Lycée Charlemagne

.2018.– MPSI2 –.2019.

COMBLE EN COURS

1FO 2

FIFA 1fo 2

Ce qui vient est une adaptation d'un sujet de INT Telecom (une fois entré en école, je sais).

On dispose d'une base de données sur les matchs de football *(pas besoin de savoir jouer, ni de trouver le moindre intérêt à ce jeu pour répondre)*. Les tables dont les noms suivent sont expliquées ainsi dans le sujet :

`Pays(NomPays, Superficie, Continent)` : un pays, identifié par son nom, est décrit par sa superficie et le continent dans lequel il se situe. Cette relation[1] mémorise tous les pays dotés d'une fédération nationale de football.

`EditionCoupeDuMonde(Annee, PaysOrganisateur, PaysVainqueur)` : une édition de coupe du monde est identifiée par l'année de son organisation, et est organisée par un pays. Le pays vainqueur de l'édition est aussi mémorisé.

`ParticipationPays(NomPays, Annee, Entraineur, Parcours)` : ne sont répertoriés dans cette table que les pays ayant participé à une ou plusieurs éditions de coupes du monde. L'attribut entraineur désigne le nom de l'entraineur du pays à une édition donnée. L'attribut parcours mémorise la phase atteinte par le pays, et ne peut donc prendre que l'une des valeurs suivantes ('poule', 'huitième', 'quart', 'demi', 'finales').

`Match(NomPays1, NomPays2, Annee, NbreButsMarque1, NbreButsMarque2, Phase)` : un match désigne la confrontation entre deux pays, le score final de la rencontre est mémorisé par le nombre de buts marqués par chaque équipe. Pour simplifier, les tirs aux buts sont considérés comme des buts marqués en cours du match. L'attribut phase prend les mêmes valeurs que l'attribut parcours. Pour simplifier, on suppose que deux pays ne se confrontent qu'une fois au plus dans chaque édition.

`Joueur (NomJoueur, NomPays, Poste, DateNaiss)` : cette table mémorise les joueurs des différents pays, le poste occupé par chaque joueur dans l'équipe ('gardien de but', 'défenseur', 'attaquant', etc.) ainsi que son année de naissance.

`StatJoueur (NomJoueur, NomPays, Annee, NombreButs)` : cette table indique, pour chaque joueur d'un pays participant, le nombre de buts inscrits à une édition donnée de coupe du monde.

-0- Indiquez pour chaque table si elle a une clef primaire, et si oui, précisez quel est le champ qui joue ce rôle.

-1- Quels sont les pays ayant participé à la coupe du monde 1986 ? (on veut la liste des noms de pays, et le continent, regroupés par continent[a]).

a. attention, ne vous faites pas avoir, `GROUP BY Continent` tasserait toute l'Europe en une seule réponse en indiquant juste le nom du premier pays européen trouvé

-2- Quel est le plus petit pays (par sa superficie) référencé dans la table `Pays` ? (et si ils sont plusieurs à égalité ?)

-3- Quels sont les pays d'Afrique ayant participé à au moins une coupe entre 1990 et 2000 (inclus) ? (liste des noms de pays, si possible sans doublons)

-4- Qui sont les entraineurs ayant amené au moins un fois leur équipe en finale en étant entraineur d'une équipe européenne ? (liste des noms d'entraineurs, avec nom du pays).

1. à Télécom, ils appellent relation ce que l'on appelle aussi table

-5-
```
SELECT J.NomJoueur, J.NomPays, Annee, Annee-DateNaissance AS Truc
FROM Joueur AS J JOIN StatJoueur AS S
WHERE J.NomJoueur=S.NomJoueur AND J.NomPays=S.NomPays
ORDER BY Truc
```
Que fait cette requête ? Pouvez vous la rédiger plus simplement ?

-6- Quels sont les dix joueurs ayant marqué le plus de buts ? (nom du joueur et pays)

-7- Donnez la liste de tous les joueurs africains de moins de vingt ans (c'est pour Bintou).

-8- Donnez la liste des années où la France est allée en quart de finale (au moins).

-9- Pouvez vous donnez la liste des gardiens de but de l'édition 2018 de la coupe, avec pays pour chacun, du plus jeune au plus vieux. J'avais d'abord mis 2017, mais on 'a dit que c'était trop simple...

-10- Donnez la liste (triée par ordre croissant) des années où la coupe a été remportée par le pays organisateur (avec le nom du pays quand même).

-11- Donnez la liste des pays que le France a battus depuis 1970 en quart, demi finale ou finale.

-12- Quels sont les pays n'ayant jamais dépassé la phase des poules ?

