Épreuve d'Informatique – Session 2020 – Filière MP

Les candidats sont informés que la précision des raisonnements algorithmiques ainsi que le soin apporté à la rédaction et à la présentation des copies seront des éléments pris en compte dans la notation. Il convient en particulier de rappeler avec précision les références des questions abordées. Si, au cours de l'épreuve, un candidat repère ce qui peut lui sembler être une erreur d'énoncé, il le signale sur sa copie et poursuit sa composition en expliquant les raisons des initiatives qu'il est amené à prendre.

Remarques générales :

- ✓ Cette épreuve est composée d'un exercice et de trois parties toutes indépendantes ;
- ✓ Toutes les instructions et les fonctions demandées seront écrites en Python ;
- ✓ Les questions non traitées peuvent être admises pour aborder les questions ultérieures ;
- ✓ Toute fonction peut être décomposée, si nécessaire, en plusieurs fonctions.

≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈≈

Exercice : (4 points)

Les coefficients binomiaux

Un **coefficient binomial** est défini pour deux entiers positifs **n** et **k** tels que **n≥k**. C'est le nombre de parties de **k** éléments dans un ensemble de **n** éléments. On le note : $\binom{n}{k}$, et sa valeur est calculée par la formule suivante :

$$\binom{n}{k} = \frac{\mathbf{n} * (\mathbf{n}-\mathbf{1}) * (\mathbf{n}-\mathbf{2}) * \ldots * (\mathbf{n}-(\mathbf{k}-\mathbf{1}))}{k!}$$

1 pt **Q1-** Écrire la fonction **`fact(p)`** qui reçoit en paramètre un entier positif **p**, et qui retourne la valeur de **factorielle p** : **p! = 1 * 2 * 3 * … * (p−1) * p**.

NB : La fonction **fact(0)** retourne 1

1 pt **Q2-** Écrire la fonction **`produit(n,k)`** qui reçoit en paramètres deux entiers positifs **n** et **k** tels que **n≥k**, et qui retourne la valeur du produit : **n * (n−1) * (n−2) * … * (n− (k−1))**

0.75 pt **Q3-** Écrire la fonction **`binomial(n,k)`** qui reçoit en paramètres deux entiers positifs **n** et **k** tels que **n≥k**, et qui retourne la valeur du coefficient binomial $\binom{n}{k}$.

Exemple : La fonction **binomial (6, 3)** retourne le nombre **20**

1.25 pt **Q4-** Écrire la fonction **`liste_binomiaux(n)`** qui reçoit en paramètre un entier positif **n**, et qui retourne la liste des coefficients binomiaux $\binom{n}{k}$ tel que : **k = 0, 1, 2, 3, …, n**

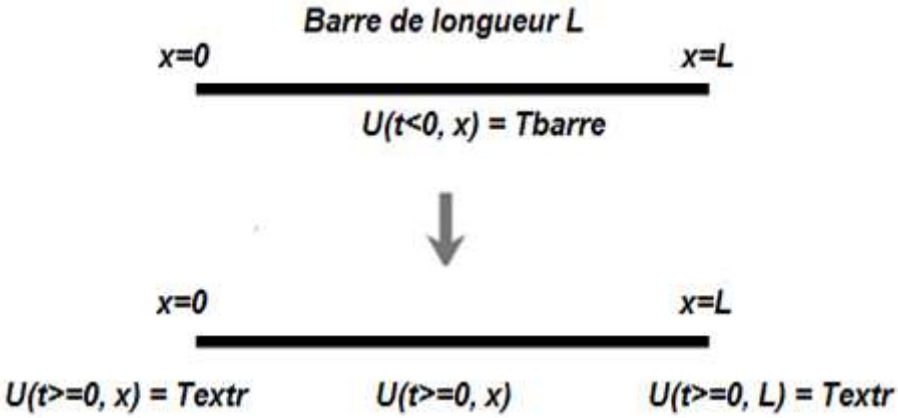
Exemple : La fonction **liste_binomiaux (6)** retourne la liste **[1, 6, 15, 20, 15, 6, 1]**

Partie I : Calcul numérique

Équation de la diffusion thermique

On considère une barre solide de longueur L, de coefficient de diffusion thermique D.

La barre est initialement "préparée" dans un état de température $Tbarre$. Les deux extrémités de la barre sont maintenues à une température extérieure constante $Textr$.



