

**PROTOTYPE**  
**2022**

**Épreuve d'informatique**

Sections : **Mathématiques, Sciences  
expérimentales et Sciences techniques**

Durée : **1 heure 30 mn**

Coefficient : **0.5**

**Exercice 1 (2 points)**

On veut déterminer et afficher le nombre de diviseurs d'un entier  $n$  strictement positif.

**Exemple :**

Pour  $n=6$ , le nombre de diviseurs de 6 est égal à 4. En effet les diviseurs de 6 sont  $\{1 ; 2 ; 3 ; 6\}$ .

On vous propose les trois séquences d'instructions algorithmiques suivantes :

<i>Séquence 1</i>	<i>Séquence 2</i>	<i>Séquence 3</i>
$c \leftarrow 0$ Pour $k$ de 1 à $n$ faire Si $(n \bmod k = 0)$ alors $c \leftarrow c + 1$ Fin Si Fin Pour Ecrire ( $c$ )	$c \leftarrow 1$ Pour $k$ de 2 à $(n \text{ div } 2)$ faire Si $(n \bmod k = 0)$ alors $c \leftarrow c + 1$ Fin Si Fin Pour Ecrire ( $c$ )	$c \leftarrow 2$ Pour $k$ de 2 à $(n-1)$ faire Si $(n \bmod k = 0)$ alors $c \leftarrow c + 1$ Fin Si Fin pour Ecrire ( $c$ )

- 1) Compléter le tableau ci-dessous par la valeur de la variable  $c$  après exécution de chaque séquence, et ce pour  $n=4$ .

Séquence	Valeur de la variable $c$
1	
2	
3	

- 2) Donner le numéro de la séquence qui ne permet pas d'afficher le nombre de diviseurs. Justifier votre réponse.

.....  
.....



## Exercice 2 (3 points)

Un médecin veut chercher la fiche d'un de ses patients en connaissant son nom. Pour cela, il utilise un tableau **T** contenant **N** noms.

- 1) Compléter la séquence algorithmique présentée ci-dessous afin de vérifier l'existence d'un nom donné **NOM** dans un tableau **T** non vide.

```

Algorithme recherche
Début
  Écrire ("Donner le nom à chercher : ")
  Lire (.....)
  Existe ← .....
  i ← .....
  Répéter
    Si (T[i] = NOM) alors
      Existe ← .....
    Sinon
      i ← .....
  Finsi
  Jusqu'à (.....) ou (.....)
  Si (.....) alors
    Écrire ("Le nom recherché existe dans ce tableau.")
  Sinon
    Écrire ("Le nom recherché n'existe pas dans ce tableau.")
  Finsi
Fin
    
```

T.D.O.	
Objet	Type/Nature
T	Tableau de N chaînes
N, i	Entier
NOM	Chaîne
Existe	Booléen

- 2) Ce médecin veut chercher les numéros des fiches de ses patients ayant le même nom. Modifier la séquence algorithmique présentée ci-dessus afin d'afficher ces numéros.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....



### Exercice 3 (5 points)

La propagation de l'épidémie Covid-19 suit une croissance exponentielle. Pour déterminer et afficher le nombre total de personnes contaminées pendant un nombre de jours donné  $N$  et pour  $x$  personnes initialement contaminées on utilise la formule suivante :

$$e^x = \sum_{i=0}^N \frac{(x)^i}{i!} = 1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \dots + \frac{(x)^N}{N!}$$

Donner un algorithme solution à ce problème.

### Exercice 4 (10 points)

Un débutant en anglais veut élaborer son propre **carnet** comme étant un dictionnaire **FRANÇAIS/ANGLAIS** pour l'utiliser afin de traduire des phrases au cours de sa formation. Pour cela, on admettra qu'un traducteur du français à l'anglais peut être simplifié par une traduction de mot-à-mot.

Ce carnet **FRANÇAIS/ANGLAIS** est formé par  $N$  mots en français et par  $N$  mots en anglais, de sorte que chaque mot en français lui correspond son équivalent en anglais, avec  $2 \leq N \leq NMAX$  ( $NMAX$  est une constante égale à 100).

Pour chaque mot ajouté en français, dans le carnet, en lui ajoute en même temps son équivalent en anglais. Sachant que chaque mot en français qu'en anglais est une chaîne non vide de longueur maximale 15 lettres non accentuées.

Après l'élaboration du carnet, on veut traduire une phrase donnée en français (formée seulement par des lettres et par des espaces) en son équivalent en anglais. Dans le cas où l'un des mots ne figure pas dans le carnet, le mot en français va figurer dans la phrase en anglais mais entre deux accolades.

La phrase traduite doit être suivie par un message de succès ou un message d'échec indiquant le nombre de mots non traduits.

**NB** : On suppose que la phrase ne contient pas un espace au début, un espace à la fin et des espaces superflus (un seul espace sépare deux mots).

**Exemple** : Pour  $N=5$ , le carnet sera présenté comme suit :

FR	ecole	je	libre	suis	un
	0	1	2	3	4
ENG	School	i	free	am	a
	0	1	2	3	4

**FR** contient les mots en français et **ENG** contient les mots en anglais.

- Pour la phrase "ecole libre"  
Le résultat affiché sera : "school free : Traduction totale "
- Pour la phrase "je suis un etre libre"  
Le résultat affiché sera : "i am a {etre} free : Traduction partielle, 1 mot(s) non traduit(s)"

On vous demande d'élaborer :

- 1) un algorithme du programme principal, solution à ce problème, en le décomposant en modules,
- 2) l'algorithme de chaque module.





**EXERCICE 1**

- 1) Compléter le tableau ci-dessous par la valeur de la variable  $c$  après exécution de chaque séquence, et ce pour  $n=4$ .

Séquence	Valeur de la variable $c$
1	3
2	2
3	3

- 2) Donner le numéro de la séquence qui ne permet pas d'afficher le nombre de diviseurs. Justifier votre réponse.

la séquence 2 ne permet pas d'afficher le nombre de diviseurs:  
k commence de 2 à  $n \div 2 \implies$  On a éliminé les 2 diviseurs 1 et  $n$   
alors, il faut initialiser  $c$  par 2 ( $c \leftarrow 2$ )

**EXERCICE 2**

1)

```

Algorithme recherche
Début
  Écrire ("Donner le nom à chercher : ")
  Lire ( NOM )
  Existe  $\leftarrow$  Faux
   $i \leftarrow 0$ 
  Répéter
    Si ( $T[i] = \text{NOM}$ ) alors
      Existe  $\leftarrow$  Vrai
    Sinon
       $i \leftarrow i + 1$ 
  Finsi
  Jusqu'à ( Existe ) ou (  $i = N$  )
  Si ( Existe ) alors
    Écrire ("Le nom recherché existe dans ce tableau.")
  Sinon
    Écrire ("Le nom recherché n'existe pas dans ce tableau.")
  Finsi
Fin
  
```

2)

```

Algorithme recherche
Début
  Écrire ("Donner le nom à chercher : ")
  Lire ( NOM )
  Existe  $\leftarrow$  Faux
  Pour  $i$  de 0 à  $N-1$  faire
    Si ( $T[i] = \text{NOM}$ ) alors
      Ecrire("le numéro de ", $i$ )
      Existe  $\leftarrow$  Vrai
    Finsi
  Fin pour
  Si ( Existe = Faux ) alors
    Écrire ("Le nom recherché n'existe pas dans ce tableau.")
  Finsi
Fin
  
```

### **EXERCICE 3**

#### **Algorithme de la fonction somme**

Fonction somme(N :entier, x : entier) :réel

Debut

$S \leftarrow 0$

    Pour i de 0 a N faire

$S \leftarrow S + \text{Pow}(x,i)/\text{Fact}(i)$

    Fin pour

    Retourner S

Fin

TDOL

Objet	Type/Natur
S	Reel
i	Entier
Pow,Fact	fonction

#### **Algorithme de la fonction Pow**

Fonction Pow(x :entier, y : entier) :réel

Debut

$P \leftarrow 1$

    Pour i de 0 a y faire

$P \leftarrow P * x$

    Fin pour

    Retourner P

Fin

TDOL

Objet	Type/Natur
P	Entier
i	Entier

#### **Algorithme de la fonction Fact**

Fonction Fact(n : entier) :entier

Debut

$F \leftarrow 1$

    Si N=0 alors

        Retourner 1

    Si non

        Pour i de 1 a N+1 faire

$F \leftarrow F * i$

        Fin pour

        Retourner F

    Fin si

Fin

TDOL

Objet	Type/Natur
F	Entier
i	Entier

## **EXERCICE 4 : CORRECTION PROBLÈME**

### **Algorithme du programme principal**

Début Problème

N ← saisie()

remplirFR\_ENG(FR,ENG,N)

ch ← saisiech()

chtr ← Traduire(FR,ENG,ch,N)

Ecrire(chtr)

Fin

### **TDNT**

Type
Tab=tableau de NMAX chaine de caractères

### **TDOG**

Objet	Type/Natur
N	Entier
ch , chtr	Chaine
NMAX	Constante=100
FR,ENG	Tab
Saisie saisiech Traduire	Fonction
remplirFR_ENG	Procédures

### **Algorithme de la fonction saisie**

Fonction saisie() : entier

Debut

Repete

Ecrire("donner N ")

Lire(N)

Jusqu'à  $2 \leq N \leq NMAX$

Retourner N

Fin

TDOL

Objet	Type/Natur
N	Entier

### **Algorithme de la procedure remplirFR\_ENG**

Procedure remplirFR\_ENG(@FR:Tab,@ENG:Tab,N:entier)

Debut

Pour i de 0 a N-1 faire

Repete

Ecrire(" FR[" , i , "] = ")

Lire(FR[ i ])

Jusqu'à  $1 \leq \text{long}(\text{FR}[ i ]) \leq 15$  et Verif(FR[ i ])

Repeater

Ecrire(" ENG[" , i , "] = ")

Lire(ENG[ i ])

Jusqu'à  $1 \leq \text{long}(\text{ENG}[ i ]) \leq 15$  et Verif(ENG[ i ])

Fin pour

Fin

TDOL

Objet	Type/Natur
.i	Entier
Verif	Fonction

### Algorithme de la fonction Verif

fonction Verif(ch :chaîne) :booléen

Debut

Cond ← vrai

Pour i de 0 a long(ch)-1 faire

    Si NON("a" ≤ ch[i] ≤ "z" ou "A" ≤ ch[i] ≤ "Z")

        Cond ← faux

    Fin si

Fin pour

Retourner cond

Fin

TDOL

Objet	Type/Natur
i	Entier
Cond	Booléen

### Algorithme de la fonction Traduire

fonction Traduire(FR,ENG:Tab,ch:chaîne,N:entier):chaîne

Debut

Trad ← ""

K ← 0

Tant que ch ≠ "" faire

    P ← pos(" ",ch)

    Si P = -1 alors

        Mot ← ch

        Ch ← ""

    Sinon

        Mot ← sous\_chaine(ch,0,pos)

    Fin si

    Test ← faux

    Pour i de 0 a N faire

        Si Mot = FR[i] alors

            Trad = trad + " " + ENG[i]

            Test ← vrai

        Fin si

    Fin pour

    Si Test = faux alors

        Trad = trad + " " + "{" + Mot + "}"

        K ← K + 1

    Fin si

    Ch ← sous\_chaine(ch,P+1,long(ch))

Fin tant que

Trad ← sous\_chaine(trad,1,long(Trad))

Si k = 0 alors

    Retourner " « "+trad+" : Traduction totale »"

Sinon

    Retourner " « "+trad+" : Traduction partielle, "+convch(k)+ " mot(s) non traduit(s) »"

Fin si

Fin

TDOL

Objet	Type/Natur
Trad , Mot	chaines
P , k	Entiers
Test	Booleen

PROTOTYPE

2022

Épreuve pratique d'informatique

Sections :

Mathématiques, Sciences expérimentales et  
Sciences techniques

Durée : 1 heure

Coefficient : 0.5

**Important :** Dans le répertoire Bac2022, créez un dossier de travail ayant comme nom votre numéro d'inscription (6 chiffres) et dans lequel vous devez enregistrer, au fur et à mesure, tous les fichiers solutions de ce sujet.

Dans le cadre d'une campagne publicitaire, une société commerciale a décidé d'organiser chaque semaine un jeu de chance pour ses clients.

