الجمهورية العربية السورية

اللاذقية- جامعة تشرين

كلية الهندسة الميكانيكية والكهربائية

قسم هندسة الاتصالات والالكترونيات

السنة الخامسة: الوظيفة ١ برمجة شبكات

Network Programming Homework

سوسن محمد نور زير

٢٢٧٧

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} .

الخطوة ١: إنشاء قوائم:

- يتم إنشاء قائمة تسمى `LI` تحتوي على العناصر التالية `['HTTP', 'HTTPS', 'FTP', 'DNS']`
- يتم إنشاء قائمة تسمى `L2` تحتوي على الأرقام التالية: `[80, 443, 21, 53]`

الخطوة ٢: إنشاء قاموس:

- يتم استخدام دالة `zip()` لربط عناصر قائمة `LI` مع العناصر المقابلة لها في قائمة `L2`.
- يتم تخزين النتيجة في متغير يسمى `d`.

الخطوة ٣: طباعة القاموس:

- يتم استخدام دالة `print()` لعرض محتوى القاموس `d`.

شرح القاموس:

- القاموس `d` هو عبارة عن بنية بيانات تربط بين مفاتيح وقيم.
- في هذه الحالة، المفاتيح هي عناصر قائمة `LI` (بروتوكولات الشبكة) والقيم هي الأرقام المقابلة لها في قائمة `L2` أرقام المنافذ
- على سبيل المثال، المفتاح "HTTP" له القيمة ٨٠، مما يعني أن بروتوكول HTTP يستخدم عادةً المنفذ .

```
LI = ['HTTP', 'HTTPS', 'FTP', 'DNS']
L2 = [80, 443, 21, 53]
d = dict(zip(LI, L2))
print(d)
```

الخرج :

```
PROBLEMS    OUTPUT    TERMINAL    DEBUG CONSOLE
[Running] python -u "e:\python\ass1.py"
{'HTTP': 80, 'HTTPS': 443, 'FTP': 21, 'DNS': 53}

[Done] exited with code=0 in 0.199 seconds
```

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

الخطوة ١: تعريف دالة العامل(factorial)

- يتم تعريف دالة تسمى `factorial` التي تأخذ عدد صحيح `n` كمدخل.
- الدالة تحتوي على شرطين:
  - الشرط الأول :إذا كان `n` يساوي ٠، فإن الدالة ترجع القيمة ١ (عامل ٠ يساوي ١).
  - الشرط الثاني :إذا كان `n` أكبر من ٠، فإن الدالة تقوم بعملية الضرب التالية:
    - `n` مضروبا بدعوة جديدة للدالة `factorial` ولكن بقيمة (`n-1` أي عامل العدد السابق.)

الخطوة ٢: أخذ مدخل المستخدم

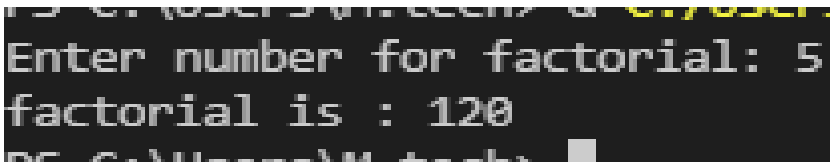
- يتم استخدام `input` لطلب من المستخدم إدخال رقم صحيح.
- يتم تحويل قيمة الإدخال إلى عدد صحيح باستخدام `int`.
- القيمة المدخلة يتم تخزينها في المتغير `x`.

الخطوة ٣: حساب عامل الرقم

- يتم استدعاء دالة `factorial` ويتم تمرير قيمة `x` كوسيطة لها.
- دالة `factorial` تقوم بحساب عامل `x` وتعيد النتيجة.

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

x = int(input("Enter number for factorial: "))
print("factorial is :", factorial(x))
```

الخرج :



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 .

يُستخدم هذا الكود لطباعة الكلمات من قائمة `L` التي تبدأ بالحرف "B".

١. **إنشاء القائمة"list"**:

- يتم إنشاء قائمة تسمى `L` تحتوي على العناصر التالية:

```
['Network', 'Bio', 'Programming', 'Physics', 'Music']
```

٢. **حلقة `for`**:

- يتم استخدام حلقة `for` لتكرار كل عنصر في القائمة `L`.
- في كل تكرار، يتم تعيين العنصر الحالي للقائمة إلى المتغير `item`.

٣. **شرط `if`**:

- داخل حلقة `for`، يتم استخدام شرط `if` للتحقق مما إذا كان العنصر `item` يبدأ بالحرف "B".
  - يتم استخدام دالة `startswith()` لتحديد ما إذا كان العنصر يبدأ بحرف معين.
  - إذا كان الشرط صحيحًا (أي أن العنصر يبدأ بـ "B")، يتم طباعة العنصر على الشاشة.