L'équation de la diffusion thermique à une dimension, est la suivante :

$$(E) \qquad \frac{dU(x,t)}{dt} = D * \frac{d^2U(x,t)}{dx^2}$$

Avec : $U(x, t)$ est la température de la position x dans la barre, à un instant t.

L'équation (E) admet une solution unique :

$$U(x,t) = Textr + c * \sum_{k=1}^{+\infty} \frac{1}{k}(1 - cos(k\pi)) * sin\left(\frac{k\pi x}{L}\right) * e^{-\left(\frac{k\pi}{L}\right)^2 * D * t}$$

$$avec : \quad c = \frac{2}{\pi}(Tbarre - Textr)$$

On suppose que les variables globales suivantes, sont déclarées et initialisées par les valeurs suivantes :

- ✓ $L = 1.0$ Longueur de la barre
- ✓ $D = 0.2$ Coefficient de diffusion thermique
- ✓ $T = 1.4$ Durée totale de l'évolution de la température
- ✓ $Tbarre = 100.0$ Température de la barre
- ✓ $Textr = 20.0$ Température extérieure

Dans cette partie, on suppose que les modules **numpy** et **matplotlib.pyplot** sont importés :

```
import numpy np
import matplotlib.pyplot as pl
```

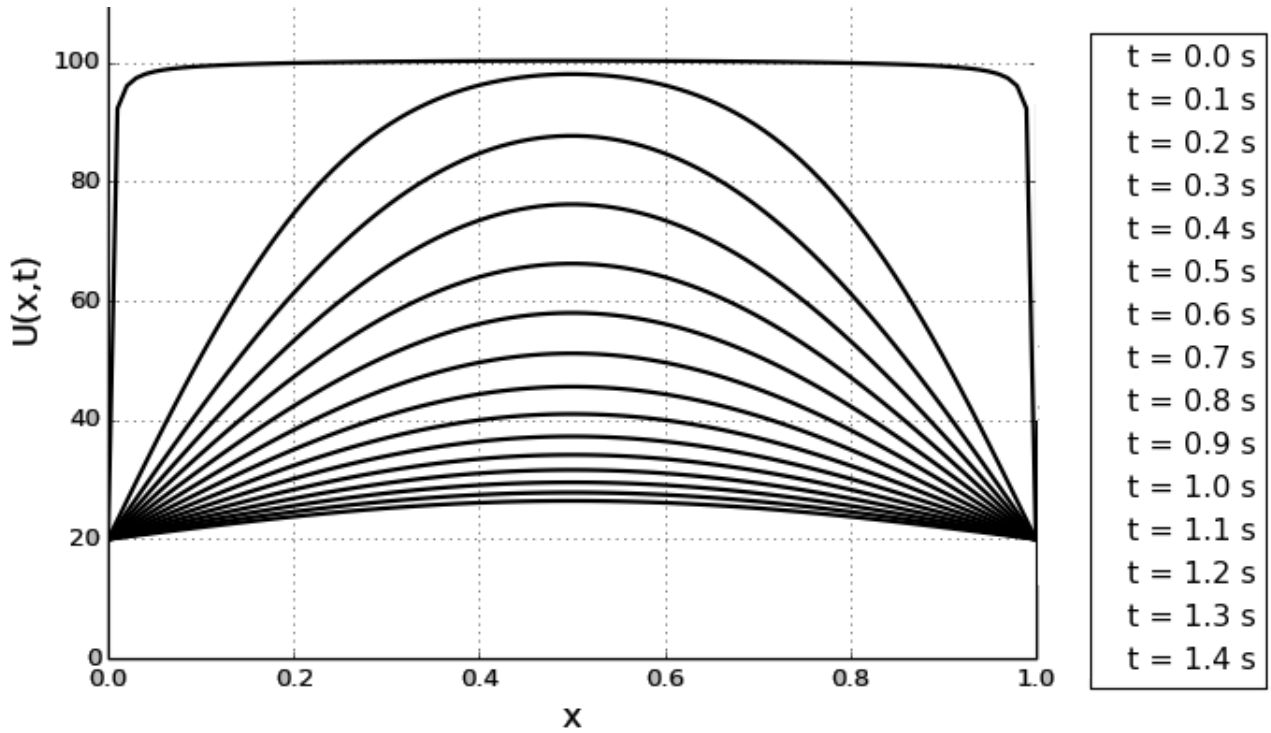
Q.1- Écrire, en Python, la fonction f définie par :

$$f(x,k,t) = \frac{1}{k}(1 - cos(k\pi)) * sin\left(\frac{k\pi x}{L}\right) * e^{-\left(\frac{k\pi}{L}\right)^2 * D * t}$$