-13- Que donne `SELECT Continent, COUNT(*) FROM Pays GROUP BY Continent` ?

Python — 1fo 2

Le phénomène étudié est un signal périodique f (non connu) de période `T` (connue), à valeurs réelles.

-a- Dans tout ce qui suit, on va avoir besoin des fonctions sinus et cosinus. Que devez vous faire pour que `Python` les connaisse ?

On dispose pour l'étude de f d'une série de `N` (connu) mesures/observations équiréparties sur l'intervalle de temps $[0,\ T]$, sous forme d'une liste `FObs` de `N` éléments (`FObs[k]` correspond donc à une mesure (entachée d'erreurs, merci le physicien) de $f(k.T/N)$). Le flottant `dt` sera calculé comme étant `T/N`.

-b- Amis informaticiens pratiques, commentez le résultat de la case ci contre :

```
...a0, a1, dt = 0 , 0, 1e-5
...for k in range(10**5+1) :
.......a0 = k*dt
.......a1 += dt
...print(a0, a1)
>>> 1.0, 1.0000099999980838
```

-c- Écrivez une procédure `fApprox` qui prend en entrée un réel `t` et renvoie la valeur de l'observation `FObs` à l'instant le plus proche de `t` (attention, `t` n'est pas forcément entre `0` et `T`). Cette procédure ne servira pas par la suite.

-d- Écrivez une procédure `Integrale` qui prend en entrée une application `g` et retourne une valeur approchée de $\int_0^T f(x).g(x).dx$ calculée par la méthode des rectangles, dans laquelle on aura assimilé f à `fObs` (combien de rectangles alors ?).

-e- On veut à présent estimer la dérivée de f le long de l'intervalle $[0,\ T]$. Expliquez pourquoi la procédure suivante n'est pas un bon choix :

```
def fPrime(t) :
....return(fApprox(t+dt)-fApprox(t))/dt)
```

-f- On va passer par une autre approche et assimiler f à sa décomposition en série de Fourier : $f_K = \theta \longmapsto \dfrac{a_0}{2} + \sum_{k=1}^{K} a_k.\cos\left(k.\theta.\dfrac{2.\pi}{T}\right) + b_k.\sin\left(k.\theta.\dfrac{2.\pi}{T}\right)$ où les $a_k$et b_k sont appelés coefficients de Fourier de f, en prenant K « assez grand » *(dont la coïncidence avec f ne sera pas prouvée ici).*

-g- Amis matheux, montrez alors : $\dfrac{T}{2}.a_n = \int_0^T f(t).\cos\left(n.t.\dfrac{2.\pi}{T}\right).dt$ pour tout n et pour tout entier n entre 1 et K.

-h- Amis matheux, montrez, quitte à intégrer plusieurs fois par parties : $a_k = O\left(\dfrac{1}{k^3}\right)$.

-i- Écrivez une procédure `Fourier` qui prend en entrée `K` et retourne la liste des `2.K` valeurs approchées $[a_1, b_1, a_2, b_2, \dots a_K, b_K]$ calculées en remplaçant dans les intégrales f par `fObs` (on n'aura pas besoin de a_0, puisqu'on va dériver).

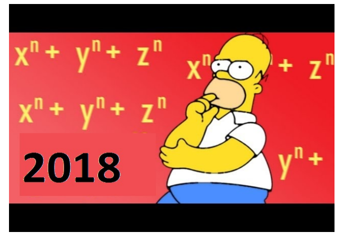
-j- Écrivez une procédure qui prend en entrée K et trace sur un même graphe f, f_K et f'_K sur l'intervalle $[0, T]$.

Lycée Charlemagne

.2018.– MPSI2 –.2019.

CORRECTION

1FO 2



-0- *Indiquez pour chaque table si elle a une clef primaire, et si oui, précisez quel est ce champ.*

Une clef primaire doit garantir l'injectivité.

- `Pays` : NomPays
- `EditionCoupeDuMonde` : Année
- `Match` : pas de clef primaire
- `Joueur` : NomJoueur ne suffit pas, il peut y avoir des homonymes, venant par exemple de pays différents [a].
- `StatJoueur` : le nom ne suffit pas là aussi

[a] on laissera le concepteur se débrouiller si il y a plusieurs Pogba par exemple (et il y en a...)