Le principe du jeu consiste à calculer le nombre de chance à partir du numéro de téléphone du client donné et d'afficher le message "*Félicitations, vous avez gagné.*" dans le cas où ce nombre est **premier** ou le message "*Désolé, vous n'avez pas gagné.*" dans le cas contraire.

Sachant que :

- Le numéro de téléphone devrait commencer par 2, 4, 5 ou 9.
- Le nombre de chance est la somme de chaque chiffre du numéro de téléphone multiplié par son indice avec l'indice du premier chiffre est 0.
- Un nombre premier est un nombre qui est divisible par 1 et par lui-même.

**Exemple :**

Donner le numéro : 29234560

Le programme affiche : *Désolé, vous n'avez pas gagné.*

En effet, le nombre de chance est égal à 99 qui n'est pas un nombre premier.

$99 = 2 * 0 + 9 * 1 + 2 * 2 + 3 * 3 + 4 * 4 + 5 * 5 + 6 * 6 + 0 * 7$  c'est la somme de chaque chiffre du numéro de téléphone multiplié par son indice :

Numéro téléphone	2	9	2	3	4	5	6	0
Indice	0	1	2	3	4	5	6	7

Ci-après, un algorithme de la fonction "Chance" à exploiter pour résoudre le problème posé.

**Fonction Chance (Ch : Chaîne) : Chaîne**

**DEBUT**

**Si NON** ( Estnum ( Ch ) **ET** long ( Ch ) = 8 **ET** Ch[0] ∈ [ "2", "4", "5", "9" ] ) **Alors**  
msg ← "Vérifier le numéro de téléphone !"

**Sinon**

msg ← "Désolé, vous n'avez pas gagné."

s ← 0

**Pour** i de 0 à long ( Ch ) - 1 **Faire**

s ← s + valeur ( Ch [i] ) \* i

**Fin Pour**

**Si** premier (s) **Alors**

msg ← "Félicitation, vous avez gagné."

**FinSi**

**FinSi**

**Retourner** msg

**FIN**





La société a décidé de créer l'interface graphique présentée ci-dessus, comportant les éléments suivants :

- Un label contenant le nom de la société.
- Un label demandant la saisie du numéro de téléphone.
- Une zone de saisie permettant la saisie du numéro de téléphone.
- Un bouton nommé "**Jouer**".
- Un label pour afficher un message.

### Société Commerciale

Entrer votre numéro de téléphone :

**Travail demandé :**

- 1) Concevoir une interface graphique comme illustré ci-dessus et l'enregistrer, dans votre dossier de travail, sous le nom "**Interface\_Jeu**".
- 2) Implémenter en Python la fonction "**Chance**" dans un programme et l'enregistrer sous le nom "**Jeu0**", dans votre dossier de travail.
- 3) Développer la fonction "**Premier**" permettant de vérifier si un nombre, passé comme paramètre, est premier ou non puis l'enregistrer dans votre dossier de travail sous le nom "**Jeu1**".
- 4) Dans le programme "**Jeu1**", ajouter les instructions permettant :
  - D'appeler l'interface graphique intitulée "**Interface\_Jeu**" en exploitant l'annexe ci-dessous.
  - D'implémenter un module "**Play**", qui s'exécute à la suite d'un clic sur le bouton "**Jouer**", permettant de récupérer le numéro de téléphone saisi puis d'exploiter la fonction "**Chance**" afin d'afficher le message retourné via un **label** de l'interface "**Interface\_Jeu**".

**Annexe**

```
from PyQt5.uic import loadUi
from PyQt5.QtWidgets import QApplication
.....
.....
app = QApplication([])
windows = loadUi ("Nom_Interface.ui")
windows.show()
windows.Nom_Bouton.clicked.connect (Nom_Module)
app.exec_()
```



*Exemples d'exécution :*

Société Commerciale

Entrer votre numéro de téléphone :

Société Commerciale

Entrer votre numéro de téléphone :

Verifier le numéro de téléphone !

Société Commerciale

Entrer votre numéro de téléphone :

Désolé, vous n'avez pas gagné.

Société Commerciale

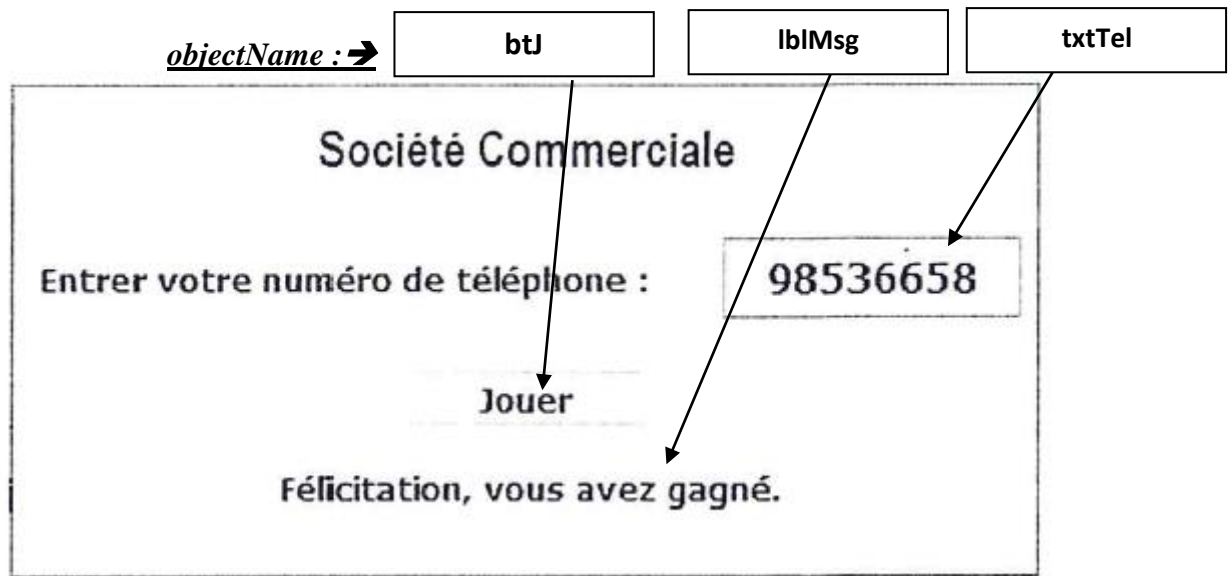
Entrer votre numéro de téléphone :

Félicitation, vous avez gagné.



