Le but de cet exercice est d'implémenter un algorithme donnant la décomposition de Jordan d'une matrice. Pour ce faire, on n'utilisera que les commandes ne faisant pas appel, d'une manière ou d'une autre, à la forme de Jordan : par exemple `gausselim` ou `kernel`, mais pas `eigenvect` ou `minpoly`.

a. À l'aide des commandes `JordanBlock`, `diag` et `randmatrix`, construire une matrice dont la forme de Jordan ne soit pas trop simple, que nous utiliserons par la suite.

b. Montrer que A annule son polynôme charactéristique. parmi les polynômes qui annulent A, il y en a un plus petit (soit, pour le degré, soit pour la relation de divisibilité), appélé polynôme minimal. Quel est-il dans notre exemple ?

c. Que dire du polynôme minimal d'un endomorphisme diagonalisable ? Donner les vecteurs propres de A : forment-ils une base de $\mathbb{R}^n$? Si le polynôme minimal s'écrit $P(X) = \prod(X - \lambda_i)^{n_i}$, posons $E_i = \mathrm{Ker}(A - \lambda_i)^{n_i}$. Donner une base de chacun de ses sous-espaces. Montrer qu'ils sont en somme directe et qu'ils sont stables par A. Que dire de la matrice de A dans cette base ?

d. On restreint maintenant A à l'un des E_i et on considère l'endomorphisme nilpotent $B = A - \lambda_i \mathrm{Id}$. Notre problème de trouver une base dans laquelle un endomorphisme quelconque s'exprime simplement se trouve ramené au cas des endomorphismes nilpotents. Si $B^m = 0$ mais $B^{m-1} \neq 0$, il existe un vecteur x tel que $Bx \neq 0$. Montrer que $(x, Bx, \ldots, B^{m-1}x)$ est libre. Plus généralement, si $(x_1, \ldots, x_k)$ est une base d'un supplémentaire de $\mathrm{Ker}\, B^{m-1}$, alors

$$(x_1, Bx_1, \ldots, B^{m-1}x_1, \ldots, x_k, Bx_k, \ldots, B^{m-1}x_k)$$

est libre. Calculer une telle famille dans notre exemple.

e. Dans le cas général, on n'obtient pas une base : il faut continuer et chercher des vecteurs y qui forment une base d'un supplémentaire de $\mathrm{Ker}\, B^{m-2} + \mathrm{Vect}(x_1, \ldots, x_k)$ dans $\mathrm{Ker}\, N^{m-1}$.

f. Programmer tout ce qui précède en général.

g. Donner quelques applications de la forme de Jordan.

a. Il suffit de construire une matrice de Jordan (diagonale par blocks) et de prendre une matrice semblable. Je rappelle que le produit des matrices s'écrit `&*` (c'est la forme inerte de la multiplication) et qu'il est nécessaire de forcer Maple à évaluer les opérations sur les matrices à l'aide de la commande `evalm`.

```
restart;
with(linalg):
JordanBlock(1,1);
B:=diag(
  JordanBlock(1,1),
  JordanBlock(5,2),
  JordanBlock(5,2),
  JordanBlock(5,3));
fini:=false;
while(not(fini))
do
  Q:=randmatrix(8,8,entries=rand(-1..1));
  fini:=(abs(det(Q))=1);
od:
A:=evalm(Q^(-1)&*B&*Q);
```

On peut vérifier que la forme de Jordan est bien celle dont on était partis. Par contre, la matrice de passage n'est pas unique et Maple risque d'en trouver une autre.

```
jordan(A,'P');
evalm(P), evalm(Q);
```

b. Vérifions que le polynôme charactéristique de A annule A.

```
Pc:=factor(charpoly(A,t));
evalm(subs(t=A,Pc));
```

Essayons maintenant de trouver le polynôme minimal de A : il sera de la forme $(x-1)^a(x-5)^b$, avec $0 \leqslant a \leqslant 1$ et $0 \leqslant b \leqslant 7$. Comme Maple est incapable d'évaluer des expressions booléennes faisant intervenir des matrices, on utilise la commande `norm` pour se ramener à des nombres.

```
k:=7;
while(norm(evalm( (A-1)&*(A-5)^k ))=0)
do k:=k-1; od:
(t-1)*(t-5)^(k+1);
evalm((A-5)^3);
```

Le polynôme minimal de notre exemple est donc $(t-1)(t-5)^3$. (On pouvait le voir très facilement sur la forme de Jordan dont nous sommes partis.)

c. Si l'endomorphisme A était diagonalisable, on pourrait l'écrire sous forme diagonale pour calculer son polynôme minimal, qui serait alors $\prod(X-\lambda_i)$, où les λ_i sont ses valeurs propres *distinctes*. Nous allons bientôt voir que la réciproque est vraie : si le polynôme minimal n'a que des racines simples, alors l'endomorphisme est diagonalisable.

