CORRECTION

QUESTION 1

1) `A:=point(-53,-28):;B:=point(17,35):;eq:=equation(droite(A,B))`

$$(\text{Done, Done, y} = (\frac{9 * \text{x}}{10} + \frac{197}{10}))$$

2) `simplifier(10*droit(eq)-10*gauche(eq))=0)`

$$(9 * \text{x} - 10 * \text{y} + 197) = 0$$

QUESTION 2

Voici un algorithme possible :

```
saisir a,b
temp:=9a-10b+197
si temp==0 alors
  afficher("M est sur la droite (AB)")
  retourne 1
sinon
  afficher("M n'est pas sur la droite (AB)")
  retourne 0
fsi
```

En langage Xcas on obtient le programme suivant :
(les // indiquent un commentaire ignoré par le compilateur)
(consultez l'aide pour comprendre la syntaxe des commandes)

```
SurLaDroite(a,b):={ // NomDeLaFonction(parametre1,parametre2):={
  local temp; //temp est une variable qui n'existe qu'à l'intérieur de ce programme
  temp:=9a-10b+197; // temp vaut 0 si et seulement si M est sur la droite
  si temp==0 alors
    //affichage d'une chaîne de caractères "..." dans la zone intermédiaire
    //en bleu au dessus de la valeur de retour
    afficher("M est sur la droite (AB)")
    retourne 1 // valeur de retour 1 ou vrai si le point est sur la droite
  sinon
    afficher("M n'est pas sur la droite (AB)")
    retourne 0 // valeur de retour 0 ou faux si le point n'est pas sur la droite
  fsi
} // } pour refermer le bloc d'instructions
:; // facultatif
```

```
SurLaDroite(a,b):={ // NomDeLaFonction(parametre1,parametre2):={
  local temp; //temp est une variable qui n'existe qu'à l'intérieur de ce programme
  temp:=9a-10b+197; // temp vaut 0 si et seulement si M est sur la droite
  si temp==0 alors
    //affichage d'une chaîne de caractères "..." dans l'écran de géométrie
    retourne legende([50,50],"M est sur la droite (AB)")
  sinon
    retourne legende([50,50],"M n'est pas sur la droite (AB)")
  fsi
} // } on referme le bloc d'instructions
:; // facultatif
```

Le même programme, plus lisible, en ne conservant que le commentaire important :

```
SurLaDroite(a,b):={
  local temp;
  temp:=9a-10b+197; // temp vaut 0 si et seulement si M est sur la droite
  si temp==0 alors
    retourne legende([50,50],"M est sur la droite (AB)")
  sinon
    retourne legende([50,50],"M n'est pas sur la droite (AB)")
  fsi
}
:;
```

QUESTION 3

Les élèves rapides pourront élaborer un programme pour rechercher tous les couples d'entiers (x, y) vérifiant $9x - 10y + 197 = 0$, avec $p \leqslant x \leqslant n$.

```
CoordEntieres(p,n):={
  local L,x,y;
  L:=NULL; // ou L:=[] (voir question 5)
  pour x de p jusque n faire
    y:=(9x+197)/10;
    si irem(9x+97,10)==0 alors
      L:=L,[x,y]; // ou L:=append(L,[x,y]) (voir question 5)
    fsi
  fpour
  retourne L
}
```

Pour les élèves rapides et désireux d'aller en Terminale S (Spécialité Math) sachez qu'il est possible de trouver la forme générale des solutions entières de l'équation $9x - 10y + 197 = 0$. Montrez (c'est facile) la proposition suivante : si $M(-3 + 10k\,;17 + 9k)$ et k entier alors M appartient à la droite d'équation $9x - 10y + 197 = 0$ et a des coordonnées entières. Question ouverte (ce n'est pas très facile !) : prouvez que la proposition réciproque est vraie. Sinon attendez deux ans !

QUESTION 4

1) `normal(subst(eq,x=127))`

$$\mathrm{y} = 134$$

2) `sol:=solve(subst(eq,y=-73),x);sol[0]`

$$[-103], -103$$

Notez la différence entre `sol` qui contient une liste et `sol[0]` qui contient le premier élément de cette liste.

QUESTION 5

```
Faisceau():={ // fonction sans paramètre
  local L,m; // déclaration des variables locales
  m:=-12; // initialisation de la variable m
  L:=NULL; // L est une séquence vide avant d'entrer dans la boucle
  tantque m<=15 faire // début de la boucle
    L:=L,droite(y=m*x+53m-28); // à chaque passage la séquence contient une droite de plus
    m:=m+1; // incrémentation (augmentation) de la variable m de 1
  ftantque // fin de la boucle
  retourne L
}
```

Voici une autre version plus légère où la boucle tantque est remplacée par une boucle pour.
Dans ce programme L est une liste.
Remarquez la commande `append` pour ajouter un élément en fin de liste.

```
Faisceau():={
  local L,m;
  L:=[]; // L est une liste vide avant d'entrer dans la boucle
  pour m de -12 jusque 15 faire // début de la boucle
    // à chaque passage la liste contient une droite de plus
    L:=append(L,droite(y=m*x+53m-28));
  fpour // fin de la boucle
  retourne L // la valeur de retour est une liste d'objets de type droite
}
```

QUESTION 6

Le programme ci-dessous plante car si $m = 0$ alors `sol` est une liste vide.
`sol[0]` équivaut à `[][0]` qui renvoie `Erreur: Dimension incorrecte`.

```
AbsInter1():={
  local L,m,sol;
  L:=[]; // L est une liste vide avant d'entrer dans la boucle
  pour m de -12 jusque 15 faire
    sol:=solve(m*x+53m-28=0,x); // sol: liste, sol[0]: le premier élément de la liste
    L:=append(L,sol[0]); // à chaque passage la liste contient une abscisse de plus
  fpour
  retourne L
}

AbsInter2():={
  local L,m,sol,c;
  L:=[];
  c:=0; // pour compter le nombre de points d'intersection
  pour m de -12 jusque 15 faire
    // si m=0 on passe à l'itération suivante, c n'est pas incrémenté
    si m==0 alors continue fsi;
    sol:=solve(m*x+53m-28=0,x);
    L:=append(L,sol[0]);
    c:=c+1; // incrémentation du compteur de points
  fpour
  afficher("ce faisceau coupe l'axe des abscisses en "+c+" points")
  afficher("voici la liste des abscisses de ces points")
  retourne L // valeur de retour ici une liste de 27 éléments
}
```