-1- Quels sont les pays ayant participé à la coupe du monde 1986 ? (on veut la liste des noms de pays, et le continent).

```
SELECT P.NomPays, P.Continent
FROM ParticipationPays AS Part JOIN Pays AS P
     ON Part.NomPays = P.NomPays
WHERE Annee = 1986
ORDER BY P.Continent
```

C'est `ORDER BY` qui va transformer la liste

BRÉSIL,AMÉRIQUE
FRANCE,EUROPE
ALLEMAGNE,EUROPE
ARGENTINE,AMERIQUE
NIGERIA,AFRIQUE

en

NIGERIA,AFRIQUE
BRÉSIL,AMÉRIQUE
ARGENTINE,AMERIQUE
FRANCE,EUROPE
ALLEMAGNE,EUROPE

La jointure avec son ON pourra être convertie en `PartcipationPays NATURAL JOIN Pays`.

-2- *Quel est le plus petit pays (par sa superficie) référencé dans la table Pays ?*

```
SELECT NomPays
FROM Pays
ORDER BY Superficie ASC LIMIT 1
```

Ou pour les plus tordus, mais si il y a des pays à égalité

```
SELECT NomPays
FROM Pays
WHERE Superficie = SELECT Min(Superficie)
                   FROM Pays
```

-3- *Quels sont les pays d'Afrique ayant participé à au moins une coupe entre 1990 et 2000 (inclus) ?*

```
SELECT P.NomPays
FROM ParticipationPays AS Part JOIN Pays AS P
     ON
WHERE Part.Annee BETWEEN 1990 AND 2000
      AND P.Continent = 'Afrique'
```

On a besoin d'une jointure pour sélectionner le contient

Mais on peut aussi utiliser un `IN`

```
SELECT NomPays
FROM ParticipationPays
WHERE Annee BETWEEN 1990 AND 2000
      AND NomPays IN SELECT NomPays
                     FROM Pays
                     WHERE Continent = 'Afrique'
```

Et pour éviter les doublons `GROUP By NomPays`

-4- *Qui sont les entraineurs ayant amené au moins un fois leur équipe en finale en étant entraineur d'une équipe européenne ?*

```
SELECT Part.Entraineur, Part.NomPays
FROM Participation AS Part JOIN JOIN Pays AS P
     ON P.NomPays = Part.NomPays
WHERE P.Continent='Europe'
      AND Part.Parcours='Finale'
```

Quels sont les pays n'ayant jamais dépassé la phase des poules ? (liste des pays, regroupés si possible par continents)

SELECT

-5- *Que fait cette requête ?*

```
SELECT J.NomJoueur, J.NomPays, Annee, Annee-DateNaissance
AS Truc
FROM Joueur AS J JOIN StatJoueur AS S
WHERE J.NomJoueur=S.NomJoueur AND J.NomPays=S.NomPays
ORDER BY Truc
```

La condition `WHERE J.NomJoueur=S.NomJoueur AND J.NomPays=S.NomPays` peut être remplacée par un `NATURAL JOIN`, puisque SQL verra spontanément les champs `NomJoueur` et `NomPays` à apparier entre les deux tables.

On édite la liste des joeurs, avec leur pays, leur année de naissance et l'âge qu'ils avaient lors de la coupe à laquelle ils ont participé. On trie ensuite sur cet âge.

-6- *Quels sont les dix joueurs ayant marqué le plus de buts ? (nom du joueur et pays)*

Attention, un joueur peut avoir participé à plusieurs coupes. Il faudra sommer :

```
SELECT NomJoueur, NomPays, SUM(NombreButs)AS TotalButs
FROM StatJoueurs
GROUP BY NomJoueur
ODER BY TotalButs DESC LIMIT 10
```

On doit fusionner par un GROUP BY NomJoueur pour avoir tous les buts d'un même joueur (on somme en fait sur les diverses années). On crée le champ TotalButs en lui donnant un nom, on trie sur ce champ et on ne garde que les dix premiers.

-7- *Donnez la liste de tous les joueurs africains de moins de vingt cinq ans.*

On va sortir la liste des pays d'Afrique par

```
SELECT NomPays
FROM Pays
WHERE Continent='Afrique'
```

.

```
SELECT NomJoueur
FROM Joueur
WHERE NomPays IN SELECT NomPays
                 FROM Pays
                 WHERE Continent = 'Afrique'
      AND DateNaiss > 1996
```

Ou si vous préférez les jointures

```
SELECT J.NomJoueur
FROM Joueur AS J JOIN Pays AS P
     ON J.NomPays = P.NomPays
WHERE P.Continent = 'Afrique'
      AND DateNaiss > 1996
```

-8- *Donnez la liste des années où la France est allée en quart de finale (au moins).*

```
SELECT Annee
FROM ParticipationPays
WHERE NomPays = 'France'
      AND Parcours IN ('Quart', 'Demi, 'Finale')
ORDER BY Annee
```

(je fais un peu de zèle)

Je doute que SQL puisse mettre spontanément une relation d'ordre sur le champ `Parcours`.

