TD0sup_corrige

August 25, 2026

1 TD0_sup - code capytal : dc83-6947790

1.1 Exercice 1 :Vérification d'un email

Ecrire une fonction Est_mail(adresse) qui vérifie si une adresse contient au moins un "@" et un "." et qui prend une chaine de caractère adresse en argument. La fonction retourne True si l'adresse est correct False dans le cas contraire.

[5]:
```
def Est_mail(adresse):
    etat1,etat2=False,False
    for i in adresse :
        if i=="@":
            etat1=True
        elif i==".":
            etat2=True
    return etat1 and etat2
test("meretalex@yahoo.fr")
```

[5]:
```
True
```

[6]:
```
def Est_mail(adresse):
    if ("@" in adresse and "." in adresse):
        return True
    else :
        return False
adresse="Alexis.Meret@ac-poitiers.fr"
Est_mail(adresse)
```

[6]:
```
True
```

1.2 Exercice 2 : Conversion

Ecrire une fonction conversion(T) qui convertit une température T en degré Celsius en degré Faranheit.

[7]:
```
def conversion(T):
    return T*1.8+32
```

1.3 Exercice 3 : Voyelle ?

Ecrire une fonction Est_voyelle(c) qui retourne True si le caractère c est une voyelle et faux sinon.

[8]:
```
def Est_voyelle(c):
    if c in 'aeiouy' or 'AEIOUY':
        return True
    else :
        return False
Est_voyelle("A")
```

[8]:
```
True
```

1.4 Exercice 4 : Qualité d'un mot de passe

Ecrire une fonction test(MP) qui prend en argument une chaîne de caractère MP représentant une mot de passe et qui renvoie : - "faible" si MP contient mois de 6 caractères - "moyen" s'il contient entre 6 et 10 - "fort" sinon

[2]:
```
def test(MP):
    n=len(MP)
    if n>10:
        return "fort"
    elif n>=6:
        return "moyen"
    else :
        return "faible"
test("alexis")
```

[2]:
```
'moyen'
```

[4]:
```
def test2(MP):
    n=len(MP)
    if n<=6:
        return "faible"
    elif n<=10 :
        return "moyen"
    else :
        return "fort"
test2("alexis")
```

[4]:
```
'faible'
```

[7]:
```
def test3(MP):
    n=len(MP)
    return "faible" if n<=6 else "moyen" if n<=10 else "fort"
test3("alexos")
```

[7]:
```
'faible'
```

1.5 Exercice 5: TVA

Ecrire une fonction prix_ttc(HT,TVA) qui prend en argument le prix hors HT et la tva en pourcentage notée TVA et qui renvoie le prix TTC.

[12]:
```
def prix_ttc(HT,TVA):
    return HT+TVA*HT/100
prix_ttc(10,19.6)
```

[12]:
```
11.96
```

1.6 Exercice 6: Comparaison de 3 nombres

Ecrire une fonction qui prend en argument trois nombres et qui renvoie le plus grand des trois nombres.

[23]:
```
def f1(a,b,c):
    if a>b:
        if a>c:
            return a
        else :
            return c
    else :
        if b>c:
            return b
        else :
            return c
f1(10,11,12)
```

[23]:
```
12
```

[11]:
```
def f2(a,b,c):
    if a>=b and a>=c:
        return a
    elif b>=a and b>=c:
        return b
    else :
        return c
f2(10,11,12)
```

[11]:
```
12
```

[15]:
```
def f3(a,b,c):
    plus_grand=a
    if b>plus_grand:
        plus_grand=b
```

```
    if c>plus_grand:# très bon exemple de la nécessité du if et pas du elif
        plus_grand=c
    return plus_grand
f3(10,20,30)
```

[15]: 30

1.7 Exercice 7 : initiales

Ecrire une fonction qui prend en arguments un prénom (sous la forme d'une chaîne de caractères en minuscule) et un nom (sous la forme d'une chaîne de caractères en minuscule) et qui renvoie les initiales en majuscule. On donne les éléments suivants :

[51]:
```
print(ord("a"))
print(ord("A"))
print(chr(97))
print(chr(65))
```

```
97
65
a
A
```

[52]:
```
def initiales(prenom,nom):
    p,n=prenom[0],nom[0]
    P,N=chr(ord(p)-32),chr(ord(n)-32)
    return f"{P}.{N}"
initiales("alexis","meret")
```

[52]: 'A.M'

1.8 Exercice 8 : Comparaison de deux dates

Ecrire une fonction plus_recente(date1,date2) qui compare deux dates au format "LL/MM/AAAA". La fonction retourne la plus récente.

[12]:
```
def plus_recente(date1,date2):
    annee1,annee2=int(date1[6:]),int(date2[6:])
    mois1,mois2=int(date1[3:5]),int(date2[3:5])
    jour1,jour2=int(date1[:2]),int(date2[:2])
    if annee1>annee2:
        return date1
    elif annee1<annee2:
        return date2
    if mois1>mois2:
        return date1
    elif mois1<mois2:
        return date2
    if jour1>jour2:
```

```
        return date1
    else :
        return date2
plus_recente("31/10/1981","03/10/2016")
```

[12]: '03/10/2016'

[2]:
```
def plus_recente(date1,date2):
    a1,a2=int(date1[6:10]),int(date2[6:])
    m1,m2=int(date1[3:5]),int(date2[3:5])
    j1,j2=int(date1[0:2]),int(date2[0:2])
    if (a1,m1,j1)>(a2,m2,j2):#comparaison élément par élément (arrêt dès qu'une␣
 ↪condition est vraie)
        return date1
    else:
        return date2
plus_recente("31/10/1981","17/11/1980")
```

[2]: '31/10/1981'

1.9 Exercice 9: mesure de durée

Ecrire une fonction duree(depart,arrivée) qui calcule la duree d'un trajet exprimée en heures et minutes. Les temps sont données au format "HH:MM".

[19]:
```
def duree(depart,arrivee):
    H1,H2=int(depart[:2]),int(arrivee[:2])
    M1,M2=int(depart[3:]),int(arrivee[3:])
    diff1=H2-H1
    diff2=M2-M1
    if diff2<0:
        diff2 = 60-abs(diff2)
        diff1=diff1-1
    return f"{diff1}:{diff2}"
duree("10:50","12:00")
```

[19]: '1:10'

1.10 Exercice 10: Calculatrice simple

Ecrire une fonction calcul(a,b,opération) qui prend en argument deux nombres a et b ainsi qu'une opération ("*","+","-","/") à effectuer et qui renvoie la valeur du calcul.

[20]:
```
def calcul(a, b, op):# bel exemple de la nécessité des elif au lieu des if
    if op == "+":
        return a + b
    elif op == "-":
        return a - b
```

```
    elif op == "*":
        return a * b
    elif op == "/":
        if b != 0:
            return a / b
        else:
            return "Erreur"
    else:
        return "Opération inconnue"
calcul(2, 5, "/")
```

[20]: 0.4

[3]:
```
def calcul(a,b,operation):
    dico={"+":a+b,"-":a-b,"*":a*b,"/" :a/b if b!=0 else False}
    return dico[operation]
calcul(3,4,"/")
```

[3]: 0.75

[ ]: