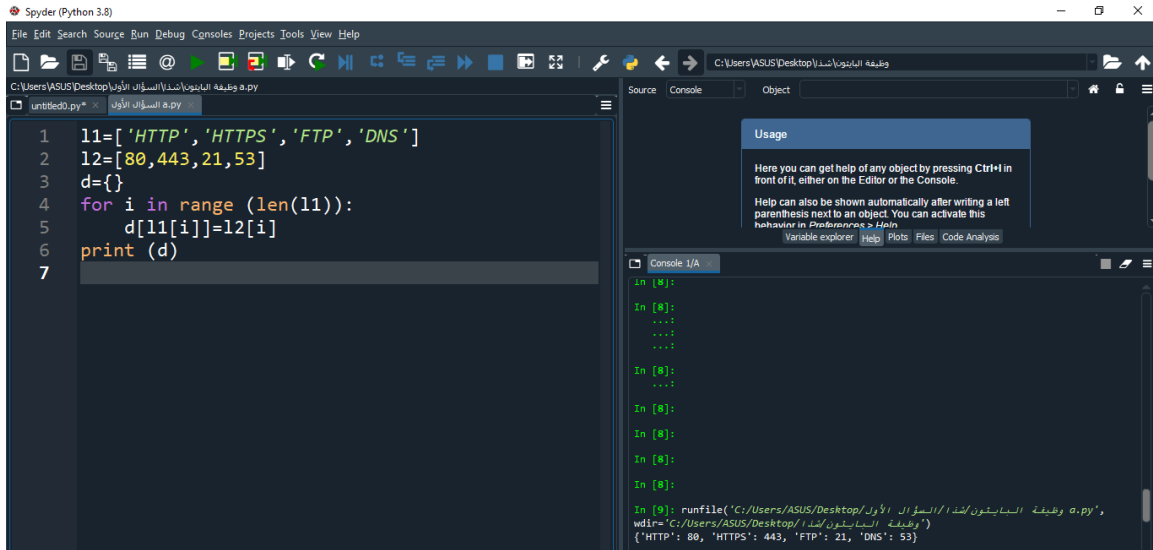


شذا رمضان 2973

السؤال الأول:

a. لدينا مصفوفتين نقوم بدمج المصفوفتين عن طريق القاموس بنسب كل قيمه من المصفوفة الأولى (المفتاح) الى قيمتها في المصفوفة الثانية على نفس الترتيب

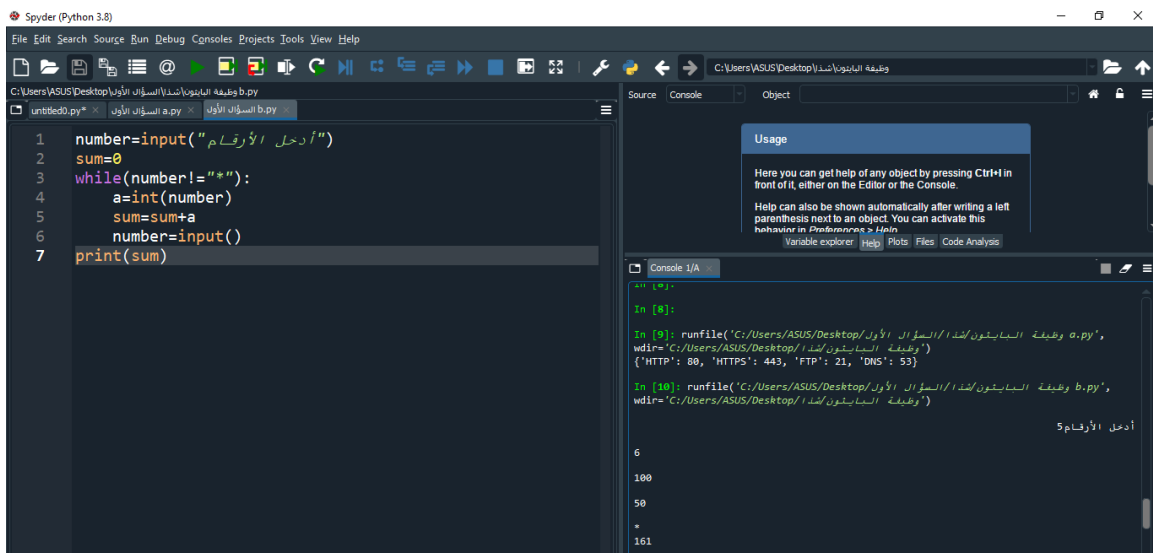


```
1 11=['HTTP','HTTPS','FTP','DNS']
2 12=[80,443,21,53]
3 d={}
4 for i in range(len(11)):
5     d[11[i]]=12[i]
6 print (d)
7
```

