MPSI 1 **Samedi 9 Janvier 2021**

Devoir surveillé 4 : maths-informatiques (4h)

Les calculatrices et autres appareils électroniques (téléphones, montres connectées etc.) sont interdits.

Soignez la précision de la rédaction, faites des phrases liées logiquement.
Encadrez ou soulignez vos résultats, séparez clairement vos questions, la clarté de votre présentation est un élément important d'appréciation.
L'essentiel du problème porte sur les partie 1 et 2. Les parties 0 et 3 sont indépendantes du reste.
Les questions comptant dans le barème d'info seront notées `Info`. **Bon courage !**

0 La méthode d'Euler promise :

0.1 Le banquier réel

Un banquier vous crédite vos intérêts de manière continue, en temps réel, c'est moderne : si votre taux d'intérêt est un nombre $r \in \mathbb{R}^{+*}$, et votre capital au temps 0 est $c(0) = \lambda$, la fonction $t \mapsto c(t)$ donnant l'évolution de votre capital vérifie alors l'équation différentielle :

$$(E) \ : \ \forall \ t \geq 0, \ c'(t) = r.c(t).$$

Ensuite, parce qu'il a oublié son bon cours sur les E.D. de l'époque où il était en prépa ECS, il confie la résolution de cette E.D. à un ordinateur où est seulement implémentée la méthode d'Euler, avec un pas $1/n$.

Disons que la variable t représente le temps en année, donc $t = 0.1$ représente un dixième d'année. S'il choisit $n = 10$, il exécutera donc la méthode d'Euler avec 10 pas de 1/10 pour savoir à combien sera votre capital au bout d'un an.

a) `Info` Pour n fixé, on note c_n la solution approchée de (E) obtenue par la méthode d'Euler avec un pas de $1/n$ et la C.I. $c_n(0) = \lambda$. On note alors $t_k = k/n$. Donner, en la justifiant, une formule explicite pour $c_n(k/n)$ pour chaque $k \in [\![0, n \,]\!]$.

b) Il se dit ensuite qu'en faisant tendre n vers l'infini, $c_n(1)$ devrait tendre vers $c(1)$ où c est la *vraie* solution du problème de Cauchy donné par (E) et $c(0) = \lambda$. Démontrez cette intuition dans cet exemple particulier (on dit qu'ici la méthode d'Euler converge).

0.2 Le banquier imaginaire

A force de chercher ses cours de prépa, le banquier est tombé sur le chapitre sur les nombres complexes. Il se prend à rêver des comptes où l'argent sera représenté par un nombre complexe $z = x + iy$ avec x en euro et y en bitcoin.

La fonction $t \mapsto c(t)$ est maintenant de $\mathbb{R}^+$ dans $\mathbb{C}$ avec $c(0) = \lambda \in \mathbb{C}$ et il vaut propose un taux d'intérêt $z \in \mathbb{C}$ définissant l'évolution de votre capital suivant la même E.D. :

$$\forall \, t \geq 0, \ c'(t) = z.c(t).$$

a) Déterminer la solution exacte de ce problème de Cauchy.

b) On note pour tout $n \in \mathbb{N}$, $z_n = (1 + \frac{z}{n})^n$ et $z = x + iy$ avec $(x, y) \in \mathbb{R}^2$.

i) Montrer que $|z_n| \underset{n \to +\infty}{\longrightarrow} \exp(x)$.

ii) On pose $1 + \frac{z}{n} = |1 + \frac{z}{n}|e^{i\theta_n}$ où $\theta_n \in]-\pi, \pi]$. Montrer que $\theta_n \underset{n \to +\infty}{\longrightarrow} 0$.

iii) Montrer que $\tan(\theta_n) = \frac{y}{n} + o(\frac{1}{n})$ quand $n \to +\infty$.

c) Déduire de ce qui précède qu'ici encore la méthode d'Euler converge.

