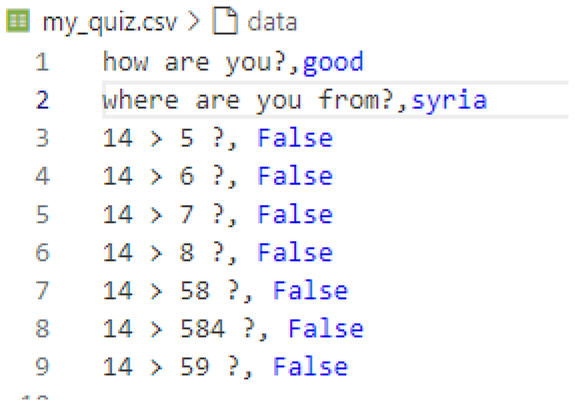
Question-2.py > ...

```python
def binary_to_decimal(binary):
    try:
        if not all(char in ('0', '1') for char in binary):
            raise ValueError("Invalid binary string. Only '0' and '1' characters allowed.")
        decimal_value = int(binary, 2)
        return decimal_value
    except ValueError as err:
        print("Error: {}".format(err))
        return None

binary_number = input("Enter a binary number: ")
decimal_equivalent = binary_to_decimal(binary_number)

if decimal_equivalent is not None:
    print("The decimal equivalent of {} is {}".format(binary_number, decimal_equivalent))

```



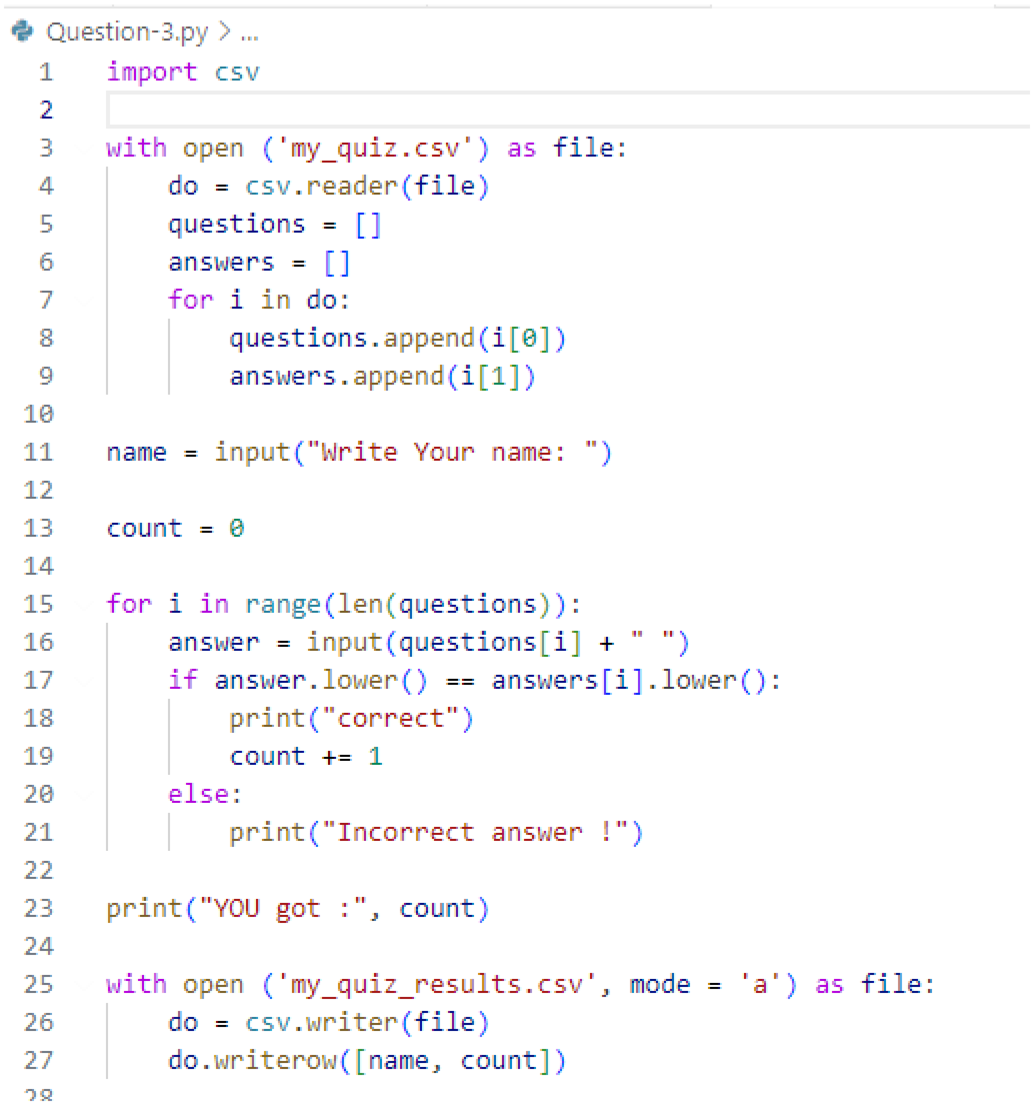
```
Enter a binary number: 111101
The decimal equivalent of 111101 is 61
```

<u>Question 3:</u> Working with Files" Quiz Program" Type python quiz program that takes a text or json or csv file as input for (20 (Questions, Answers)). It asks the questions and finally computes and prints user results and store user name and result in separate file csv or json file.