# Correction TP prototype2022 4SC

## Qt Designer : (interface\_Jeu.ui)



## Thonny

```
from PyQt5.uic import loadUi
from PyQt5.QtWidgets import QApplication

def premier(n):
    cond=True
    if n <=1 :
        cond= False
    else:
        for i in range(2,n):
            if n % i==0:
                cond= False
    return cond

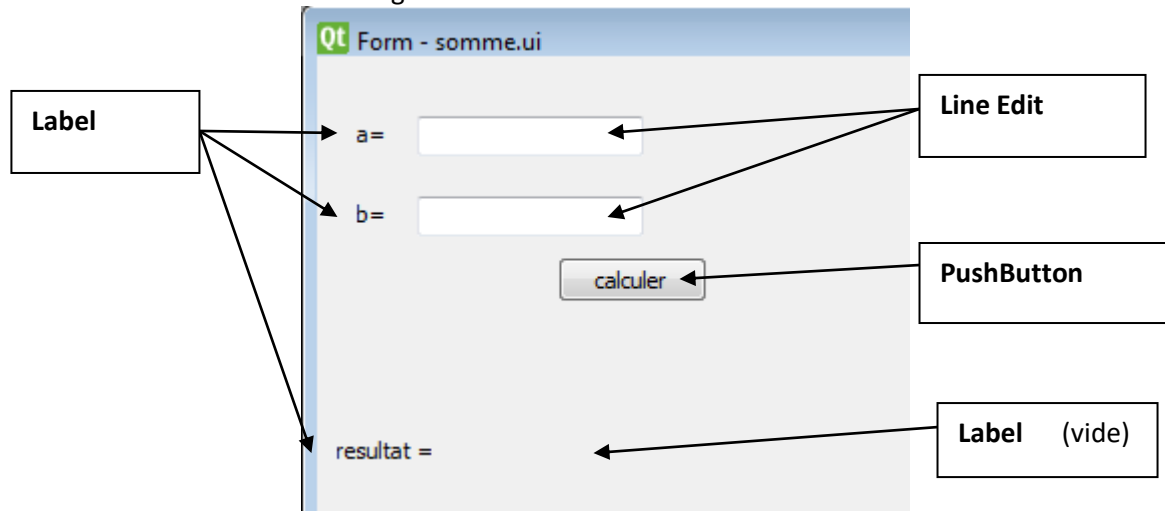
def chance(ch):
    if not (ch.isdigit() and len(ch)==8 and ch[0] in ['2','4','5','9']):
        msg='Vérifier le num de Tel !'
    else:
        msg="Désolé, vous n'avez pas gagné."
        s=0
        for i in range(len(ch)):
            s=s+int(ch[i])*i
        if premier(s):
            msg='Félicitation, vous avez gagné.'
    return msg

def play():
    x= windows.txtTel.text()
    windows.lblMsg.setText(chance(x))
app=QApplication([])
windows=loadUi("interface_Jeu.ui")
windows.show()
windows.btJ.clicked.connect(play)
app.exec()
```

# Problème N°1 Graphique

créer un dossier qui va contenir les deux fichiers : **somme.ui** et **calcul.py**

- Réaliser une interface graphique (sous **QtDesigner**) contenant deux entrées **a** et **b**, et un résultat qui va contenir la somme de a et b. et l'enregistrer sous le nom : **somme.ui**



- Lancer Thonny, puis écrire le programme (et l'enregistrer sous le nom : **calcul.py**) qui permet de calculer la somme de deux entiers a et b , en utilisant l'interface graphique créée précédemment pour saisir a et b et afficher le résultat.

**NB** : syntaxe générale pour un programme utilisant l'interface graphique :

```
from PyQt5.uic import loadUi

from PyQt5.QtWidgets import QApplication

#les fonction

def Play() :

    #partie données

    #partie traitement

    #partie résultat

#pp

app = QApplication([])

windows = loadUi ("somme.ui")

windows.show()

#bouton calculer

windows.calculer.clicked.connect (Play)

#execution de l'application

app.exec_()
```



## CORRECTION

```
prog.py × | calcul.py × |
1  from PyQt5.uic import loadUi
2  from PyQt5.QtWidgets import QApplication
3  #les fonction
4  def somme(a,b):
5      s=a+b
6      return s
7
8  def Play() :
9      #partie données
10     a=windows.a.text()
11     a=int(a)
12     b=windows.b.text()
13     b=int(b)
14     #partie résultat
15     r=somme(a,b)
16     r=str(r)
17     windows.res.setText(r)
18 #pp
19 app = QApplication([])
20 windows = loadUi ("somme.ui")
21 windows.show()
22 #bouton calcul
23 windows.calcul.clicked.connect (Play)
24 #execution de l'application
25 app.exec_()
```

## Problème N°2 Graphique

Dans le but de sécuriser les messages à envoyer, on peut faire appel à une méthode de cryptage.

Une des méthodes utilisées consiste à remplacer chaque lettre du message à crypter par celle qui la suit de  $p$  positions dans l'alphabet français, où  $p$  désigne le nombre de mots du message.

**NB :**

- On suppose que le caractère qui suit la lettre "Z" est le caractère "A" et celui qui suit la lettre "z" est le caractère "a".
- Le caractère espace ne subit aucune modification.
- Le code ASCII de la lettre "a" est égal à 97 et celui de la lettre "A" est égal à 65.

**Exemple :**

Pour le message "Examen Pratique En Informatique"

Etant donné que le message à crypter est formé de 4 mots, pour la lettre alphabétique "E" par exemple, elle sera remplacée par "I" car en ajoutant au code Ascii de "E" qui est 69 la valeur 4, on obtient 73 qui est le code Ascii de "I".