N.B. La méthode présentée ici, en séparant x et y n'est pas très jolie... une plus conceptuelle sera possible en deuxième année

1 Être ou ne pas être un carré modulo p : symbole de Legendre

Motivation et terminologie – On note dans ce problème $\mathbb{P}$ l'ensemble des nombres premiers. Le but de cette partie 1 est d'étudier, pour tout nombre premier $p \in \mathbb{P}$ et tout entier $a \in \mathbb{Z}$ à quelle condition l'équation $x^2 \equiv a\,[p]$ a-t-elle une solution, autrement dit, de savoir quand $\bar{a} \in \mathbb{Z}/p\mathbb{Z}$ peut s'écrire $\bar{x}^2$ avec $\bar{x} \in \mathbb{Z}/p\mathbb{Z}$. Dans ce cas, on dira simplement que : *a est un carré modulo p.*

1.1. Une première caractérisation des carrés

a) **Un détour par les groupes et les corps**

i) Soit $(G,\cdot)$ un groupe commutatif de neutre noté e.
Soit $k \in \mathbb{N}$ fixé et $H = \{a \in G,\ a^k = e\}$. Montrer que H est un sous-groupe de $(G,\cdot)$.

ii) Soit K un corps (commutatif) quelconque où $2 \neq 0$. Justifier que les fonctions polynomiales du second degré $f : x \mapsto ax^2 + bx + c$ admettent au plus deux racines dans K.

iii) **Résultat admis utile pour la suite :** on *admet* ici (cf. chapitre K) que dans un corps (commutatif) K *quelconque* une équation de degré d, i.e. de la forme $a_d x^d + \cdots + a_1 x + a_0 = 0$, avec $a_0, \ldots, a_n$ dans K et $a_d \neq 0$ admet au plus d solutions $x \in K$.

b) **Cas de $G = \mathbb{Z}/p\mathbb{Z}^*$: Soit $p \in \mathbb{P}$, $p \neq 2$**
On note $G = \mathbb{Z}/p\mathbb{Z}^*$ i.e. $G = \mathbb{Z}/p\mathbb{Z} \setminus \{\bar{0}\}$. On sait que $(G,\cdot)$ est un groupe ayant $p-1$ éléments.
Soit $C = \{a \in \mathbb{Z}/p\mathbb{Z}^*,\ \exists\, x \in \mathbb{Z}/p\mathbb{Z}^*,\ a = x^2\}$.
Autrement dit C est l'ensemble des carrés dans $\mathbb{Z}/p\mathbb{Z}^*$.
On note $H = \{a \in \mathbb{Z}/p\mathbb{Z}^*, a^{(p-1)/2} = \bar{1}\}$.

i) Montrer que $C \subset H$.

ii) Montrer aussi que pour tout $a \in \mathbb{Z}/p\mathbb{Z}^*$, $a^{(p-1)/2} = \bar{1}$ ou $a^{(p-1)/2} = -\bar{1}$.

iii) Démontrer que C admet exactement $(p-1)/2$ éléments, autrement dit qu'il y a exactement $(p-1)/2$ carrés différents dans $\mathbb{Z}/p\mathbb{Z}^*$.

iv) A l'aide d'un résultat précédemment cité, justifié que H ne peut pas avoir plus de $(p-1)/2$ éléments et conclure que $C = H$.

> **Conclusion de cette partie :** on a démontré que a est un carré dans $\mathbb{Z}/p\mathbb{Z}^*$ ssi $a^{(p-1)/2} = \bar{1}$ et que si a n'est pas un carré alors $a^{(p-1)/2} = -\bar{1}$.

v) A l'aide de la conclusion, déterminer la CNS sur p (modulo 4) pour que -1 soit un carré modulo p.

1.2. Introduction du symbole de Legendre

Définition (*symbole de Legendre*) : soit $p \in \mathbb{P}$ et $a \in \mathbb{Z}$. On va noter :

$$\begin{cases} L(a,p) = 0 & \text{si } a \equiv 0\,[p], \\ L(a,p) = 1 & \text{si } a \not\equiv 0\,[p], \text{ et } a \text{ est un carré mod p.} \\ L(a,p) = -1 & \text{si } a \not\equiv 0\,[p], \text{et } a \text{ n'est pas un carré mod p.} \end{cases}.$$

a) Démontrer **la formule d'Euler** suivante :

$$\forall\, a \in \mathbb{Z}, \forall\, p \in \mathbb{P} \setminus \{2\},\ L(a,p) \equiv a^{(p-1)/2}\,[p].$$

b) En déduire que pour tout $p \in \mathbb{P}$ et tout $(a,b) \in \mathbb{Z}^2$: $L(ab,p) = L(a,p)L(b,p)$.

c) `Info` Montrer qu'avec la formule d'Euler du a), on peut calculer $L(a,p)$ en un $O(\log(p))$ opérations : préciser l'algorithme à utiliser.