Console I/A

```
In [8]:
...:
...:
...:
In [8]:
...:
...:
In [8]:
...:
...:
In [8]:
...:
...:
In [8]:
...:
...:
In [8]:
...:
...:
In [9]: runfile('C:/Users/ASUS/Desktop/السؤال الأول/السؤال الأول a.py',
wdir='C:/Users/ASUS/Desktop/السؤال الأول/السؤال الأول a.py',
{'HTTP': 80, 'HTTPS': 443, 'FTP': 21, 'DNS': 53})
```

b. ننشئ حلقة تقوم بإدخال عدد لانهاى من الأعداد بحيث ينتهي الإدخال بالرمز * وتتم طباعة المجموع:



```
1 number=input("أدخل الأرقام ")
2 sum=0
3 while(number!="*"):
4     a=int(number)
5     sum=sum+a
6     number=input()
7 print(sum)
```

Console I/A

```
In [9]: runfile('C:/Users/ASUS/Desktop/السؤال الأول/السؤال الأول a.py',
wdir='C:/Users/ASUS/Desktop/السؤال الأول/السؤال الأول a.py',
{'HTTP': 80, 'HTTPS': 443, 'FTP': 21, 'DNS': 53})

In [10]: runfile('C:/Users/ASUS/Desktop/السؤال الأول/السؤال الأول b.py',
wdir='C:/Users/ASUS/Desktop/السؤال الأول/السؤال الأول b.py',
{'HTTP': 80, 'HTTPS': 443, 'FTP': 21, 'DNS': 53})

أدخل الأرقام
6
100
50
*
151
```

c. ننشئ حلقة تمر على جميع عناصر المصفوفة وتختبر أول حرف من كل عنصر إذا كان B يتم طباعة العنصر.

```

1 l=['Network','Bio','Programming','Physics','Music']
2 for a in l:
3     if a[0]=='B':
4         print(a)

```

The console shows the following output:

```

In [13]: runfile('C:/Users/ASUS/Desktop/السؤال الأول/السؤال الأول.py',
             wdir='C:/Users/ASUS/Desktop/السؤال الأول/السؤال الأول.py',
             line 4, in
             <module>
             print(a[a])
             TypeError: list indices must be integers or slices, not str

In [14]: runfile('C:/Users/ASUS/Desktop/السؤال الأول/السؤال الأول.py',
             wdir='C:/Users/ASUS/Desktop/السؤال الأول/السؤال الأول.py',
             line 4, in
             <module>
             print(a[a])
             TypeError: string indices must be integers

```

d. نلاحظ وجود علاقة بين المفتاح والقيمة وهي +1

```

1 d={x:x+1 for x in range(11)}
2 print(d)
3

```

The console shows the following output:

```

In [13]: runfile('C:/Users/ASUS/Desktop/السؤال الأول/السؤال الأول.py',
             wdir='C:/Users/ASUS/Desktop/السؤال الأول/السؤال الأول.py',
             line 4, in
             <module>
             print(a[a])
             TypeError: string indices must be integers

In [14]: runfile('C:/Users/ASUS/Desktop/السؤال الأول/السؤال الأول.py',
             wdir='C:/Users/ASUS/Desktop/السؤال الأول/السؤال الأول.py',
             line 4, in
             <module>
             print(a[a])
             TypeError: string indices must be integers

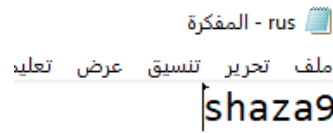
In [15]: runfile('C:/Users/ASUS/Desktop/السؤال الأول/السؤال الأول.py',
             wdir='C:/Users/ASUS/Desktop/السؤال الأول/السؤال الأول.py',
             line 4, in
             <module>
             print(a[a])
             TypeError: string indices must be integers

```

السؤال الثاني:

يتم إدخال سلسلة أصفار وواحدات يتم تحويلها إلى أرقام مع الحفاظ على نفس ترتيب المواقع ومن ثم تطبيق تابع التحويل إلى القيمة العشرية.