-9- *Pouvez vous donner la liste des gardiens de but de l'édition 2018 de la coupe, avec pays pour chacun, du plus jeune au plus vieux.*

Non. Aucune jointure ne permet de savoir pour un joueur en quelle année il participait à la coupe. Sauf si on le fait entrer dans la table `StatJoueurs` avec une statistique nulle sur `NombreButs`.
On m'objectera qu'un gardien peut marquer un but si il tire loin, ou même qu'on peut indiquer une statistique négative pour les buts encaissés.
Alors dans ce cas, on y va.

```
SELECT J.NomJoueur, J.NomPays
FROM Joueur AS J JOIN StatJoueur AS S
     ON J.NomJoueur=S.NomJoueur AND J.NomPays=S.NomPays
     #commentaire : ou même juste FROM Joueur AS J NATURAL JOIN StatJoueur AS S
WHERE S.Annee = 2018
      AND J.Poste LIKE '%Gardien%'
ORDER BY J.DateNaiss DESC
```

Vais je donner des points à Eros et d'autres si au lieu d'une requête SQL, ils répondent Keylor Navas, Patrick Pemberton, Leonel Moreira... *(une liste explicite de 32 noms).*

-10- *Donnez la liste triée par ordre croissant des années où la coupe a été remportée par le pays organisateur.*

```
SELECT Annee
FROM EditionCoupeDuMonde
WHERE PaysOrganisateur = PaysVainqueur
ORDER BY Annee
```

-11- *Donnez la liste des pays que le France a battus depuis 1970 en quart, demi finale ou finale.*

```
SELECT NomPays2
FROM Match
WHERE Annee >= 1970
      AND Phase IN ('Finale', 'Demi', 'Quart')
      AND NomPays1 = 'France'
      AND NbrsButsMarques1>NbrButsMarques2
UNION SELECT NomPays2
      FROM Match
      WHERE Annee >= 1970
            AND Phase IN ('Finale', 'Demi', 'Quart')
            AND NomPays2 = 'France'
            AND NbrsButsMarques1<NbrButsMarques2
```

Oui, la `France` peut être `Pays1` ou `Pays2`.

Si on veut un tri :

```
SELECT NomPays2 AS Nom
FROM Match
WHERE Annee >= 1970
      AND Phase IN ('Finale', 'Demi', 'Quart')
      AND NomPays1 = 'France'
      AND NbrsButsMarques1>NbrButsMarques2
UNION SELECT NomPays2 AS Nom
      FROM Match
      WHERE Annee >= 1970
            AND Phase IN ('Finale', 'Demi', 'Quart')
            AND NomPays2 = 'France'
            AND NbrsButsMarques1<NbrButsMarques2
ORDER BY Nom DISTINCT
```

Je ne sais plus trop ce que ces deux réponses font ici. Peut être des réponses à des questions supprimées depuis.

```
SELECT PaysVainqueur
FROM EditionCoupeDuMonde
WHERE Annee > = 1986
```

```
SELECT Ed.PaysVainqueur, P.Continent
FROM EditionCoupeDuMonde AS Ed JOIN Pays
AS P
ON PaysVainqueur=NomPays
```

Et aussi `WHERE Ed.Annee >= 1986`

-13- Que donne `SELECT Continent, COUNT(*) FROM Pays GROUP BY Continent` ?

Réponse : une mavuvaise note car la requète est mal présentée.
Mais sinon, on aura la liste EUROPE, 15
ASIE, 10
AMÉRIQUE, 13
AFRIQUE, 18
OCÉANIE, 5
avec, pour chaque continent le nombre d'équipes répertoriéesj'ai mis ici un nombre au hasard
C'est le `GROUP BY` avec le `COUNT(*)` qui font ce travail.
L'ordre d'édition des continents ne sera pas imposé, et dépendra du traitement fait par SQL.

Je laisse de côté l'hitoire de `not(Parcours= 'Poule')` et je vous offre

```
SELECT P.Annee, P.Entraineur
FROM ParticipationPays AS P JOIN StatsJoueurs AS S
     ON P.NomPays=S.NomPayP.Annee=S.Annee
WHERE P.Annee > 1950
GROUP BY NomJoueur, NomPays, P.Annee, P.Entraineur
HAVING SUM(NombreButs) >= ALL (SELECT SUM(NombreButs)
                              FROM StatJOUEUR
                              GROUP BY NomJoueur
                              WHERE Annee > 1950)
```

C'est la réponse à
pour chaque édition ultérieure à 1950, quels sont les entraineurs des joueurs ayant marqué le plus grand nombre de buts.
Vivement que vous soyez à Télécom Sud Paris (INT Telecom) pour savoir faire ça tout seuls !