1.3. Propriétés clef du calcul du symbole de Legendre

On *admet* les deux propriétés cruciales (et plus difficiles) suivantes :
• pour tout $p \in \mathbb{P} \setminus \{2\}$, $L(2,p) = (-1)^{(p^2-1)/8}$, autrement dit 2 est un carré modulo p ssi $p \equiv 1$ ou $p \equiv -1$ [8].
• pour tout $(p,q) \in (\mathbb{P} \setminus \{2\})^2$, $L(q,p) = (-1)^{\frac{(p-1)}{2}\cdot\frac{(q-1)}{2}} L(p,q)$ (réciprocité de Gauss).
Autrement dit comme des nombres premiers impairs sont congrus à 1 ou 3 modulo 4, on a $L(q,p) = -L(p,q)$ ssi $p \equiv q \equiv 3$ [4] et $L(q,p) = +L(p,q)$ sinon.

La loi de réciprocité permet un calcul rapide du symbole de Legendre comme nous allons le montrer sur un exemple. Ne pas oublier que par déf. $L(a,p)$ ne dépend que de la classe de a modulo p, donc si $a \equiv a'$ [p] alors $L(a,p) = L(a',p)$. Donc on peut à chaque étape remplacer a par $a\%p$, le reste de sa division euclidienne par p.

a) Justifier chaque étape du calcul suivant en précisant le résultat utilisé : *on veut savoir si* 11 *est un carré dans* $\mathbb{Z}/83\mathbb{Z}$.
$L(11,83) = -L(83,11) = -L(6,11) = -L(2,11)L(3,11) = L(3,11) = -L(11,3) = -L(2,3) = 1$.
Donc 11 est un carré mod. 83

b) Déterminer de même si 95 est un carré modulo 191 (sachant que 191 est premier).

Remarque : *Le calcul de l'exemple précédent suggère un algorithme pour calculer les symboles de Legendre plus rapide que celui du 1.2.c).* [1]

Il s'avère plus commode d'écrire un algorithme passant par le calcul d'un objet un peu plus général, appelé symbole de Jacobi, que nous introduisons maintenant.

1.4. Généralisation : le symbole de Jacobi et ses propriétés immédiates

Au 1.3. précédent, on a défini le symbole de Legendre $L(a,p)$ pour $a \in \mathbb{Z}$ quelconque et $p \in \mathbb{P}$. Pour aboutir à un algorithme de calcul efficace de ce symbole, il est commode de commencer par étendre sa définition en remplaçant le nombre premier p par un entier b *positif, impair, quelconque.*

Définition du symbole de Jacobi – Soit $a \in \mathbb{Z}$ et $b \in \mathbb{N}$ *impair.* On décompose b en produit $b = p_1 \dots p_s$ de nombres premiers (non nécessairement distincts !), et on définit :

$$J(a,b) = \prod_{i=1}^{s} L(a,p_i)$$

Bien sûr on a toujours $J(a,b) \in \{-1,0,1\}$. Si $b = 1$, le produit est vide et $J(a,b) = 1$.

a) Déterminer une C.N.S. sur a, b pour que $J(a,b) = 0$.

b) Justifier que pour tout a, a' dans $\mathbb{Z}$ et b, b' impairs, positifs :

$$J(aa',b) = J(a,b).J(a',b) \quad \text{et} \quad J(a,bb') = J(a,b).J(a,b')$$

c) Si $a \wedge b = 1$, combien valent les symboles $J(a^2,b)$ et $J(a,b^2)$?