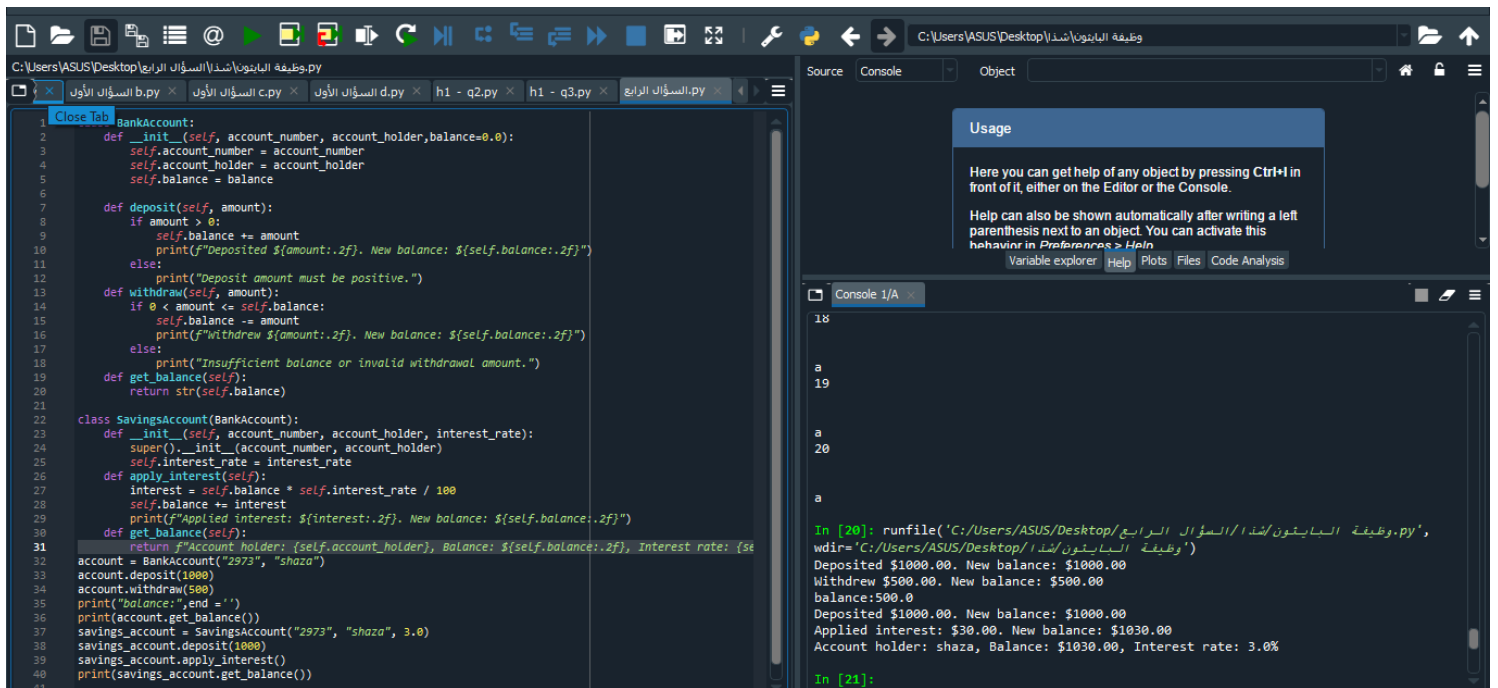
ويكون شكل ملف النتيجة:



السؤال الرابع:

إنشاء صنف لحساب بنك مع بعض التوابع البسيطة لتغيير المبلغ في الحساب والسحب منه. واشتقاق كائن من هذا الصنف واستدعاء التوابع ضمنه.

إنشاء صنف جديد يعتمد على الصنف السابق يسمح بحساب قيمة المبلغ بعد إضافة الفائدة.



```
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 get_balance(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 get_balance(self):
        return f"Account holder: {self.account_holder}, Balance: ${self.balance:.2f}, Interest rate: {self.interest_rate:.2f}"

account = BankAccount("2973", "shaza")
account.deposit(1000)
account.withdraw(500)
print("balance:", end='')
print(account.get_balance())
savings_account = SavingsAccount("2973", "shaza", 3.0)
savings_account.deposit(1000)
savings_account.apply_interest()
print(savings_account.get_balance())
```

Usage

Here you can get help of any object by pressing **Ctrl+I** in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in **Preferences > Help**

Variable explorer | Help | Plots | Files | Code Analysis

Console 1/A

```
18
a
19
a
20
a
a

In [20]: runfile('C:/Users/ASUS/Desktop/السؤال الرابع/وظيفة البانكون/هذا.py',
wdir='C:/Users/ASUS/Desktop/السؤال الرابع/وظيفة البانكون/هذا.py')
Deposited $1000.00. New balance: $1000.00
Withdraw $500.00. New balance: $500.00
balance:500.0
Deposited $1000.00. New balance: $1000.00
Applied interest: $30.00. New balance: $1030.00
Account holder: shaza, Balance: $1030.00, Interest rate: 3.0%

In [21]:
```