Pour obtenir les vecteurs propres de A, on peut utiliser la commande `kernel`, qui nous donne un ensemble de vecteurs qui forment une base du sous-espace propre correspondant. Signalons toutefois qu'il y a une commande `eigenvects`.

```
kernel(A-1); kernel(A-5);
```

On peut s'amuser à vérifier que les quatre vecteurs que l'on obtient forment bien une famille libre, par exemple en regardant le rang de cette famille (que Maple calcule à l'aide du pivot de Gauss, `gausselim`). La commande `evalb` sert à forcer l'évaluation du booléen, car Maple a tendance à ne pas évaluer les choses jusqu'au bout quand il est dans une procédure.

```
libre:=proc(s::set)
  evalb(rank(matrix([op(s)]))=nops(s));
end;
libre(kernel(A-1) union kernel(A-5));
```

On aurait pu écrire cette procédure « à la main », à l'aide du pivot de Gauss.

```
libre:=proc(l::list)
  local A,i,j,resultat;
  A:=gausselim(matrix(l));
  for i from 1 to rowdim(A) do
    resultat:=false;
    for j from 1 to coldim(A) do
      if A[i,j] <> 0
      then resultat:=true;
      fi;
    od;
    if( resultat=false )
    then RETURN(resultat);
    fi;
  od;
  true;
end;
libre([ [1,0,0],[0,1,0] ]);
libre([ [0,0,0], [1,0,0], [0,1,0] ]);
```

Pour obtenir une base des sous-espaces E_i (qui s'appellent *sous-espaces caractéristiques*), on procède de même. On vérifie qu'ils forment une famille libre : on a donc obtenu une base de $\mathbb{R}^8$.

```
kernel(A-1); kernel((A-5)^3);
libre(kernel(A-1) union kernel((A-5)^3));
```

Pour vérifier que les E_i sont stables par A, il suffit de remarquer que ma matrice de A dans une base adaptée à cette décomposition est diagonale par blocs : c'est bien le cas ici, mais l'un des blocs est très petit (1×1). Ici encore, il ne faut pas oublier d'utiliser `&*` pour le produit matriciel.

```
P:=transpose(matrix([op(kernel(A-1)),
                      op(kernel((A-5)^3))]));
evalm(P^(-1)&*A&*P);
```

d. Nous cherchons maintenant un vecteur x tel que $(A-5)^3x = 0$ mais $(A-5)^2x \neq 0$. Il suffit de prendre une base de $\mathrm{Ker}(A-5)^2$ et essayer de lui rajouter un à un les vecteurs d'une base de $\mathrm{Ker}(A-5)^3$: si on obtient une famille libre, on retient le vecteur. À la fin, on se retrouve avec une base d'un supplémentaire de $\mathrm{Ker}(A-5)^2$ dans $\mathrm{Ker}(A-5)^3$. C'est juste le théorème de la base incomplète. Dans le code suivant, il ne faut pas oublier de mettre `evalm` devant `x`.

```
supplementaire:=proc(A::set,B::set)
  local resultat, base, x;
  resultat:={};
  base:=A;
  for x in B
  do
    if( libre(base union {evalm(x)}) )
    then
      base:=base union {evalm(x)};
      resultat:=resultat union {evalm(x)};
    fi;
  od;
  resultat;
end;
supplementaire(kernel((A-5)^2), kernel((A-5)^3));
```

On vérifie maintenant que $x, (A-5)x, (A-5)^2x$ est libre.

```
x:="[1];
libre(map(evalm,{x,(A-5)&*x,(A-5)^2&*x}));
```

e. On poursuit en diminuant les exposants.

```
s:=supplementaire(basis(kernel(A-5) union
                        {evalm((A-5)&*x)}),
                  kernel((A-5)^2));
y:=s[1]; z:=s[2];
libre(map(evalm,{y,(A-5)&*y, z,(A-5)&*z}));
```

