L'objet de cet exercice est de faire manipuler (définir et appeler) des procédures dans le cadre d'un exercice de géométrie.

1. Consignes - Outils

Toutes les définitions des procédures que vous écrirez doivent se trouver dans la même zone d'exécution (bibliothèque).

Dans cet exercice, un plan est muni d'un repère orthonormé. Les fonctions coordonnées sont désignées par les variables `x` et `y`.

Il est très important de ne jamais rien assigner à ces variables.

Vous serez amené à utiliser les procédures `subs`,`expand`, `vect_normal`, `solve`, `plot`.

1. La procédure `subs` permet de substituer des expressions à l'intérieur d'une autre expression.Elle figure parmi les procédures de base que connait Maple. Considérons par exemple l'appel `subs({x=truc,lambda=bidule},expr)` où `expr` est une expression (arbre) contenant les variables `x` et `y`. Cet appel va renvoyer le résultat de la substitution dans l'expression de chaque ocurrence de `x` par `truc` et de `lambda` par `bidule`.
   Par exemple, si `expr` désigne `3*x+2*y+c`, l'appel de `subs({x=A,y=B},expr)` va renvoyer `3*A+2*B+c`. Il est utile de bien remarquer que le premier argument de `subs` est un ensemble d'égalités.
2. La procédure `expand` sert à développer.
3. L'appel `vect_normal(expr)` renvoie la liste `[a,b]` des coefficients lorsque `expr` désigne une expression de la forme `a*x+b*y+c`. La définition de cette fonction est la suivante :

```
> vect_normal := proc(expr)
    local a,b,c;
    c:= subs({x=0,y=0},expr);
    a:= subs({x=1,y=0},expr) -c;
    b:= subs({x=0,y=1},expr) -c;
    return[a,b];
  end proc:
```

   Vous reproduirez ce code dans votre bibliothèque pour pouvoir utiliser cette procédure.
4. La procédure `solve` permet de résoudre des systèmes d'équation. Lors de l'appel, le premier argument doit être un ensemble d'équations et le deuxième un ensemble de variables (inconnues).
5. La procédure `plot` permet de former plusieurs sortes de dessin. La syntaxe qui nous interesse ici est celle qui permet de tracer le support d'une courbe paramétrée. Lorsque `u` et `v` désignent deux expressions de `t`, l'appel `plot([u,y,t=0..1])` renvoie le support de la courbe paramétrée avec le paramètre variant entre 0 et 1.

2. Procédures

1. Définir une procédure `eq_dte` réalisant la tâche suivante.
   Lorsque `A` et `B` désignent des listes de deux nombres représentant les coordonnées de deux points, l'appel `eq_dte(A,B)` renvoie une expression de `x` et `y` dont le ligne de niveau 0 est la droite passant par les deux points. On appelera *équation de la droite* une telle expression.
   Par exemple, l'appel `eq_dte([0,1],[1,2])` renvoie l'expression `x-y+1`.
2. Définir une procédure `proj_dte` réalisant la tâche suivante.
   Lorsque `M` désigne la liste des deux coordonnées d'un point, et `expr` une équation de droite, l'appel `proj_dte(expr,M)` doit renvoyer la liste des deux coordonnées du projeté orthogonal du point sur la droite.
3. Définir une procédure `sym_dte` réalisant la tâche suivante.
   Lorsque `M` désigne la liste des deux coordonnées d'un point, et `expr` une équation de droite, l'appel `sym_dte(expr,M)` doit renvoyer la liste des deux coordonnées du symétrique orthogonal du point par rapport à la droite.

3. Exercice de géométrie plane

Soit A, B, C un triangle équilatéral. Pour tout point M du plan, on note P, Q, R les symétriques respectifs de M par rapport aux droites (BC), (CA), (AB).

1. Montrer que les trois droites (AP), (BQ), (CR) sont concourantes sauf si M est sur un certain ensemble à préciser. On note N le point d'intersection.
2. Tracer l'ensemble des points N lorsque M décrit le cercle circonscrit au triangle A, B, C.