Q.2- Écrire la fonction `U(x,t)` qui reçoit en paramètres deux réels x et t. La fonction retourne la valeur de $U(x,t)$ qui correspond à la somme partielle d'indice $m=200$:

$$U(x,t) = Textr + \frac{2}{\pi}(Tbarre - Textr) * \sum_{k=1}^{m} f(x,k,t)$$

Q.3- Écrire le programme permettant de tracer la représentation graphique suivante, qui représente l'évolution de la température dans les points x de la barre, à intervalle de temps égal à $0.1s$, sachant que la barre est subdivisée en 100 points, et l'indice de la somme partielle est $m=200$.



NB : Chaque courbe représente la température à chaque point x de la barre à un instant t.

Partie II : Base de donnés et langage SQL

Classement des candidats

Une grande école d'ingénieurs organise un concours au profit des étudiants (candidats), qui veulent y accéder pour suivre leurs études supérieures. Le concours est composé de plusieurs épreuves : une seule épreuve par matière.

Après le passage des épreuves, une note totale est calculée pour chaque candidat. Ensuite les candidats sont classés dans l'ordre décroissant de la note totale.

*NB : Une note inférieure strictement à **5.0** est une note éliminatoire du classement final. Si un candidat possède au moins une note éliminatoire, alors ce candidat sera exclu du classement final.*

L'école d'ingénieurs utilise une base de données relationnelle composée de trois tables :

Candidats	
numéro	*(entier)*
nom	*(texte)*
prénom	*(texte)*

Notes	
numC	*(entier)*
codM	*(texte)*
note	*(réel)*

Matières	
code	*(texte)*
nom	*(texte)*
coeff	*(entier)*

La table **'Candidats'** contient les numéros, les noms et les prénoms des candidats. Le champ **numéro** est la clé primaire dans cette table.

Exemples :

numéro	nom	prénom
416	Jarfaoui	Hicham
70	Hilal	Samira
162	Senhaji	Amal
23	Bakouri	Ahmed
…	**…**	**…**

La table **'Matières'** contient les codes, les noms et les coefficients correspondants à chaque matière. Le champ **code** est la clé primaire dans cette table.

Exemples :

code	nom	coeff
M	Mathématiques	14
P	Physique	10
SI	Sciences de l'ingénieur	6
Ch	Chimie	3
…	**…**	**…**

La table **'Notes'** contient, pour chaque candidat, la note correspondante à chaque matière. Les champs **numC** et **codM** sont deux clés étrangères, qui font respectivement référence, aux champs **numéro** et **code** des tables 'Candidats' et 'Matières'.

Exemples :

numC	codM	note
70	**SI**	14,50
162	**SI**	17,00
416	**SI**	12,25
70	**M**	16,00
162	**M**	13,50
416	**M**	08,35
70	**P**	04,75
162	**P**	11,05
416	**P**	10,10
…	**…**	**…**

Q.1 – Déterminer la clé primaire de la table 'Notes', et justifier votre réponse.

Q.2 – Écrire, en algèbre relationnelle, une requête qui donne les numéros, les noms et prénoms de tous les candidats, qui ne possèdent pas de note éliminatoire, en matière de code **'M'**.

Q.3 – Écrire la requête précédente (**Q.2**) en langage SQL.

Q.4 – Écrire, en langage SQL, une requête qui donne les noms des matières, la note maximale et la note minimale de chaque matière, triés dans l'ordre décroissant de la moyenne des notes de chaque matière.

Q.5 – Écrire, en langage SQL, une requête qui donne le compte des candidats qui sont exclus du classement final.

Q.6– Pour les candidats non exclus du classement final, la **note totale** de chaque candidat est calculée par la formule suivante :

$$\boldsymbol{note\ totale} = \sum (\boldsymbol{note} * \boldsymbol{coefficient})$$

La note d'une matière est multipliée par le coefficient correspondant à cette matière.

Écrire, en langage SQL, une requête qui donne le numéro, le nom, le prénom et la note totale de chaque candidat non exclu, ayant la note totale supérieure strictement à ***1000.0***, triés dans l'ordre décroissant de la note totale.