En continuant à appliquer ce principe de codage, le message crypté sera : "Ibeqir Tvexmuyi Ir Mrjsvqexmuyi"

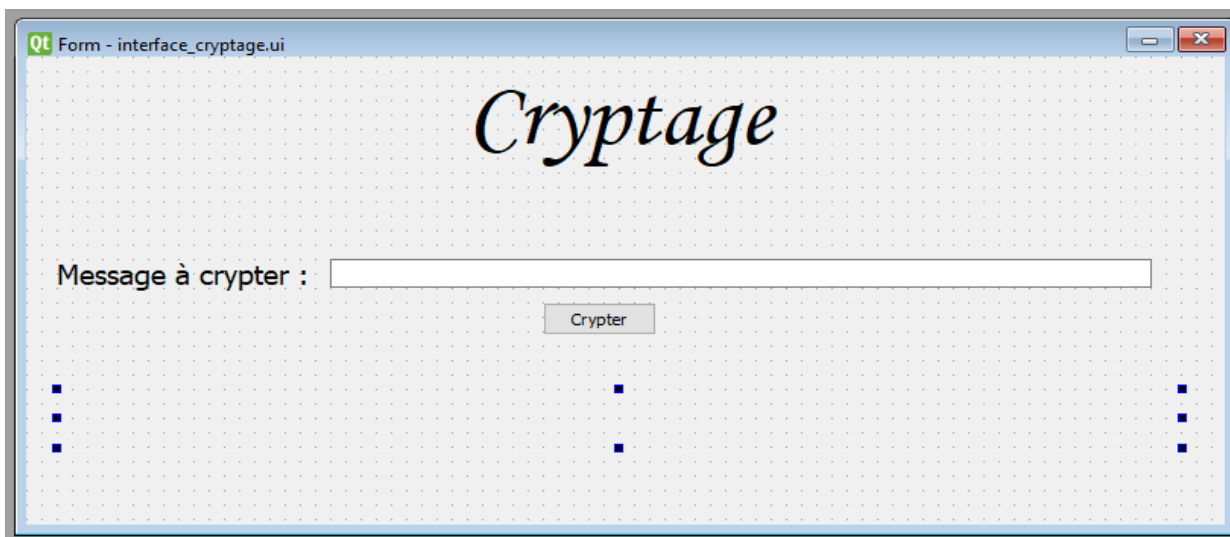
**NB :** Le message peut contenir des espaces superflus (inutiles).

```
fonction cryptage(M:chaine , p:entier) : chaine
debut
ch←""
l←long(M)
pour i de 0 a l-1 faire
    si M[i]= " " alors
        ch←ch+" "
    sinon si NON( 65<= ord(M[i])+p <=90 ou 97<= ord(M[i])+p <=122) alors
        ch←ch+chr(ord(M[i])+p-26)
    sinon
        ch←ch+chr(ord(M[i])+p)
    fin si
fin pour
retourner ch
fin
```

La fonction **cryptage** :

Pour cela on a décidé de créer l'interface graphique présentée ci-dessus, comportant les éléments suivants :

- Un label contenant « Cryptage ».
- Un label demandant la saisie d'un message à crypter.
- Une zone de saisie permettant la saisie de ce message.
- Un bouton nommé "Crypter".
- Un label pour afficher un message.



### Travail demandé :

- 1) Concevoir une interface graphique comme illustré ci-dessus et l'enregistrer, dans votre dossier de travail, sous le nom "**Interface\_cryptage**".
- 2) Implémenter en Python la fonction "**cryptage**" dans un programme et l'enregistrer sous le nom "**Prog\_Cryptage**", dans votre dossier de travail.
- 3) Développer la fonction **valide** qui permet de vérifier si le message M est non vide et formé uniquement par des lettres et des espaces.
- 4) Développer la fonction **nbMot** qui permet de déterminer le nombre de mots dans le message M à crypter en prenant en considération la possibilité d'existence des espaces superflus (inutiles).
- 5) Dans le programme "**Prog\_Cryptage**", ajouter les instructions permettant :
  - D'appeler l'interface graphique intitulée "**Interface\_cryptage**" en exploitant l'annexe ci-dessous.
  - D'implémenter un module "**affiche**", qui s'exécute à la suite d'un clic sur le bouton "**Crypter**", permettant de récupérer le message saisi puis d'exploiter les 3 fonctions "**cryptage**", "**valide**" et "**nbMot**" afin d'afficher le message retourné via un **label** de l'interface "**Interface\_cryptage**".

**Annexe**

```

from PyQt5.uic import loadUi
from PyQt5.QtWidgets import QApplication
.....
.....
app = QApplication([])
windows = loadUi ("Nom_Interface.ui")
windows.show()
windows.Nom_Bouton.clicked.connect (Nom_Module)
app.exec_()

```

Si le message saisi est invalide on va afficher dans le label "**Vérifier votre message !**".