On dispose maintenant d'une base dans laquelle on peut exprimer notre matrice. On remarquera que l'on utilise des listes `[...]` et pas des ensembles `{...}`, da manière à garder l'ordre des vecteurs.

```
u:=kernel(A-1)[1];
```

```
base:=map(evalm,[u,
                 x,(A-5)&*x,(A-5)^2&*x,
                 y,(A-5)&*y, z,(A-5)&*z]);
libre({op(base)});
P:=transpose(matrix(base));
evalm(P^(-1)&*A&*P);
```

f. Le lecteur intéressé peut s'amuser à essayer de mettre tout cela dans une unique procédure — nous ne le ferons pas ici. Il remarquera qu'il ne suffit pas de concaténer les lignes précédentes. En effet, Maple n'évalue les choses qu'une seule fois à l'intérieur d'une procédure, alors qu'il les évalue jusqu'au bout en mode interactif.
Quelques autre remarques, en vrac. Je suggère d'écrire une procédure qui évalue un polynôme en une matrice : en général, cela marche bien avec la commande `subs` que nous avons utilisée plus haut, sauf pour les polynômes constants. On fera attention à l'indéterminée du polynôme (faut-il que ce soit une variable globale ? faut-il la passer en paramètre ?). Pour calculer le polynôme minimal, on aura besoin des différents facteurs du polynôme caractéristique ; la commande `factor` le factorise (si possible, voir `AFactor`), mais il faut encore séparer ces différents facteurs et récupérer les exposants : on pourra utiliser les commandes `type` et `op`. La lenteur du programme est due aux innombrables copies et évaluations de matrices et à certaines répétitions. À l'instar de la commande `jordan` de Maple, la nôtre ne marche pas toujours, car il arrive que Maple ne parvienne pas à factoriser le polynôme caractéristique.

Nous n'avons examiné l'algorithme de Jordan que sur un exemple et les justifications ,'étaient valables que pour cet exemple. Nous montrons maintenant que ces résultats sont valables en général.
Montrons tout d'abord que toute matrice A possède un polynôme annulateur. Comme la famille $\mathrm{Id}, A, A^2, \ldots, A^{n^2}$ contient n^2+1 éléments alors que $\mathscr{M}_n$ est de dimension n^2, il existe une relation de dépendance linéaire (non triviale), *i.e.*, un polynôme (non nul) de degré inférieur ou égal à n^2 annulant A.
Vérifions maintenant que les sous-espaces caractéristiques E_i sont stables par A. De manière plus générale, montrons que si P est un polynôme, alors $\operatorname{Ker} P(A)$ est stable par A. En effet, soit $x \in \operatorname{Ker} P(A)$: on a $P(A)(x) = 0$. Donc $A \circ P(A)(x) = 0$; mais A et $P(A)$ commutent. On a donc $P(A) \circ A(x) = 0$, *i.e.*, $P(A)(Ax) = 0$, *i.e.*, $Ax \in \operatorname{Ker} P(A)$.
Nous voulons maintenant comprendre pourquoi les E_i sont en somme directe. Cela découle du résultat suivant, appliqué par récurrence aux facteurs du polynôme P : Si $P \wedge Q = 1$, alors $\operatorname{Ker} PQ(A) = \operatorname{Ker} P(A) \oplus \operatorname{Ker} Q(A)$.

En effet, si $x \in \operatorname{Ker} P(A)$, *i.e.*, $P(A)(x) = 0$, alors $PQ(A)(x) = QP(A)(x) = Q(A)(P(A)(x)) = 0$, *i.e.*, $x \in \operatorname{Ker} PQ(A)$. On a donc $\operatorname{Ker} P(A) \subset \operatorname{Ker} PQ(A)$; de même, $\operatorname{Ker} Q(A) \subset \operatorname{Ker} PQ(A)$. Pour voir que la somme est directe, remarquons qu'il existe U et V tels que $PU + QV = 1$; donc si $x \in \operatorname{Ker} P(A) \cap \operatorname{Ker} Q(A)$, on a $x = UP(A)(x) + VQ(A)(x) = 0$. Il reste à prouver la dernière inclusion. Si $x \in \operatorname{Ker} PQ(A)$, on peut écrire $x = UP(A)(x) + VQ(A)(x)$. Posons $x_1 = UP(A)(x)$ et $x_2 = VQ(A)(x)$ et remarquons que $x_1 \in \operatorname{Ker} Q(A)$: en effet, $Q(A)(x_1) = (QUP)(A)(x) = U(A)(PQ(A)(x_1)) = 0$. De même, on vérifie que $x_2 \in \operatorname{Ker} P(A)$. On a donc vien $x \in \operatorname{Ker} P(A) \oplus \operatorname{Ker} Q(A)$.
Comme les E_i soient stables par A, la matrice de A dans une base adaptée à la décomposition $\mathbb{R}^n = \bigoplus E_i$ est diagonale par blocs.

g. Donnons maintenant, en vrac et sans détails, quelques applications de la forme de Jordan.
Le polynôme caractéristique est un polynôme annulateur.
Quelles sont les matrices X de taille trois telles que $X^3 = X$?
Quelles sont les matrices qui possèdent une racine carrée?
etc.