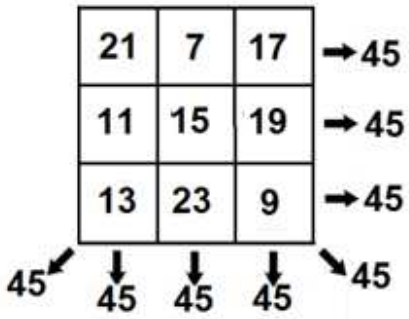
Partie III : Problème

Carré magique

On considère un entier **n** strictement positif. Un **carré magique** d'ordre **n** est une matrice carrée d'ordre **n** (**n** lignes et **n** colonnes), qui contient des nombres entiers strictement positifs. Ces nombres sont disposés de sorte que les sommes sur chaque ligne, les sommes sur chaque colonne et les sommes sur chaque diagonale principale soient égales. La valeur de ces sommes est appelée : **constante magique**.

Exemple :

Carré magique d'ordre **3,** sa constante magique **45**



Représentation d'une matrice carrée en Python :

Pour représenter une matrice carrée d'ordre **n** (**n** lignes et **n** colonnes), on utilise une liste qui contient **n** listes, toutes de même longueur **n**.

Exemple :

$$\begin{vmatrix} 4 & 7 & 10 & 3 \\ 3 & 2 & 9 & 6 \\ 13 & 0 & 5 & 8 \\ 7 & 1 & 6 & 25 \end{vmatrix}$$

Cette matrice carrée d'ordre **4** est représentée par la liste **M**, composée de **4** listes de taille **4** chacune :

```
M = [[4,7,10,3],[3,2,9,6],[13,0,5,8],[7,1,6,25]]
```

`M[i]` est la liste qui représente la ligne d'indice **i** dans **M**.

Exemples :

- **M[0]** est la liste **[4, 7, 10, 3]**
- **M[2]** est la liste **[13, 0, 5, 8]**

`M[i][j]` est l'élément à la **i**ème ligne et la **j**ème colonne, dans **M**

Exemples :

- **M[0][1]** est l'élément **7**
- **M[2][1]** est l'élément **0**

I.- Opérations sur une matrice carrée

Q.1- Écrire la fonction **`somme_ligne(M,i),`** qui reçoit en paramètres une matrice carrée **M** contenant des nombres, et un entier **i** qui représente l'indice d'une ligne dans **M**. La fonction retourne la somme des nombres de la ligne d'indice **i** dans **M**.

Exemple :

La fonction **somme_ligne (M, 1)** retourne la somme 3+2+9+6 = 20

Q.2- Écrire la fonction **`somme_colonne(M,j),`** qui reçoit en paramètres une matrice carrée **M** contenant des nombres, et un entier **j** qui représente l'indice d'une colonne dans **M**. La fonction retourne la somme des éléments de la colonne d'indice **j** dans **M**.

Exemple :

La fonction **somme_colonne (M, 0)** retourne la somme 4+3+13+7 = 27

Q.3- Écrire la fonction **`somme_diag1(M),`** qui reçoit en paramètre une matrice carrée **M** contenant des nombres, et qui retourne la somme des éléments de la première diagonale principale dans **M**.

Exemple :

La fonction **somme_diag1 (M)** retourne la somme 4+2+5+25 = 36

Q.4- Écrire la fonction **`somme_diag2(M),`** qui reçoit en paramètre une matrice carrée **M** contenant des nombres, et qui retourne la somme des éléments de la deuxième diagonale principale dans **M**. (La deuxième diagonale principale part du coin en haut à droite, jusqu'au coin en bas à gauche)

Exemple :

La fonction **somme_diag2 (M)** retourne la somme 3+9+0+7 = 19

II- Carré magique

Q.5- Écrire la fonction **`carre_magique(C),`** qui reçoit en paramètre une matrice carrée **C** contenant des entiers strictement positifs, et qui retourne :

- ✓ **True**, si la matrice **C** est un carré magique : les sommes sur chaque ligne, sur chaque colonne et sur chaque diagonale principale sont toutes égales
- ✓ **False**, sinon.