## Exemple d'exécution



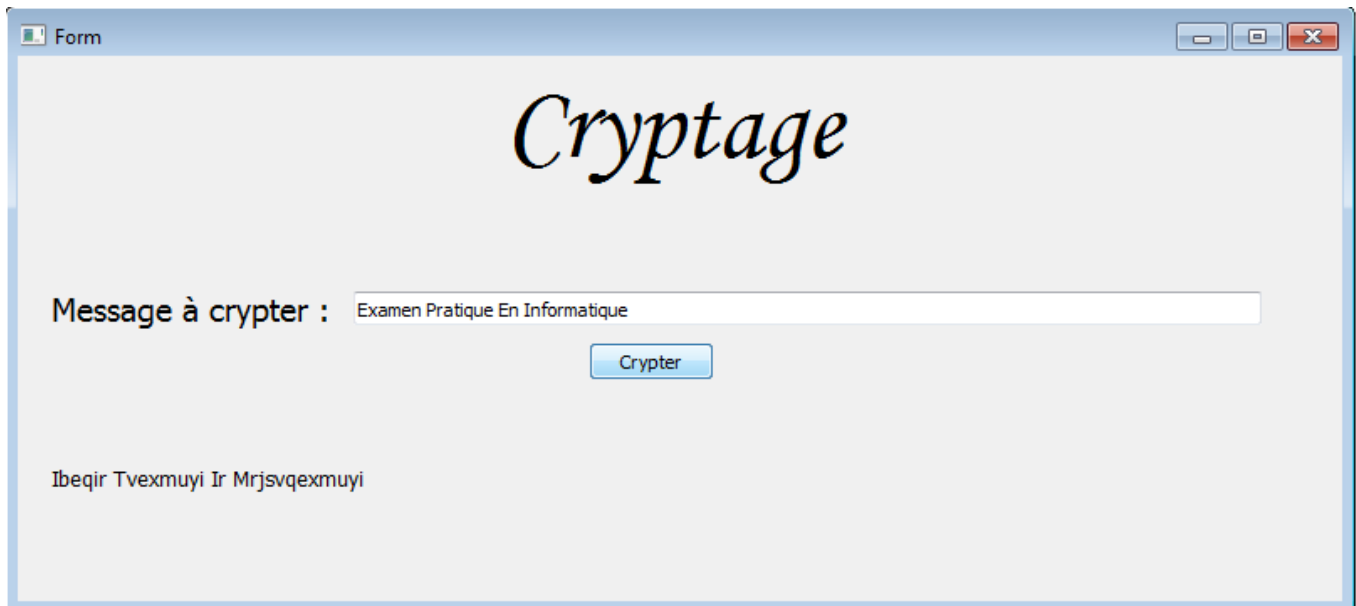
Form

# Cryptage

Message à crypter : Examen Bac2022

Crypter

verifier votre message à crypter !



Form

# Cryptage

Message à crypter : Examen Pratique En Informatique

Crypter

Ibeqir Tvexmuyi Ir Mrjsvqexmuyi



## CORRECTION

```
1 from PyQt5.uic import loadUi
2 from PyQt5.QtWidgets import QApplication
3 def cryptage(M,p):
4     ch=''
5     l=len(M)
6     for i in range(l):
7         if M[i]==' ':
8             ch=ch+' '
9         elif not(65<=ord(M[i])+p<=90 or 97<=ord(M[i])+p<=122):
10            ch=ch+chr(ord(M[i])+p-26)
11        else:
12            ch=ch+chr(ord(M[i])+p)
13    return ch
14 def valide(ch):
15     test=True
16     l=len(ch)
17     if ch==' ':
18         test=False
19     else:
20         for i in range(l):
21             if not( 'A'<=ch[i].upper()<='Z' or ch[i]==' '):
22                 test=False
23    return test
24 def nbMot(ch):
25     s=1
26     if ch[0]==' ':
27         s=0
28     for i in range(len(ch)-1):
29         if ch[i]==' ' and ch[i+1]!=' ':
30             s=s+1
31    return s
32 def affiche():
33     ch=windows.txtMsg.text()
34     print(valide(ch))
35     if valide(ch):
36         p=nbMot(ch)
37         msg=cryptage(ch,p)
38         windows.lblMsg.setText(msg)
39     else:
40         windows.lblMsg.setText('verifier votre message à crypter')
41 app=QApplication([])
42 windows=loadUi("Interface_cryptage.ui")
43 windows.show()
44 windows.btC.clicked.connect(affiche)
45 app.exec()
```

## Problème N°3 Graphique

Enregistrer les fichiers résultats dans un même dossier appelé « app\_jeu » dans votre dossier de travail dans la racine C :

### Sujet :

La Télécom décide faire une application qui offre des cadeaux à ces clients chaque mois selon un critère. Ce mois elle va donner les cadeaux aux numéros pairs.

L'application demande au client d'entrer son numéro de téléphone

Sachant que :

- Le numéro de téléphone doit commencer par 95 ,97 ou 99
- Et il doit être composé de 8 chiffres.

Le résultat affiché est :

- « vérifier le numéro de téléphone » si le numéro est erroné
- « Félicitation, vous avez gagné » si le numéro de téléphone est pair
- « Désolé, vous n'avez pas gagné » sinon

Ci\_ après la fonction «jeux » à exploiter pour résoudre le problème :

**Fonction** jeux(ch :chaine) :chaine

Début

Si **NON**(Estnum(ch) ET long(ch)=8 ET ch[0]="9" ET ch[1] ∈ ["5","7","9"]) alors

msg←"vérifier le numéro de téléphone"

sinon

x←valeur(ch)

si **pair**(x) Alors

msg←"Félicitation, vous avez gagné"

sinon

msg←"Désolé, vous n'avez pas gagné"

Fin si

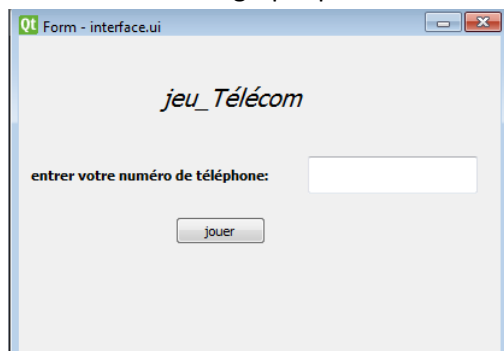
Fin si

**Retourner** msg

Fin

Travail demandé :

- Réaliser l'interface graphique suivante et l'enregistrer sous le nom « **interface.ui** »



- Ecrire le programme qui résous ce problème et l'enregistrer sous le nom « prog.py »

## CORRECTION

```
rog.py × calcul.py ×
1 from PyQt5.QtWidgets import *
2 from PyQt5.uic import *
3 def pair(x):
4     if x%2 ==0:
5         return True
6     else:
7         return False
8
9 def jeux(ch):
10    if not(ch.isdigit() and len(ch)==8 and ch[0]=='9' and ch[1] in ["5","7","9"])
11        msg='verifier le num de tel'
12    else:
13        x=int(ch)
14        if pair(x):
15            msg="Félicitation, vous avez gagné"
16        else:
17            msg="désolé, vous n'avez pas gagné"
18
19    return msg
20 def play():
21    ch=windows.tel.text()
22    r=jeux(ch)
23    windows.res.setText(r)
24 #pp
25 app=QApplication([])
26 windows=loadUi('interface.ui')
27 windows.show()
28 windows.jouer.clicked.connect(play)
29 app.exec_()
```

## Problème N°4 Graphique

Un nombre est dit super premier s'il est premier et si, en supprimant des chiffres à partir de sa droite, le nombre restant est aussi premier.

Exemple :

Le nombre 59399 est super premier car les nombres 59399, 5939, 593, 59, 5 sont tous premiers.

ci-après l'algorithme de la fonction aff\_superpremier :

Fonction aff\_superpremier (ch :chaine) :chaine

DEBUT

Si NON(Estnum(ch) Alors

    msg ← "Vérifier la saisie"

Sinon Si **superpremier(ch)** Alors

    msg ← "le nombre est super premier"

Sinon

    msg ← "le nombre n'est pas super premier"

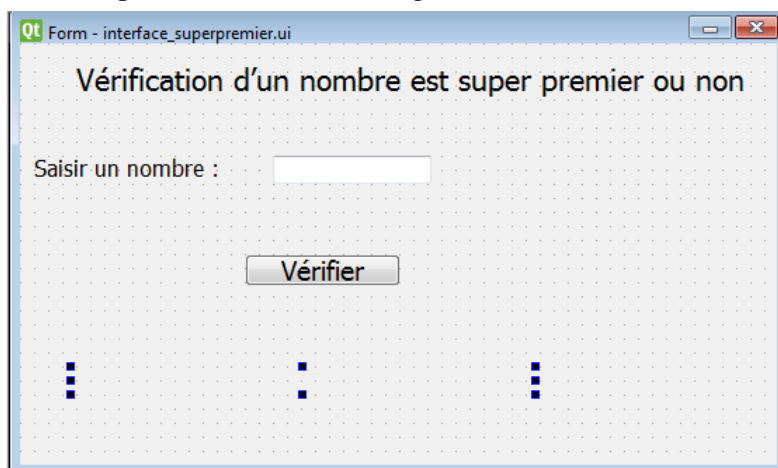
Finsi

Retourner msg

FIN

On veut créer l'interface graphique présentée ci-dessus, comportant les éléments suivants :

- Un label contenant le titre suivant « **Vérification d'un nombre est superpremier ou non** »
- Un label contenant le texte suivant « **Saisir un nombre** »
- Une zone de saisie permettant la saisie d'un nombre
- Un bouton nommé "**Vérifier**"
- Un label pour afficher un message.





### Travail demandé :

- 1) Concevoir une interface graphique comme illustré ci-dessus et l'enregistrer, dans votre dossier de travail, sous le nom "**Interface\_superpremier**"
- 2) Implémenter en python la fonction " aff\_superpremier " dans programme et l'enregistrer sous le nom "**superpremier\_1**"
- 3) Développer la fonction **superpremier** permettant de vérifier si un nombre, passé en paramètre, est superpremier ou non puis l'enregistrer dans votre dossier de travail sous le nom " **superpremier\_2**".
- 4) Dans le programme " **superpremier\_2**", ajouter les instructions permettant :
  - D'appeler l'interface intitulée "**Interface\_superpremier**" en exploitant l'annexe ci-dessous.
  - D'implémenter un module "**Verif**", qui s'exécute à la suite d'un clic sur le bouton "**Vérifier**" permettant de récupérer le nombre saisi puis exploiter la fonction **aff\_superpremier** afin d'afficher le message retourné via un label de l'interface "**Interface\_superpremier**"

```
Annexe  
from PyQt5.uic import loadUi  
from PyQt5.QtWidgets import QApplication  
.....  
.....  
app = QApplication([])  
windows = loadUi ("Nom_Interface.ui")  
windows.show()  
windows.Nom_Bouton.clicked.connect (Nom_Module)  
app.exec_()
```

## Problème N°5 Graphique

Un nombre fort est un nombre spécial dont la somme de la factorielle de ses chiffres est égale au nombre d'origine.

Exemple :

145 est nombre fort puisque  $1! + 4! + 5! = 145$

ci-après l'algorithme de la fonction `aff_fort` :

Fonction `aff_fort` (ch :chaîne) :chaîne

DEBUT

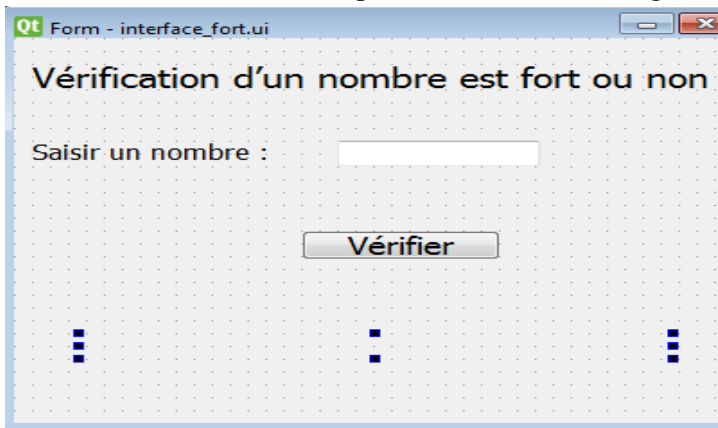
```
Si NON(Estnum(ch)) Alors
    msg ← "Vérifier la saisie"
Sinon Si Fort(ch) Alors
    msg ← "le nombre est fort"
Sinon
    msg ← "le nombre n'est pas fort"

Finsi
Finsi
Retourner msg
```

FIN

On veut créer l'interface graphique présentée ci-dessus, comportant les éléments suivants :

- Un label contenant le titre suivant « **Vérification d'un nombre est fort ou non** »
- Un label contenant le texte suivant « **Saisir un nombre** »
- Une zone de saisie permettant la saisie d'un nombre
- Un bouton nommé "**Vérifier**"
- Un label pour afficher un message.



```
Annexe
from PyQt5.uic import loadUi
from PyQt5.QtWidgets import QApplication
.....
.....
app = QApplication([])
windows = loadUi("Nom_Interface.ui")
windows.show()
windows.Nom_Bouton.clicked.connect(Nom_Module)
app.exec_()
```

**Travail demandé :**

- 5) Concevoir une interface graphique comme illustré ci-dessus et l'enregistrer, dans votre dossier de travail, sous le nom "**Interface\_fort**"
- 6) Implémenter en python la fonction "`aff_fort`" dans programme et l'enregistrer sous le nom "**fort\_1**"
- 7) Développer la fonction **Fort** permettant de vérifier si un nombre, passé en paramètre, est fort ou non puis l'enregistrer dans votre dossier de travail sous le nom "**fort\_2**".
- 8) Dans le programme "**fort\_2**", ajouter les instructions permettant :
  - D'appeler l'interface intitulée "**Interface\_fort**" en exploitant l'annexe ci-dessous.
  - D'implémenter un module "**Verif**", qui s'exécute à la suite d'un clic sur le bouton "**Vérifier**" permettant de récupérer le nombre saisi puis exploiter la fonction **aff\_fort** afin d'afficher le message retourné via un label de l'interface "**Interface\_fort**"