d) Calculer $J(2,15)$. En déduire qu'on peut avoir $J(a,b) = 1$ sans que a ne soit un carré modulo b.

e) Justifier que la loi de réciprocité de Gauss se généralise aux symboles de Jacobi, autrement dit que pour tout couple (a,b) d'entiers impairs, positifs, premiers entre eux :

$$J(a,b) = (-1)^{\frac{(a-1)}{2}\cdot\frac{(b-1)}{2}} J(b,a)$$

1. A lire à la maison : On pourrait s'étonner de la recherche d'un algorithme plus rapide quand on a dit que celui-ci est en $O(\log(p))$ Mais cette estimation a été faite en considérant que toutes les multiplications ont le même coût. En fait quand on fait des calculs sur les très grands entiers, ce modèle n'est plus pertinent, et avec un modèle plus fin, l'algo du 1.2. c) devient en $O((\log(p))^3)$. Dans ce qui suit on obtiendra au 1.5. un algo en $O((\log(p))^2)$.

f) On va illustrer l'intérêt du symbole de Jacobi pour le calcul du symbole de Legendre, en donnant une deuxième méthode de calcul de $L(95,191)$, déjà calculé au 1.3. b).

Bien sûr $L(95,191) = J(95,191)$ et en utilisant la loi de réciprocité pour les symboles de Jacobi, on a immédiatement $J(95,191) = -J(191,95) = -J(1,95) = -1$.

Ainsi $L(95,191) = -1$ et 95 n'est pas un carré modulo 191.

Info expliquer (sans rentrer dans les détails, donner juste une idée) l'intérêt de la différence entre ces deux méthodes en terme de complexité de programmation informatique.

1.5. Algorithme de calcul du symbole de Jacobi

Le calcul informatique du symbole de Jacobi $J(a,b)$ repose sur les six règles suivantes, qui découlent immédiatement des propriétés vues précédemment :

Trois règles de terminaison :

(R1) $J(0,b) = 0$

(R2) $J(1,b) = 1$

(R3) $J(2,b) = -1$ si $b \equiv 3$ ou $b \equiv 5$ mod. 8 et $J(2,b) = +1$ si $b \equiv 1$ ou $b \equiv 7$ mod. 8.

Trois règles de réduction :

(R4) (réduction des facteurs 2) : $J(2a,b) = J(2,b).J(a,b)$

(R5) (réduction modulo b) : si $a > b$ ou $a \leq 0$, $J(a,b) = J(a\%b,b)$ où $a\%b$ est le reste de la division euclidienne de a par b.

(R6) (réciprocité de Gauss pour les impairs premiers entre eux) : si a, b sont impairs et $a \wedge b = 1$ alors :

$$J(a,b) = \begin{cases} -J(b,a) & \text{si } a \equiv b \equiv 3 \ [4], \\ +J(b,a) & \text{sinon} \end{cases}$$

Programme Python :

```
def jacobi(a,b):
    j=1
    a=a%b # R?
    while a!=0:
        t=0
        while a%2==0:
            a=a//2
            t=t+1
        if t%2==1 and (b%8==3 or b%8==5):
            j=-j  # R?  et  R?
        if a%4==b%4 and a%4==3:
            j=-j # R?
        a,b=b%a,a
    if b==1:
        return j
    else :
        return 0
```

Info **Travail à faire**

a) Compléter (sur la feuille de script séparée p.7) les commentaires donnant la règle utilisée à chaque étape.

b) Expliquer le rôle de la variable `t`.

c) Expliquer le rôle de la variable `j`.

d) Justifier la terminaison de l'algorithme

e) Justifier que si au départ a et b sont premiers entre eux, alors à tout moment dans l'exécution de l'algorithme a et b restent premiers entre eux.

f) Justifier la correction de l'algorithme en distinguant les deux cas $a \wedge b \neq 1$ et $a \wedge b = 1$ au départ.

2. Algorithmes de calcul des racines carrées modulo p

2.1. Cas où $p \not\equiv 1\ [8]$: des formules rusées qui donnent un algo. évident.

Soit $p \in \mathbb{P}$ et $a \in \mathbb{Z}/p\mathbb{Z}^*$ dont on sait (par exemple grâce au calcul du symbole de Legendre $L(a,p)$) qu'il admet une racine carrée $x \in \mathbb{Z}/p\mathbb{Z}^*$. On cherche ici à *expliciter* une telle racine.