Exemples :

$A =$

21	*7*	*17*
11	*15*	*19*
13	*23*	*9*

$B =$

7	*1*	*6*
1	*15*	*9*
3	*2*	*4*

- La fonction **carre_magique (A)** retourne **True**
- La fonction **carre_magique (B)** retourne **False**

III- Carré magique normal

Un **carré magique normal** d'ordre **n** est un carré magique d'ordre **n**, constitué de tous les nombres entiers positifs compris entre **1** et $\mathbf{n^2}$.

Exemple :

Carrée magique normal d'ordre **4**, composé des nombres entiers : **1, 2, 3, …, 15, 16**.

4	*14*	*15*	*1*
9	*7*	*6*	*12*
5	*11*	*10*	*8*
16	*2*	*3*	*13*

NB : Il n'existe pas de carré magique normal d'ordre **2**.

Q.6.a- Écrire la fonction `magique_normal(C)`, qui reçoit en paramètre une matrice carrée **C** qui représente un carré magique. La fonction retourne **True** si le carré magique **C** est normal, sinon, elle retourne **False**.

Exemples:

- La fonction **magique_normal ([[8, 1, 6] , [3, 5, 7] , [4, 9, 2]])** retourne **True**
- La fonction **magique_normal ([[21, 7, 17] , [11, 15, 19] , [13, 23, 9]])** retourne **False**

Q.6.b- Déterminer la complexité de la fonction **magique_normal(C).**

IV- Construction d'un carré magique normal d'ordre impair

La méthode ***siamoise*** **est** une méthode qui permet de construire un carré magique normal d'ordre **n** impair.

Le principe de cette méthode est le suivant :

1. Créer une matrice carrée d'ordre **n**, remplie de **0**.

2. Placer le nombre **1** au milieu de la ligne d'indice **0**.

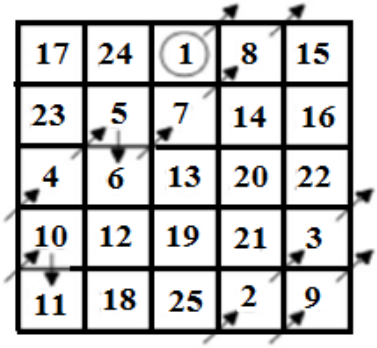
3. Décaler d'une case vers la droite puis d'une case vers le haut pour placer le nombre **2**, et faire de même pour le nombre **3**, puis le nombre **4**, … jusqu'au nombre $\mathbf{n^2}$.
   Le déplacement doit respecter les deux règles suivantes (voir l'exemple dans la page suivante) :

   - Si la pointe de la flèche sort du carré, revenir de l'autre côté, comme si le carré était enroulé sur un tore.

   - Si la prochaine case est occupée par un entier non nul, alors il faut décaler d'une case vers le bas.

Exemple :

Construction d'un carré magique normal d'ordre **5**



Q.7- Écrire la fonction **`matrice_nulle(n)`**, qui reçoit en paramètre un entier **n** strictement positif, et qui retourne une liste qui représente la matrice carrée d'ordre **n**, remplie de **0**.

Exemple :

La fonction **matrice_nulle (5)** retourne la matrice suivante :

```
[[0,0,0,0,0],[0,0,0,0,0],[0,0,0,0,0],[0,0,0,0,0] ,[0,0,0,0,0]]
```

Q.8- Écrire la fonction **`siamoise(n)`**, qui reçoit en paramètre un entier positif **n** impair. En utilisant le principe de la méthode siamoise, la fonction retourne la matrice carrée qui représente le carré magique normal d'ordre **n**.

Exemple :

La fonction **siamoise (7)** retourne la matrice carrée qui représente le carré magique normale d'ordre **7** suivant :

30	*39*	*48*	*1*	*10*	*19*	*28*
38	*47*	*7*	*9*	*18*	*27*	*29*
46	*6*	*8*	*17*	*26*	*35*	*37*
5	*14*	*16*	*25*	*34*	*36*	*45*
13	*15*	*24*	*33*	*42*	*44*	*4*
21	*23*	*32*	*41*	*43*	*3*	*12*
22	*31*	*40*	*49*	*2*	*11*	*20*

Q.9- Écrire la fonction, de complexité constante, **`constante_magique(n)`**, qui reçoit en paramètre un entier positif **n** impair, et qui retourne la valeur de la constante magique du carré magique normal d'ordre **n**.

≈≈≈≈≈≈ **FIN DE L'ÉPREUVE** ≈≈≈≈≈≈