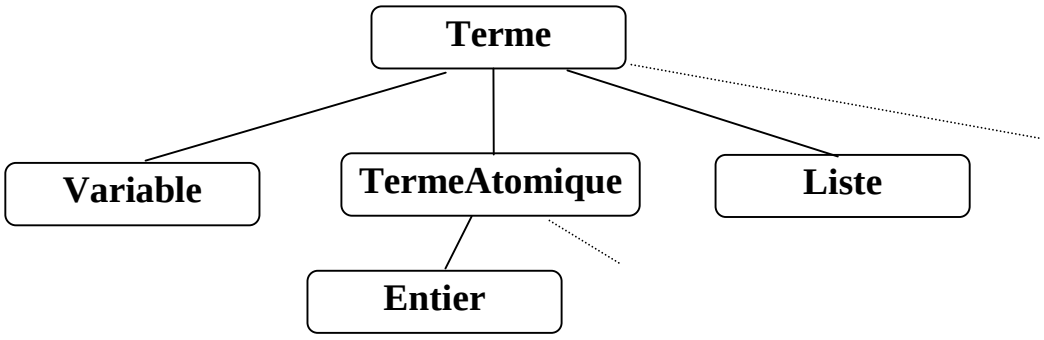
programmation

examen

La hiérarchie suivante décrit les « objets » manipulés par le langage prolog :



Nous nous proposons de construire une hiérarchie de classes permettant de manipuler ces différents objets. Chaque classe sera munie de quatre fonctions membres :

- un constructeur.
- *est_un* qui retourne le type de l'objet.
- *affiche* qui affiche à l'écran un objet de la classe.
- *unifie* qui renvoie *true* si un objet est unifiable avec un autre objet en argument et *false* sinon.

Question 1.

Parmi les classes de la hiérarchie, quelles sont celles qui sont abstraites et pourquoi ? Quelles sont, dans ce cas, celles de leurs fonctions membres qui sont déclarées virtuelles pures ? Définir complètement la classe *Terme*, ainsi que la classe *TermeAtomique*.

Question 2.

Ecrire la définition de la fonction membre *Affiche* pour la classe *Variable*, et pour la classe *Liste*.

- L'affichage d'une variable se fait en affichant le nom de la variable si la variable est libre, et en affichant le terme auquel elle est liée sinon.
- L'affichage d'une liste se fait en affichant la liste des termes entre [et], les termes étant séparés par des virgules. La liste vide est représentée par un objet de la classe *Liste* dont les membres *tête* et *queue* sont égaux à NULL.

Question 3.

Ecrire la définition de la fonction membre *Unifie* pour les classes *Variable*, *Entier*, et *Liste* :

- L'unification d'une variable avec un terme *t* est définie par : si la variable est libre, elle est liée au terme *t* et l'unification réussit, sinon l'unification réussit si on peut unifier le terme auquel la variable est liée avec *t*.
- L'unification d'un entier *e* avec un terme *t* est définie par : si le terme *t* est une variable, l'unification réussit si le terme *t* est unifiable à *e*, sinon l'unification réussit

si le terme t est un entier de valeur égale à l'entier *e*.

- L'unification d'une liste *l* avec un terme *t* est définie par : si le terme *t* est une variable, l'unification réussit si la variable *t* est unifiable à *l*, sinon l'unification réussit si le terme *t* est une liste, et que la tête s'unifie avec la tête de *t* , et que la queue s'unifie avec la queue de *t*.

annexe :

```
enum type {terme, variable, liste, termeatomique, atome, entier,
compose};
classe Terme { ...};
classe TermeAtomique : public Terme { ... };
```

```
class Variable : public Terme
{  public :
           Variable(char * n){ strcpy(nom, n); lieea = NULL;}
           void Affiche();
           bool Unifie( Terme * t);
           type est_un (){return variable;}

           Terme * lieea;
           char nom[20];
};
```

```
class Liste : public Terme
{  public :
           Liste(Terme * t, Terme * q){tete = t; queue = q;}
           void Affiche();
           bool Unifie( Terme * t);
           type est_un (){return liste;}

           Terme * tete;
           Terme * queue;
};
```

```
class Entier : public TermeAtomique
{  public :
           Entier(int a){ valeur = a;}
           void Affiche();
           bool Unifie( Terme * t);
           type est_un (){return entier;}

           int valeur;
};
```