```
L = ['Network', 'Bio', 'Programming', 'Physics', 'Music']
for item in L:
    if item.startswith('B'):
        print(item)
```

الخرج:

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE
[Running] python -u "e:\python\ass1.py"
Bio

[Done] exited with code=0 in 0.602 seconds
```

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} .

يُستخدم هذا الكود لإنشاء قاموس (dictionary) يحتوي على مفاتيح (keys) وقيم (values) محددة.

١. **إنشاء القاموس:**" dictionary "

- يتم إنشاء متغير يسمى `d`.
- يتم تعيين قيمة `d`باستخدام دقة قاموس.(dictionary comprehension)
  - دقة القاموس هي طريقة مختصرة لإنشاء قواميس في لغة بايثون.
  - تتكون دقة القاموس من جزئين:

الجزء الأول `i: i+1`

- هذا الجزء يحدد كيفية إنشاء مفاتيح وقيم القاموس.
- `i`هو متغير يتكرر عبر الأرقام من ٠ إلى ١٠ (باستثناء ١١).
- `i+1`هي القيمة التي ستُربط بكل مفتاح `i`.

الجزء الثاني `for i in range(11)` :

- هذا الجزء يحدد كيفية تكرار المتغير `i`.
- يتم استخدام دالة `range(11)` لجعل `i`يتخذ كل قيمة من ٠ إلى ١٠.

```
d = {i: i+1 for i in range(11)}
print(d)
```

الخرج:

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE
[Running] python -u "e:\python\ass1.py"
{0: 1, 1: 2, 2: 3, 3: 4, 4: 5, 5: 6, 6: 7, 7: 8, 8: 9, 9: 10, 10: 11}

[Done] exited with code=0 in 0.183 seconds
```

Question 2: 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 .

- **دالة `binary_to_decimal_alt` :**

- تأخذ سلسلة ثنائية (`binary_str`) كمدخل.
- تُنشئ متغيرًا `القيمة_العشرية` لتخزين القيمة العشرية المكافئة.
- تستخدم دورة `for` مع `enumerate`للتكرار من خلال السلسلة الثنائية بترتيب عكسي (من اليمين إلى اليسار).
- داخل الدورة:
  - يتم التحقق من صحة كل رقم للتأكد من أنه إما "٠" أو "١".
  - يتم تحويل الرقم الثنائي الحالي إلى عدد صحيح باستخدام `int()`.رقم
  - يتم حساب المساهمة العشرية للرقم الحالي باستخدام `الرقم_كعدد_صحيح << i.`
    - `<<`يُمثل عملية الإزاحة الثنائية لليسار.
    - `i`يمثل القوة الحالية لـ ٢ (تبدأ من ٠ وتزداد مع كل تكرار).
  - يتم إضافة المساهمة العشرية إلى `القيمة_العشرية`.
- تُعيد الدالة `القيمة_العشرية` المحسوبة.

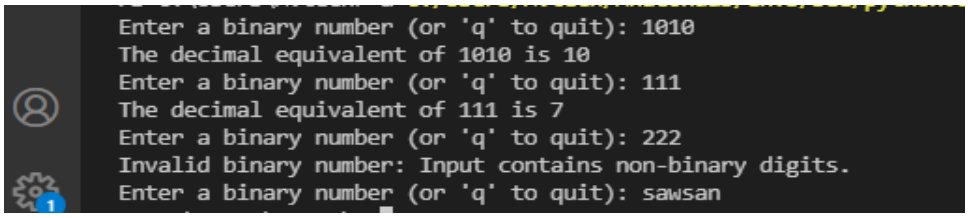
الجزء الرئيسي:(`if __name__ == "__main__":`)

- يستخدم دورة `while`لتكرار عملية التحويل حتى يدخل المستخدم "q" للخروج.
- داخل الدورة:
  - يُطلب من المستخدم إدخال رقم ثنائي) أو "q" للخروج.(
  - يتم التحقق من صحة إدخال المستخدم للتأكد من أنه سلسلة ثنائية صالحة.
  - يتم استدعاء دالة `binary_to_decimal_alt`لتحويل الرقم الثنائي إلى عشري.
  - يتم عرض المكافئ العشري للمستخدم.
- يتم التعامل مع أي أخطاء `ValueError`(مثل أرقام غير ثنائية في الإدخال) وعرض رسالة خطأ مناسبة.

```
def binary_to_decimal_alt(binary_str):
    """Converts a binary string to its equivalent decimal number using
bitwise operations.
```

```
    Args:
        binary_str: The binary string to convert.

    Returns:
        The decimal equivalent of the binary string, or None if the inp
ut is invalid.