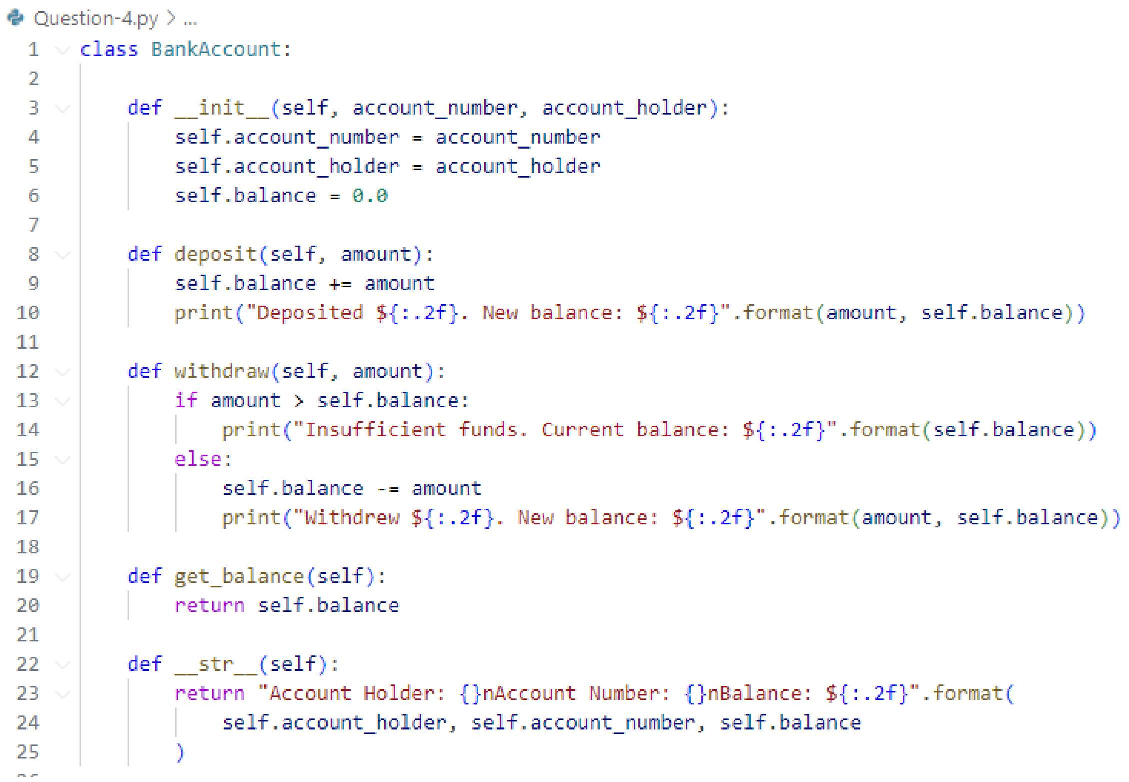
my_quiz.csv > data

```
how are you?,good
where are you from?,syria
14 > 5 ?, False
14 > 6 ?, False
14 > 7 ?, False
14 > 8 ?, False
14 > 58 ?, False
14 > 584 ?, False
14 > 59 ?, False
```


Question-3.py > ...

