**Chapitre 3 - Parcours d’un graphe**

***Algorithme de parcours en profondeur***

initialiser Visite à faux;

Profond(G,x) :

{Parcours en profondeur du graphe G à partir du sommet x}

Début

Visite[x] = vrai;

{Traiter x en premier visite}

Pour chaque voisin y de x faire

Si Visite[y] = faux alors

profond(G,y);

{sinon on détecte une revisite de y}

{Traiter x en dernière visite}

Fin

***Algorithme de parcours en profondeur - version généralisée***

initialiser Visite à faux;

Pour i de 1 à n faire

Si Visite[i] = faux alors

profond(G,i);

***Algorithme de parcours en largeur***

initialiser Visite à faux;

largeur(G,x) :

{parcours en largeur du graphe G à partir du sommet x}

F:=[x]; Visite[x] := vrai

Tant que F n’est pas vide faire

Début

considérer y la tête de F (et l’enlever de F)

{traiter y}

pour chaque successeur z de y faire

si Visite[z] = faux alors

Début

Visite[z]:=vrai

ajouter z à la fin de la liste F

Fin

Fin

***Algorithme de plus court chemin***

trouvé := faux;

initialiser Visite à faux;

initialiser Dist à l’infini;

F:=[x]; Visite[x] := vrai;

Dist[x] :=0;

Pere[x] :=x;

tant que F <> [ ] faire

Début

y := enleveTete(F);

Pour chaque successeur z de y faire

si Visite[z] = faux alors

Début

Visite[z] := vrai;

ajoutFin(z,F);

Dist[z] := Dist[y] + 1;

Pere[z] = y;

Fin

Fin