Python 2

Pour pouvoir utiliser sinus, cosinus et autres, on les importe de puis le module `math`
(certains opteront pour les modules spécifiques de `numpy`, `scipy` et autres).
On peut choisir `from math import *` ou `from math import sin, cos`
Reste `import math as m` et ensuite on sollicite `m.sin` et `m.cos`.

C'est une question du concours Mines-Ponts 2019.
La formule a0 = k*dt calcule les k.dt les uns après les autres. A chaque fois, on en recrée un nouveau. Le dernier, pour k égal à 10^5 (le range(10**5+1) nous amène bien sur lui) donne le flottant $10^5.(10^{-5})$.
Il vaut 1.0 (et pas 1, on a un flottant).

En revanche, avec l'itération des a1+= dt, on calcule $\sum_{i=0}^{k} dt$ ce qui fait mathématiquement $k.dt$, mais en réalité $dt + dt + \ldots + dt$. Comme chaque flottant 10^{-5} est une valeur approchée, les 10^5 additions finissent par créer une incertitude : `.0000099999980838`.
De surcroit, on a additionné trop de termes, on déborde, on en est à $(10^5 + 1).10^{-5}$.

*En plus, on fera observer qu'avec N=10**5+5, dt=1/N, le nombre N*dt ne vaut plus 1.0 mais 0.9999999999999999.*
Je me demande ce que les correcteurs attendaient à part : Python ne fait pas des calculs de matheux, mais des calculs approchés.

On veut arrondir `f(t)` à un certain `f(k.T/N)` d'abscisse proche.
Il suffit d'avoir `t` peu différent de `k.T/N`, mais avec `k` entier. On demande à `k` d'être proche de $\frac{t.N}{T}$, ou encore $\frac{t}{dt}$.
On va arrondir grâce à la partie entière :

```
def fApprox(t) :
....k = int(t/dt)
....return(FObs[k])
```

avec des crochets, s'il vous plait, `PObs` est une liste.
Mais il reste le problème de l'intervalle de travail, en profitant de la périodicité.

```
def fApprox(t) :
....k = int(t/dt)
....return(FObs[k%N])
```

Pour intégrer, on va profiter des endroits où on connaît f en tout cas en valeur approchée : les $k.T/N$.
La formule sera donc $\frac{T}{N}.\sum_{k=0}^{N} f\Big(k.\frac{T}{N}\Big).g\Big(k.\frac{T}{N}\Big).dt$.

```
def Integrale(f) :
....S = 0
....for k in range(N) :
........S += fObs[k]*g(k*dt)
....return(S*dt)
```

La formule

```
def fPrime(t) :
....return(fApprox(t+dt)-fApprox(t))/dt)
```

revient à écrire $f'(x) = \frac{f(x+h)-f(x)}{x}$ avec h égal à $\frac{T}{N}$ (il ne tend pas vers 0, mais ce n'est pas grave, tendre vers 0 n'a pas de sens dans l'univers fini de l'ordinateur).
Le problème est que $f(x+h)$ et $f(x)$ ne sont définis qu'à l'imprécision de mesure près. La différence est donc à ε près. Et on divise par un réel « très petit » . Le résultat fait que l'erreur est donc en $N.\varepsilon/T$. Plus N est grand, plus cette imprécision est énorme.

Sinon, y a-t-il un risque qu'en prenant `fApprox` on ait `fApprox(t+dt)-fApprox(t)` car les deux renverraient au même `fObs[k]` ? Non, car si on a posé `fApprox(t+dt)=fObs[k]` alors on a $k = \left[\frac{x}{T}.N\right]$ et $k = \left[\frac{x+dt}{T}.N\right] = \left[\frac{x+T/N}{T}.N\right] = \left[\frac{x}{T}.N+1\right] = k+1$. On va donc chercher à coup sûr l'élément suivant de la liste des observations.

