الجمهورية العربية السورية

جامعة تشرين

كلية الهندسة الميكانيكية والكهربائية

قسم هندسة الاتصالات والإلكترونيات

السنة الخامسة

وظيفة python

إعداد الطالب :

جلنار بدور

إشراف :

د. مهند عيسى

العام الدراسي : 2023 - 2024

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 }

```
L1 = ['HTTP', 'HTTPS', 'FTP', 'DNS']

L2 = [80, 443, 21, 53]

d = dict(zip(L1, L2))

print(d)
```

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

```
def factorial(n):
    if n < 0:
        return "Factorial is not defined for negative numbers."
    elif n == 0 or n == 1:
        return 1
    else:
        result = 1
        for i in range(2, n + 1):
            result *= i
        return result
# Get user input
number = int(input("Enter a number: "))
# Calculate factorial
result = factorial(number)
# Display the result
print(f"The factorial of {number} is {result}.")
```

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.

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

# Identify and print items starting with 'B'

for item in L:

   if item.startswith('B'):

      print(item)
```

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}

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

print(d)
```

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.Tips: solve input errors.

```
def binary_to_decimal(binary_str):

   try:

      # Convert binary string to decimal number

      decimal_number = int(binary_str, 2)

      return decimal_number

   except ValueError:

      # Handle cases where input is not a valid binary number

      return None

def main():

   while True:

      # Get binary number input from user
```

```
        binary_str = input("Enter a binary number: ")
        # Convert to decimal
        decimal_number = binary_to_decimal(binary_str)
        if decimal_number is not None:
            # If conversion is successful, display the result
            print(f"The decimal equivalent of binary {binary_str} is {decimal_number}.")
            break
        else:
            # If conversion fails, display an error message and prompt again
            print("Invalid binary number. Please enter a valid binary number (only 0 and 1).")
if _name_ == "_main_":
    main()
```

Question 3: 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.

```
[
    {"question": "What is the capital of France?", "answer": "Paris"},
    {"question": "What is 2 + 2?", "answer": "4"},
    {"question": "What is the color of the sky?", "answer": "Blue"},
    {"question": "What is the largest planet in our solar system?", "answer": "Jupiter"},
    {"question": "What is the boiling point of water?", "answer": "100"},
    {"question": "What is the currency of the United States?", "answer": "Dollar"},
    {"question": "Who wrote 'To Kill a Mockingbird'?", "answer": "Harper Lee"},
    {"question": "What is the chemical symbol for gold?", "answer": "Au"},
    {"question": "What is the capital of Japan?", "answer": "Tokyo"},
    {"question": "What is the largest mammal?", "answer": "Blue Whale"},
    {"question": "What is the smallest prime number?", "answer": "2"},
    {"question": "What is the main ingredient in guacamole?", "answer": "Avocado"},
    {"question": "What is the hardest natural substance on Earth?", "answer": "Diamond"},
    {"question": "What is the tallest mountain in the world?", "answer": "Mount Everest"},
```

```
    {"question": "Who painted the Mona Lisa?", "answer": "Leonardo da Vinci"},
    {"question": "What is the capital of Canada?", "answer": "Ottawa"},
    {"question": "What is the main gas found in the air we breathe?", "answer": "Nitrogen"},
    {"question": "Who is known as the Father of Computers?", "answer": "Charles Babbage"},
    {"question": "What is the square root of 64?", "answer": "8"},
    {"question": "What is the longest river in the world?", "answer": "Nile"}
]
```

```
import json
import csv

# Function to load questions from a JSON file
def load_questions(filename):
    with open(filename, 'r') as file:
        questions = json.load(file)
    return questions

# Function to ask questions and get user responses
def conduct_quiz(questions):
    score = 0
    for idx, q in enumerate(questions):
        print(f"Q{idx + 1}: {q['question']}")
        answer = input("Your answer: ")
        if answer.strip().lower() == q['answer'].strip().lower():
            score += 1
    return score

# Function to save user results to a CSV file
def save_results(filename, username, score, total_questions):
    with open(filename, 'a', newline='') as file:
        writer = csv.writer(file)
        writer.writerow([username, score, total_questions])

def main():
    # Load questions
    questions = load_questions('questions.json')

    # Get user's name
    username = input("Enter your name: ")

    # Conduct the quiz
    score = conduct_quiz(questions)

    # Display the results
    total_questions = len(questions)
    print(f"{username}, you scored {score} out of {total_questions}.")
```

```
    # Save the results
    save_results('results.csv', username, score, total_questions)

if _name_ == "_main_":
    main()
```

Question 4: Object-Oriented Programming - Bank ClassDefine 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.

```
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):
        if amount > 0:
            self.balance += amount
            print(f"Deposited ${amount:.2f}. New balance: ${self.balance:.2f}")
        else:
            print("Deposit amount must be positive.")
    def withdraw(self, amount):
        if 0 < amount <= self.balance:
            self.balance -= amount
            print(f"Withdrew ${amount:.2f}. New balance: ${self.balance:.2f}")
        else:
            print("Insufficient balance or invalid withdrawal amount.")
    def __str__(self):
        return str(self.balance)
```

```
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 / 100
        self.balance += interest
        print(f"Applied interest: ${interest:.2f}. New balance: ${self.balance:.2f}")
    def __str__(self):
        return f"Account holder: {self.account_holder}, Balance: ${self.balance:.2f}, Interest rate: {self.interest_rate}%"
# Create an instance of BankAccount
account = BankAccount("123456789", "John Doe")
# Perform a deposit of $1000
account.deposit(1000)
# Perform a withdrawal of $500
account.withdraw(500)
# Print the current balance after each operation
print("balance:",end ='')
print(account)
# Create an instance of SavingsAccount
savings_account = SavingsAccount("987654321", "Jane Doe", 2.5)
# Perform a deposit of $1000
savings_account.deposit(1000)
# Apply interest
savings_account.apply_interest()
# Print the current balance and interest rate
print(savings_account)
```