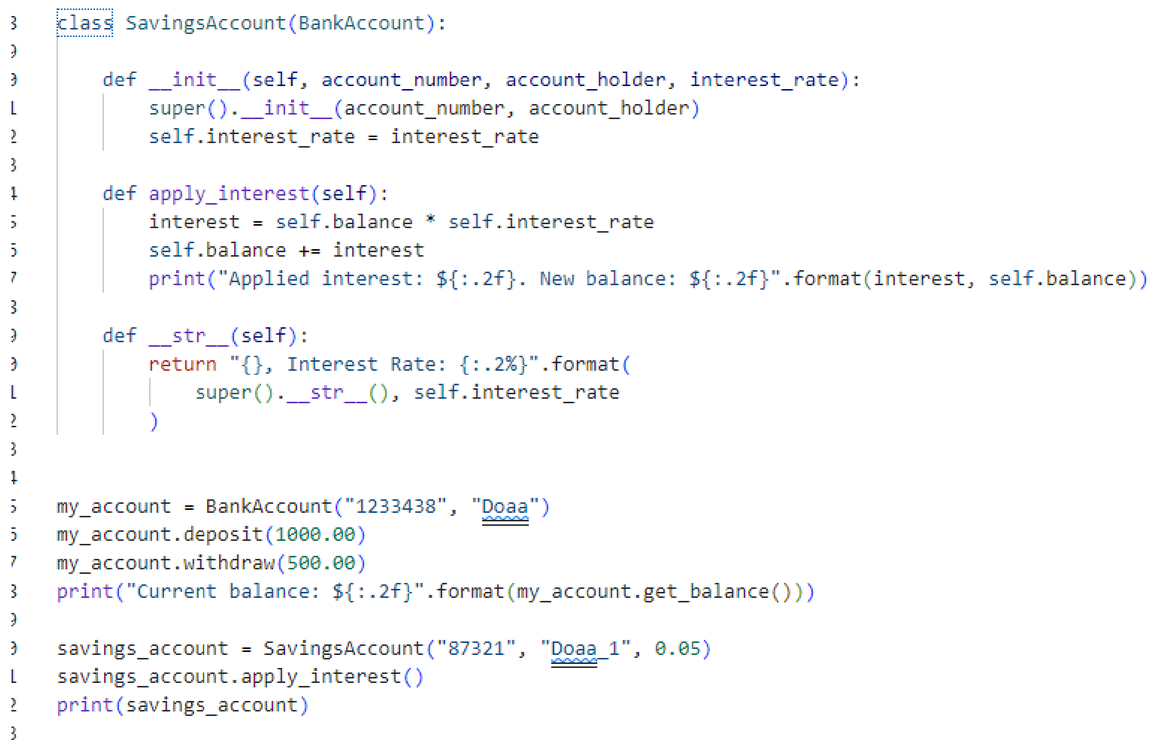
```python
import csv

with open ('my_quiz.csv') as file:
    do = csv.reader(file)
    questions = []
    answers = []
    for i in do:
        questions.append(i[0])
        answers.append(i[1])

name = input("Write Your name: ")

count = 0

for i in range(len(questions)):
    answer = input(questions[i] + " ")
    if answer.lower() == answers[i].lower():
        print("correct")
        count += 1
    else:
        print("Incorrect answer !")

print("YOU got :", count)

with open ('my_quiz_results.csv', mode = 'a') as file:
    do = csv.writer(file)
    do.writerow([name, count])
```

Question 4: Object-Oriented Programming - Bank Class Define a class BankAccount with the following attributes and methods: Attributes: account_number (string), account_holder (string), balance (float, initialized to 0.0) Methods:deposit(amount), withdraw(amount) , get_balance() - Create an instance of BankAccount, - Perform a deposit of $1000, - Perform a withdrawal of $500. - Print the current balance after each operation. - Define a subclass SavingsAccount that inherits from BankAccount and adds interest_rate Attribute and apply_interest() method that Applies interest to the balance based on the interest rate. And Override print() method to print the current balance and rate. - Create an instance of SavingsAccount , and call apply_interest() and print() functions.



Question-4.py > ...

```python
class BankAccount:

    def __init__(self, account_number, account_holder):
        self.account_number = account_number
        self.account_holder = account_holder
        self.balance = 0.0

    def deposit(self, amount):
        self.balance += amount
        print("Deposited ${:.2f}. New balance: ${:.2f}".format(amount, self.balance))

    def withdraw(self, amount):
        if amount > self.balance:
            print("Insufficient funds. Current balance: ${:.2f}".format(self.balance))
        else:
            self.balance -= amount
            print("Withdrew ${:.2f}. New balance: ${:.2f}".format(amount, self.balance))

    def get_balance(self):
        return self.balance

    def __str__(self):
        return "Account Holder: {}nAccount Number: {}nBalance: ${:.2f}".format(
            self.account_holder, self.account_number, self.balance
        )
```



```python
class SavingsAccount(BankAccount):

    def __init__(self, account_number, account_holder, interest_rate):
        super().__init__(account_number, account_holder)
        self.interest_rate = interest_rate

    def apply_interest(self):
        interest = self.balance * self.interest_rate
        self.balance += interest
        print("Applied interest: ${:.2f}. New balance: ${:.2f}".format(interest, self.balance))

    def __str__(self):
        return "{}, Interest Rate: {:.2%}".format(
            super().__str__(), self.interest_rate
        )


my_account = BankAccount("1233438", "Doaa")
my_account.deposit(1000.00)
my_account.withdraw(500.00)
print("Current balance: ${:.2f}".format(my_account.get_balance()))

savings_account = SavingsAccount("87321", "Doaa_1", 0.05)
savings_account.apply_interest()
print(savings_account)
```