On pose donc $f \simeq f_K = x \longmapsto \frac{a_0}{2} + \sum_{k=1}^{K} a_k.\cos\Big(k.x.\frac{2.\pi}{T}\Big) + b_k.\sin\Big(k.x.\frac{2.\pi}{T}\Big)$. partons alors du membre le plus compliqué : $\int_0^T f(t).\cos\Big(n.t.\frac{2.\pi}{T}\Big).dt$ avec l'espoir de trouver a_n (n est un entier naturel donné).
On remplace f par f_K et donc par la somme :
$$\int_0^T f(t).\cos\Big(n.t.\frac{2.\pi}{T}\Big).dt = \int_0^T \Big(\frac{a_0}{2} + \sum_{k=1}^{K} a_k.\cos\Big(k.x.\frac{2.\pi}{T}\Big) + b_k.\sin\Big(k.x.\frac{2.\pi}{T}\Big)\Big).\cos\Big(n.t.\frac{2.\pi}{T}\Big).dt$$
On sépare par linéarité :
$$\int_0^T f(t).\cos\Big(n.\theta.\frac{2.\pi}{T}\Big).dt = \int_0^T \frac{a_0}{2}.\cos\Big(n.\theta.\frac{2.\pi}{T}\Big).dt + \sum_{k=1}^{K} a_k.\int_0^T \cos\Big(k.x.\frac{2.\pi}{T}\Big).\cos\Big(n.\theta.\frac{2.\pi}{T}\Big).dt + b_k.\int_0^T \sin\Big(k.x.\frac{2.\pi}{T}\Big).\cos\Big(n.\theta.\frac{2.\pi}{T}\Big).dt$$
On doit calculer des intégrales comme $\int_0^T \frac{a_0}{2}.\cos\Big(n.\theta.\frac{2.\pi}{T}\Big).dt$, $\int_0^T \cos\Big(k.x.\frac{2.\pi}{T}\Big).\cos\Big(n.\theta.\frac{2.\pi}{T}\Big).dt$ et $\int_0^T \cos\Big(k.x.\frac{2.\pi}{T}\Big).\sin\Big(n.\theta.\frac{2.\pi}{T}\Big).dt$.
La première s'intègre en sinus et donne 0 ($\sin\Big(n.T.\frac{2.\pi}{T}\Big) = 0$).

Pour le second modèle, on linéarisé ou intègre par parties.
Je vais ici linéariser : $\int_0^T \cos\Big(k.x.\frac{2.\pi}{T}\Big).\cos\Big(n.\theta.\frac{2.\pi}{T}\Big).dt = \frac{1}{2}.\int_0^T \cos\Big((k+n).x.\frac{2.\pi}{T}\Big).dt + \frac{1}{2}.\int_0^T \cos\Big((k-n).x.\frac{2.\pi}{T}\Big).dx$.
Les deux termes s'intègrent en sin. En 0 on a 0 et en T on a $\frac{1}{n+k}.\sin\Big((k+n).T.\frac{2.\pi}{T}\Big) = 0$ et $\frac{1}{n-k}.\sin\Big((k-n).T.\frac{2.\pi}{T}\Big) = 0$.
Bref, $\int_0^T \cos\Big(k.x.\frac{2.\pi}{T}\Big).\cos\Big(n.\theta.\frac{2.\pi}{T}\Big).dt = 0$.

Pour le modèle $\int_0^T \cos\Big(k.x.\frac{2.\pi}{T}\Big).\sin\Big(n.\theta.\frac{2.\pi}{T}\Big).dt$, je varie. pas de linéarisation, mais un recours aux

parties :
$\displaystyle\int_0^T \cos\left(k.x.\frac{2.\pi}{T}\right).\sin\left(n.\theta.\frac{2.\pi}{T}\right).dt =$
$\displaystyle\left[\cos\left(k.x.\frac{2.\pi}{T}\right).\frac{-\cos\left(n.\theta.\frac{2.\pi}{T}\right)}{n}\right]_0^T - \int_0^T (-k.\frac{2.\pi}{T}).\sin\left(k.x.\frac{2.\pi}{T}\right).\frac{-\cos\left(n.\theta.\frac{2.\pi}{T}\right)}{n}.dt$
le terme crochet est nul par périodicité des cosinus en jeu. On refait une intégration par parties sur l'intégrale, et on obtient au final
$\displaystyle\int_0^T \cos\left(k.x.\frac{2.\pi}{T}\right).\sin\left(n.\theta.\frac{2.\pi}{T}\right).dt = A.\int_0^T \cos\left(k.x.\frac{2.\pi}{T}\right).\sin\left(n.\theta.\frac{2.\pi}{T}\right).dt$ avec a égal à $\displaystyle-\frac{k^2}{n^2}.\frac{4.\pi^2}{T^2}$.
La seule solution est que l'intégrale soit nulle.