## CORRECTION

```
1 from PyQt5.uic import loadUi
2 from PyQt5.QtWidgets import QApplication
3
4 def aff_fort (ch ) :
5     if not(ch.isnumeric()):
6         msg= 'Vérifier la saisie'
7     elif fort(ch) :
8         msg='le nombre est fort'
9     else:
10        msg='le nombre n'est pas fort'
11    return msg
12
13 def fact(nb):
14     f=1
15     if nb==0:
16         return 1
17     else:
18         for i in range(1,nb+1):
19             f=f*i
20         return f
21
22 def fort(ch):
23     l=len(ch)
24     s=0
25     for i in range(l):
26         s=s+fact(int(ch[i]))
27     if s==int(ch):
28         return True
29     else:
30         return False
31
32
33 def verif():
34     ch=windows.txtNbr.text()
35     msg=aff_fort (ch )
36     windows.lblMsg.setText(msg)
37
38 app=QApplication([])
39 windows=loadUi("Interface_fort.ui")
40 windows.show()
41 windows.btV.clicked.connect(verif)
42 app.exec()
```

## Problème N°6 Graphique

Une société de télécommunication téléphonique **3X** veut faire gagner à ses clients un bonus de 100 points pour les clients dont la recharge téléphonique portable est gagnante, si le numéro de 13 chiffres vérifie les contraintes suivantes :

- Le nombre composé des trois premiers chiffres du code est un nombre premier.
- Le nombre composé des cinq chiffres suivants, converti en **binaire (Base 2)**, doit contenir plus que cinq zéros (0).
- Le nombre composé des cinq chiffres restants, converti en **hexadécimale (base 16)**, doit contenir le caractère "D" ou "E" ou "F".

Le principe consiste à vérifier le numéro de recharge du client et d'afficher le message "**Félicitations, vous avez gagné un bonus de 100 points.**" dans le cas où ce nombre vérifie les trois conditions ci-dessus ou le message "**Désolé, vous n'avez pas gagné.**" dans le cas contraire.

### Exemple :

Numéro de recharge **1971546215309**

- Le nombre de 3 premiers chiffres **197** est premier
- La conversion des 5 chiffres suivants **15462** en binaire **0011110001100110** contient plus que 5 zéros
- La conversion des 5 chiffres suivants **15309** en hexadécimale **3BCD** contient le caractère D

Le message à afficher : "**Félicitations, vous avez gagné un bonus de 100 points.**"

Ci-après, un algorithme de la fonction "**convB**" à exploiter pour résoudre le problème posé qui permet de convertir un nombre décimal(entier) en une chaîne binaire ou en une chaîne hexadécimale en indiquant la base de conversion B (B=2 pour Binaire, B=16 pour hexadécimale)

**Fonction convB (n : entier, B : entier) : Chaîne**

**DEBUT**

**Ch** ← ""

**Repete**

**r** ← n mod B

**r** < 10 **alors**

**X** ← convch(r)

**Sinon**

**X** ← chr(r+55)

**Finsi**

**Ch** ← X+Ch

**n** ← n div B

**Jusqu'à n=0**

**Retourner Ch**

**FIN**



La société a décidé de créer l'interface graphique présentée ci-dessus, comportant les éléments suivants :

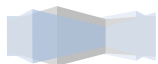
- Un label contenant le nom de la société.
- Un label demandant la saisie du numéro de recharge.
- Une zone de saisie permettant la saisie du numéro recharge.
- Un bouton nommé "**verifier**".
- Un label pour afficher un message.



### Travail demandé :

- 1) Concevoir une interface graphique comme illustré ci-dessus et l'enregistrer, dans votre dossier de travail, sous le nom "**Interface\_play**".
- 2) Implémenter en Python la fonction "**convB**" dans un programme et l'enregistrer sous le nom "**play**", dans votre dossier de travail.
- 3) Développer la fonction "**Premier**" permettant de vérifier si un nombre, passé comme paramètre, est premier ou non .
- 4) Dans le programme "**play**", ajouter les instructions permettant :  
D'appeler l'interface graphique intitulée "**Interface\_paly**" en exploitant l'annexe ci-dessous.  
D'implémenter un module "**affiche**", qui s'exécute à la suite d'un clic sur le bouton "**verifier**", permettant de récupérer le numéro de recharge saisi puis d'exploiter la fonction "**convB**" et la fonction "**premier**" afin d'afficher le message retourné via un **label** de l'interface "**Interface\_play**".

```
Annexe
from PyQt5.uic import loadUi
from PyQt5.QtWidgets import QApplication
.....
.....
app = QApplication([])
windows = loadUi ("Nom_Interface.ui")
windows.show()
windows.Nom_Bouton.clicked.connect (Nom_Module)
app.exec_()
```



# CORRECTION

```
1 from PyQt5.uic import loadUi
2 from PyQt5.QtWidgets import QApplication
3 def convB(n,b):
4     ch=""
5     while n!=0 :
6         r=n%b
7         if r<10 :
8             x=str(r)
9         else :
10            x=chr(r+55)
11            ch=x+ch
12            n=n//b
13    return ch
14 def premier(n):
15    cond=True
16    if n<=1:
17        cond=False
18    else:
19        for i in range (2,n) :
20            if n%i==0 :
21                cond=False
22    return cond
23 def compteur(ch):
24    s=0
25    for i in range (0,len(ch)) :
26        if ch[i]=="0" :
27            s=s+1
28    return s
29 def recharge(ch):
30    n1=int(ch[0:3])
31    n2=int(ch[3:8])
32    ch2=convB(n2,2)
33    n3=int(ch[8:])
34    ch3=convB(n3,16)
35    if premier(n1) and compteur(ch2)>5 and (ch3.find('D')!=-1 or ch3.find('E')!=-1 or ch3.find('F')!=-1 ):
36        msg="Félicitation vous avez gagné"
37    else :
38        msg="Désolé, vous n'avez pas gagné"
39    return msg
40 def play():
41    num=windows.txtR.text()
42    windows.lblMsg.setText(recharge(num))
43 app=QApplication([])
44 windows=loadUi("Interface_play.ui")
45 windows.show()
46 windows.btV.clicked.connect(play)
47 app.exec()
48
49
13
```