On sait donc d'après les résultats du 1.1., que $a^{(p-1)/2} = \overline{1}$.

On va distinguer suivant que $p \equiv 3\ [4]$ ou $p \equiv 1\ [4]$.

a) Montrer que si $p \equiv 3\ [4]$ alors $x = a^{(p+1)/4}$ est une racine carrée de a dans $\mathbb{Z}/p\mathbb{Z}$.

b) Montrer que si $p \equiv 1\ [4]$ et si $p \equiv 5\ [8]$,
- si $a^{(p-1)/4} = \overline{1}$, alors $x = a^{(p+3)/8}$ est une racine carrée de a dans $\mathbb{Z}/p\mathbb{Z}$.
- si $a^{(p-1)/4} = -\overline{1}$ alors $x = 2a(4a)^{(p-5)/8}$ est une racine carrée de a dans $\mathbb{Z}/p\mathbb{Z}$.

Les formules du a) et b) donnent des méthodes efficaces de calculs de ces racines carrées par exponentiation rapide. Reste donc le cas où $p \equiv 1\ [8]$ dans ce cas aucune formule explicite pour la racine carrée n'est connue. On va donner un algorithme au paragraphe suivant.

2.2. Cas où $p \equiv 1\ [8]$: pas de formules mais un algorithme

Entrée de l'algorithme : p un nombre premier tel que $p \equiv 1\ [8]$, et $a \in \mathbb{Z}$ dont on sait qu'il admet une racine carrée modulo p.

On dispose aussi un nombre $h \in \mathbb{Z}$ qui n'est pas un carré modulo p, qui sera fourni par une fonction auxiliaire non demandée.

Sortie de l'algorithme : un nombre x tel que $x^2 \equiv a\ [p]$.

Description de l'algorithme :

L'algorithme consiste à faire évoluer deux exposants $(e_1, e_2) \in \mathbb{N}^2$ tels qu'à chaque étape $a^{e_1}.h^{e_2} \equiv 1\ [p]$ (invariant de l'algorithme).

- Initialisation : on prend $e_1 = (p-1)/2$ et $e_2 = p-1$.
- Itération : à l'étape i : tant que e_1 est pair, on divise e_1 par 2.

Toujours à l'étape i, on divise aussi e_2 par 2, mais ensuite on distingue deux cas.
- •• Cas 1 : Si on a encore $a^{e_1}.h^{e_2} \equiv 1\ [p]$, l'étape i est finie.
- •• Cas 2 : Si à ce stade $a^{e_1}.h^{e_2} \not\equiv 1\ [p]$, alors on remplace e_2 par $e_2 + (p-1)/2$,
- Arrêt de l'algo. lorsque e_1 est impair. On va voir ci-dessous ce qu'on va renvoyer.

Info **Travail à faire sur cet algorithme**

Pour les justifications théoriques, on pourra noter $e_{1,i}$ et $e_{2,i}$ les valeurs contenues dans les variables e_1 et e_2 à la fin de l'étape i de l'algorithme.

On écrit le nombre pair $p-1$ sous la forme $(p-1) = 2^k q$ avec q est impair. Autrement dit k est la valuation 2-adique de $(p-1)$.

a) Avec cette notation $(p-1) = 2^k q$ et la description faite pour l'algorithme, on sait que l'algorithme s'arrête au bout de k étapes. Justifier qu'ici $k \geq 3$.

b) A l'étape i de la description, dans le cas 2, justifier qu'après avoir remplacé e_2 par $e_2 + \dfrac{p-1}{2}$ comme indiqué, on a bien $a^{e_1} h^{e_2} \equiv 1\ [p]$.

c) A l'arrêt de l'algorithme : e_1 est impair. Justifier que e_2 est pair (*demande un peu de soin.*)

d) A l'arrêt de l'algorithme : $a^{e_1+1} h^{e_2} \equiv a\ [p]$. En déduire une racine carrée de a modulo p.