Mais alors, aurait on $\displaystyle\int_0^T \frac{a_0}{2}.\cos\left(n.\theta.\frac{2.\pi}{T}\right).dt = 0$, $\displaystyle\int_0^T \cos\left(k.x.\frac{2.\pi}{T}\right).\cos\left(n.\theta.\frac{2.\pi}{T}\right).dt = 0$ et $\displaystyle\int_0^T \cos\left(k.x.\frac{2.\pi}{T}\right).\sin\left(n.\theta.\frac{2.\pi}{T}\right).dt = 0$?
Si tel est le cas, on aboutit à $\displaystyle\int_0^T f(t).\cos\left(n.\theta.\frac{2.\pi}{T}\right).dt = 0$.
Ce n'est pas ce qu'on souhaite.
Mais il y a une étape où l'on est allé trop vite. C'est quand on a écrit $\displaystyle\frac{1}{n-k}.\sin\left((k-n).T.\frac{2.\pi}{T}\right) = 0$.
C'est valable quand n est différent de k. Mais sinon, pour n égal à k, $\displaystyle\frac{1}{2}.\int_0^T \cos\left((k-n).x.\frac{2.\pi}{T}\right).dx$ vaut $\displaystyle\frac{1}{2}.\int_0^T 1.dx$, ce qui , fait $\displaystyle\frac{T}{2}$.
On reprend donc : toutes les intégrales de la liste sont nulles, sauf celle pour k égal à n, qui vaut $\displaystyle\frac{T}{2}$.
De $\displaystyle\int_0^T f(t).\cos\left(n.\theta.\frac{2.\pi}{T}\right).dt = \int_0^T \frac{a_0}{2}.\cos\left(n.\theta.\frac{2.\pi}{T}\right).dt + \sum_{k=1}^{K} a_k.\int_0^T \cos\left(k.x.\frac{2.\pi}{T}\right).\cos\left(n.\theta.\frac{2.\pi}{T}\right).dt +$ $\displaystyle b_k.\int_0^T \sin\left(k.x.\frac{2.\pi}{T}\right).\cos\left(n.\theta.\frac{2.\pi}{T}\right).dt$ il ne reste que $\displaystyle a_n.\frac{T}{2}$ comme demandé.

Cette démonstration, c'est une partie du cours sur les séries de Fourier.
Et ce cours, c'est Sup ou Spé ?
Depuis la dernière réforme, ce n'est ni l'un ni l'autre, elles ont disparu de votre formation.
Sauf en physique et en S.I.I. où on est amené à les utiliser. On décompose un signal en somme de signaux sinusoïdaux.

Notons qu'avec le même type de calculs, on arrive à $\displaystyle b_n = \frac{2}{T}.\int_0^T f(t).\cos\left(n.t.\frac{2.\pi}{T}\right).dt$.
On récupère les coefficients de Fourier en testant la fonction f contre des sinus et des cosinus.

Pour récupérer la liste des coefficients de $[a_1, b_1, a_2, b_2, \dots a_K, b_K]$, il suffit donc de calculer des intégrales.
Là où on va tester votre capacité à faire de l'informatique, c'est en voyant si vous pensez à utiliser ce que vous avez crée avant.
En effet, vous avez crée la procédure `Integrale`. A vous de l'utiliser.

```
def Fourier(K) :
....Coeffs = [ ]
....for k in range(1, K+1) :
........cosk = lambda x : cos(x*k*2*pi/T)
........Coeffs.append(Integre(cosk)*2/T)
........sink = lambda x : sin(x*k*2*pi/T)
........Coeffs.append(Integre(sink)*2/T)
....return(Coeffs)
```

La syntaxe `g = lambda x : cos(2*x)` permet de définir que g est l'application $x \longmapsto \cos(2.x)$. Vous l'avez croisée avec Solène.
ici, on l'applique à $cosk = x \longmapsto \cos(k.x.2.\pi/T)$ *(c'est un nom propre choisi par mes soins)* puis $sink = x \longmapsto \sin(k.x.2.\pi/T)$ pour pouvoir les passer dans la fonction à intégrer.

On veut tracer f, f_K et f'_K.
On va créer la liste des coefficients avec la procédure ci dessus `CoeffFourier=Fourier(k)`. Ensuite,

on va l'utiliser pour reconstruire f_K :

```
def fk(x, K) :
....S = 0
....for k in range(1, K+1) :
........S += CoeffFourier[2*k]*cos(x*2*k*pi/T)
........S += CoeffFourier[2*k+1]*sin(x*2*k*pi/T)
....return(S)
```

Je sais, il manque a_0. Il faut donc le calculer avec un `Integrale(lambda x : 1)/2` et le mettre dans `S` avant de commencer les calculs.