    Raises:
        ValueError: If the input string contains non-binary digits.
    """

    decimal_value = 0
    for i, digit in enumerate(reversed(binary_str)):
        if digit not in '01':
            raise ValueError("Invalid binary number: Input contains non
-binary digits.")

        # Convert binary digit to integer
        digit_int = int(digit)

        # Calculate decimal contribution using bitwise operations
        decimal_value += digit_int << i

    return decimal_value


if __name__ == "__main__":
    while True:
        try:
            binary_str = input("Enter a binary number (or 'sawsan' to q
uit): ")

            if binary_str.lower() == 'q':
                break

            decimal_value = binary_to_decimal_alt(binary_str)
            print("The decimal equivalent of", binary_str, "is", decima
l_value)
        except ValueError as e:
            print(e)
```

الخرج:



Question **3:** Working With Files "Quiz Program"

Type python quiz program that takes a text or json or csv file as input for (20(Question , Answer)). It asks the question and finally computes and prints user name and result in separate file csv or json file .

هذا الكود عبارة عن اختبار بسيط باستخدام لغة بايثون، يتكون من ثلاثة أجزاء رئيسية:

١ـ دالتان للتعامل مع الأسئلة:

- **دالة تحميل الأسئلة `load_quiz`:**
  - تأخذ هذه الدالة مسار ملف (متغير `file_path`) كمدخل.
  - تستخدم الدالة `open` لفتح الملف للقراءة.(`'r'`)
  - داخل كتلة `with` يتم تحميل محتوى الملف باستخدام دالة `json.load`.
  - تقوم الدالة بإرجاع القاموس الذي تم تحميله من الملف (الأسئلة).
- **دالة تقديم الاختبار `take_quiz`:**
  - تأخذ هذه الدالة قائمة من الأسئلة (متغير `questions`) كمدخل.
  - يتم تعريف متغير `score` لتخزين النتيجة (بدايةً صفر).
  - تتكرر الدالة على كل سؤال في القائمة باستخدام حلقة `for`.
    - يتم طباعة نص السؤال مخزن في مفتاح `question` لكل سؤال
    - يطلب من المستخدم إدخال إجابته باستخدام `input`.
    - يتم تحويل إجابة المستخدم والاجابة الصحيحة مخزنة في مفتاح `answer` إلى حروف صغيرة باستخدام `lower()` للتجاهل بين الحروف الكبيرة والصغيرة.
    - إذا كانت إجابة المستخدم مطابقة للإجابة الصحيحة، تتم زيادة النتيجة (`score`) بواحد.
  - الدالة ترجع قيمة النتيجة النهائية (مجموع الإجابات الصحيحة).

٢ـتحميل الأسئلة من ملف:

- يتم تعريف متغير `questions` ويستدعى دالة `load_quiz` لتخزين الأسئلة المحملة من ملف `quiz.json` يجب أن يكون موجودًا بنفس المسار

٣ـإجراء الاختبار وعرض النتيجة:

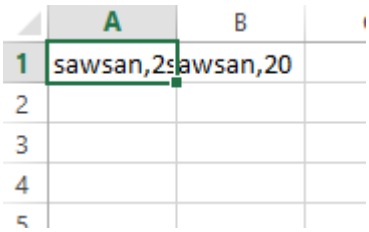
- يتم استدعاء دالة `take_quiz` و تمرير قائمة الأسئلة `questions` لها.

- تتولى الدالة عرض الأسئلة، أخذ إجابات المستخدم، وتجميع النتيجة.
- يتم تخزين نتيجة المستخدم في متغير `user_score`.
- وأخيرا، يتم طباعة عبارة "النتيجة هي.(`user_score`) followed by the actual score (`user_score`) :"

```python
import json

def load_quiz(file_path):
    with open(file_path, 'r') as file:
        return json.load(file)

def take_quiz(questions):
    score = 0
    for question in questions:
        print(question['question'])
        answer = input("Enter answer: ")
        if answer.lower() == question['answer'].lower():
            score += 1
    return score

#  'quiz.json'
questions = load_quiz('quiz.json')
user_score = take_quiz(questions)
user_name = input("Enter your name: ")
with open ("result.csv","a") as file:
    file.write(f'{user_name},{user_score}')
print("result is:", user_score)
```

الخرج:

```
9-8=
Enter answer: 1
1-1=
Enter answer: 0
8*6=
Enter answer: 48
1-1=
Enter answer: 0
2*2=
Enter answer: 4
5-5=
Enter answer: 0
4*2=
Enter answer: 8
1*1=
Enter answer: 1
what's your name=
Enter answer: sawsan
Enter your name: sawsan
result is: 20
result is: 20
```



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 rate .
- And Override print() method to print the current balance and rate .
- Create an instance of SavingsAccount , and call apply_interest() and print() function .

هذا الكود يوضح إنشاء فئتين (Classes) في لغة بايثون لتمثيل حسابات بنكية:

١- الفئة الأساسية حساب بنكي `BankAccount`:

- تُستخدم هذه الفئة لإنشاء حساب بنكي عام.
- **دالة البناء `__init__`:**
  - تأخذ هذه الدالة ثلاثة متحولات كمدخلات:
    - `account_number` رقم الحساب
    - `account_holder` اسم صاحب الحساب
    - `balance` الرصيد - اختياري، الافتراضي
  - تقوم الدالة بتعريف وتعيين قيم المتغيرات التالية داخل الكائن:(object)
    - `self.account_number`: يمثل رقم الحساب
    - `self.account_holder`: يمثل اسم صاحب الحساب
    - `self.balance`: يمثل رصيد الحساب

- **دالة إيداع deposit:**
  - تأخذ هذه الدالة مبلغ الإيداع amount كمدخل.
  - تضيف قيمة amount إلى رصيد الحساب self.balance.
  - تطبع رسالة توضيحية على الشاشة تشير بقيمة الإيداع والمبلغ الجديد للرصيد.
- **دالة سحب withdraw :**
  - تأخذ هذه الدالة مبلغ السحب amountكمدخل.
  - تتحقق من كفاية الرصيد لسحب المبلغ المطلوب.
    - إذا كان الرصيد غير كافٍ، تطبع رسالة تفيد بذلك.
  - إذا كان الرصيد كافيًا، يتم خصم قيمة السحب amountمن الرصيد self.balance .
  - تطبع رسالة توضيحية على الشاشة تشير بقيمة السحب والمبلغ الجديد للرصيد.
- **دالة الحصول على الرصيد get_balance:**
  - لا تأخذ هذه الدالة أي قيم كمدخل.
  - ترجع قيمة رصيد الحساب المخزن في المتغير self.balance.

٢- الفئة المشتقة حساب توفير:

- ترث هذه الفئة خصائص الفئة الأساسية BankAccount.
- **دالة البناء :__init__**
  - تستدعي دالة البناء للصف الأساسي __init__.()super لتعيين القيم الأساسية (رقم الحساب، اسم صاحب الحساب، الرصيد).
  - بالإضافة إلى ذلك، تعرف المتغير self.interest_rateلتخزين نسبة الفائدة على الحساب.
- **دالة تطبيق الفائدة apply_interest :**
  - تحسب قيمة الفائدة عن طريق ضرب رصيد الحساب self.balanceبنسبة الفائدة self.interest_rate.
  - تستدعي دالة depositلإضافة قيمة الفائدة المحسوبة إلى رصيد الحساب.
  - تطبع رسالة توضيحية على الشاشة تشير بنسبة الفائدة المطبقة والمبلغ الجديد للرصيد.

3.استخدام الفئات:

- يتم إنشاء كائن من الفئة الفرعية SavingsAccount وذلك بتعريف متغيّر account.
  - يتم تمرير المعلومات اللازمة عند إنشاء الكائن:
    - '12345' : رقم الحساب
    - 'sawsan' :اسم صاحب الحساب
    - 0.05 (٥%) نسبة الفائدة
- يتم استدعاء دالة depositلإيداع مبلغ ١٠٠٠ في الحساب.
- يتم استدعاء دالة withdraw لسحب مبلغ ٥٠٠ من الحساب.
- يتم استدعاء دالة apply_interest لحساب وتطبيق الفائدة على رصيد الحساب.

```python
class BankAccount:
    def __init__(self, account_number, account_holder, balance=0.0):
        self.account_number = account_number
        self.account_holder = account_holder
        self.balance = balance

    def deposit(self, amount):
        self.balance += amount
        print(f" deposit {amount}. balance is {self.balance}")

    def withdraw(self, amount):
        if amount > self.balance:
            print("balance doesn’t enough ")
        else:
            self.balance -= amount
            print(f" withdraw {amount}. balance is {self.balance}")

    def get_balance(self):
        return self.balance

class SavingsAccount(BankAccount):
    def __init__(self, account_number, account_holder, interest_rate
, balance=0.0):
        super().__init__(account_number, account_holder, balance)
        self.interest_rate = interest_rate

    def apply_interest(self):
        interest = self.balance * self.interest_rate
        self.deposit(interest)
        print(f" apply interest {self.interest_rate}. balance
is{self.balance}")

account = SavingsAccount('12345', 'sawsan', 0.05)
account.deposit(1000)
account.withdraw(500)
account.apply_interest()
```

الخرج :

```
[Running] python -u "e:\python\ass1.py"
 deposit 1000. balance is 1000.0
 withdraw 500. balance is 500.0
 deposit 25.0. balance is 525.0
 apply interest 0.05. balance is525.0
```