e) Implémenter cet algorithme en Python en supposant qu'on dispose d'une fonction `noncarre(p)` qui prend en argument un nombre premier p et renvoie un nombre `h` $\in [\![1, \mathtt{p}-1]\!]$ qui n'est pas un carré modulo p.

f) Ecrire la fonction `noncarre` en testant pour un nombre $x \in [\![1, p-1\ ]\!]$ au hasard si $x^{(p-1)/2} \equiv -1\ [p]$ et en recommençant jusqu'à ce que cette condition soit vérifiée. Quelle est la probabilité que le programme `noncarre` termine en N tours ?

3 Calcul d'inverse modulo p puis modulo p^n

En guise d'excuse : *La suite logique de ce qui précède serait la recherche de racines carrées dans les $\mathbb{Z}/p^n\mathbb{Z}$, ce qui serait bien joli, mais un peu trop long ici. On se rabat sur le même problème pour les inverses... indépendant de ce qui précède. En fait les méthodes seraient voisines pour les racines carrées, mais on comprendra mieux plus tard l'idée qui se cache derrière les formules du 3.2.*

3.1. Modulo p

a) `Info`
On considère le script python suivant où les entrées a et b sont deux entiers positifs.

```
def AEE(a,b):
    u,v=1,0 # initialisation u_(i-1),v_(i-1)
    up,vp=0,1 # initialisation de u_i,v_i
    while b!=0:


        u,v,up,vp=up,vp,u-q*up,v-q*vp
        d= ??
    return  d,u,v
```

i) Compléter les deux lignes manquantes **sur la feuille de script p.7** et le ?? pour que le triplet (d,u,v) renvoyé par cette fonction ait pour valeur $(pgcd(a,b),u,v)$ avec $au+bv=1$.

ii) Justifier la correction de cet algorithme.

b) `Info` Déduire de cette fonction une fonction `Inverse(a,n)` qui teste si la classe $\bar{a}$ de a est inversible, et si oui, renvoie le représentant de $\bar{a}^{-1}$ dans $[\![0,n-1]\!]$.

3.2. Modulo p^n

Soit a et n deux nombres entiers premiers entre eux : on sait que la classe $\overline{a}$ de a dans $\mathbb{Z}/n\mathbb{Z}$ admet un inverse pour la multiplication, on la note $\overline{b}$ où $b\in\mathbb{Z}$.
Terminologie plus commode : on dira simplement que b est inverse de a modulo n.

a) Justifier qu'alors $b(2-ab)$ est inverse de a modulo n^2.

b) Avec les notations du a), on définit une suite (b_k) d'entiers tels que pour tout $k\in\mathbb{N}$, $b_k\in[\![0,n^{2^k}[\![$ par la relation de récurrence suivante :
- $b_0\equiv b\ [n]$
- pour tout $k\in\mathbb{N}^*$, $b_k=b_{k-1}(2-ab_{k-1})\ [n^{2^k}]$.

Montrer que pour chaque $k\in\mathbb{N}$, b_k est inverse de a modulo n^{2^k}.

c) `Info` Ecrire une fonction Python `InversePlusLoin(a,n,k)` qui prend en entrée des entiers positifs, a,n,k avec $a\wedge n=1$ et renvoie le représentant dans $[\![0,n^k[\![$ de l'inverse de a modulo n^k.

Feuille de script à rendre

Votre NOM :

```
def jacobi(a,b):
    j=1
    a=a%b # R
    while a!=0:
        t=0
        while a%2==0:
            a=a//2
            t=t+1
        if t%2==1 and (b%8==3 or b%8==5):
            j=-j  # R  et  R
        if a%4==b%4 and a%4==3:
            j=-j  # R
        a,b=b%a,a
    if b==1:
        return j
    else :
        return 0
```

```
def AEE(a,b):
    u,v=1,0 # initialisation u_(i-1),v_(i-1)
    up,vp=0,1 # initialisation de u_i,v_i
    while b!=0:


        u,v,up,vp=up,vp,u-q*up,v-q*vp
        d=
    return  d,u,v
```