Mais aussi la série des dérivées : $f'_K = x \longmapsto \sum_{k=1}^{K} -\frac{a_k.k.2.\pi}{T}.\sin\Big(k.x.\frac{2.\pi}{T}\Big) + \frac{b_k.k.2.\pi}{T}.\cos\Big(k.x.\frac{2.\pi}{T}\Big)$.

```
def dfk(x, K) :
....S = 0
....for k in range(1, K+1) :
........S -= CoeffFourier[2*k]*k*sin(x*2*k*pi/T)
........S += CoeffFourier[2*k+1]*k*cos(x*2*k*pi/T)
....return(S*2*pi/T)
```

Notez que cette fois on a par exemple `S -= CoeffFourier[2*k]*k*sin(x*2*k*pi/T)` avec un signe moins et un facteur k. Ensuite, on doit même tout multiplier par $2.\pi/T$. Autant ne le faire qu'à la fin.
La représentation graphique n'est qu'un appel à `pyplot`

```
import numpy as np #du classique
import matplotlib.pyplot as plt #là aussi
x = list(range(N)) #une liste de N abscisses régulières
yf = [fObs[k] for k in range(N)] #les ordonnées de la liste des mesures
plt.plot(x, yf) #on crée un graphe
yfk = [fk(k*dt) for k in range(N)] #les ordonnées calculées avec fk()
plt.plot(x, yfk) #un second graphe
yprimefk = [fk(k*dt) for k in range(N)]
plt.plot(x, yprimefk) #un troisième graphe
plt.show() #on demande de tout afficher dans une même fenêtre
```

Ah oui, j'ai laissé de côté la question où le matheux doit montrer que les a_n sont des $o(n^{-3})$, en intégrant par parties dit on.
On a donc $a_n = \frac{2}{T}.\int_0^T f(t).\cos\Big(n.t.\frac{2.\pi}{T}\Big).dt$.
On pourrait déjà majorer (inégalité triangulaire, majoration du cosinus par 1) :
$|a_n| \leqslant \frac{2}{T}.\Big|\int_0^T f(t).\cos\Big(n.t.\frac{2.\pi}{T}\Big).dt\Big| \leqslant \frac{2}{T}.\int_0^T \Big|f(t)\Big|.\Big|\cos\Big(n.t.\frac{2.\pi}{T}\Big)\Big|.dt \leqslant \frac{2}{T}.\int_0^T \Big|f(t)\Big|.dt$
En notant M_0 un majorant de $|f|$ sur $[0,\ T]$ (existence par théorème de compacité), on a donc $|a_n| \leqslant \frac{M_1.2}{T}$.

Mais si on commence par intégrer par parties :

$f(t)$	$\hookrightarrow$	$f'(t)$
$\cos\Big(n.t.\frac{2.\pi}{T}\Big)$	$\hookleftarrow$	$\frac{T}{2.n.\pi}.\sin\Big(n.t.\frac{2.\pi}{T}\Big)$

on trouve $a_n = -\frac{1}{n.\pi}.\int_0^T f'(t).\sin\Big(n.t.\frac{2.\pi}{T}\Big).dt$.
En effet, le terme crochet est nul, les sinus étant nuls en 0 et en T.
Le même type de majoration donne cette fois $|a_n| \leqslant \frac{1}{n.\pi}.\int_0^T |f'(t)|.|\sin(\ldots)|.dt \leqslant \frac{M_1}{n.\pi}$ où M_1 est un majorant de $|f'|$ sur $[0,\ T]$. c'est mieux : $a_n = O\Big(\frac{1}{n}\Big)$.

Mais si on recommence

$f'(t)$	$\hookrightarrow$	$f"(t)$
$\frac{T}{2.n.\pi}.\sin\Big(n.t.\frac{2.\pi}{T}\Big)$	$\hookleftarrow$	$-\Big(\frac{T}{2.n.\pi}\Big).\cos\Big(n.t.\frac{2.\pi}{T}\Big)$

, on a cette fois a_n de la forme $\int_0^T f"(t).\cos\Big(n.t.\frac{2.\pi}{T}\Big).dt$ avec cette fois devant un facteur en $\frac{1}{n^2}$.
Le terme crochet a sauté non pas à cause des sinus. Cette fois, c'étaient des cosinus, valant 1. Mais f' est T périodique.
En notant M_2 un majorant de $f"$, on a $|a_n| \leqslant \frac{M_2}{n^2}$.

On recommence une intégration par parties. A chaque fois, le terme crochet est nul par périodicité de $f^{(k)}$ quand ce n'est pas grâce aux sinus. Et on a une intégrale avec un $\frac{1}{n^k}$devant et juste $f^{(k)}$ face à des sinus ou des cosinus. C'est ce que l'on souhaitait.
Plus une application est dérivable, plus ses coefficients de Fourier tendent vite vers 0.