Raffinage du routeur simple

R0: Réaliser un routeur simple	
Type T_Element est générique Type T_Cellule est ENREGISTREMENT (Valeur : T_Element; Suivant : T_LC;) Type T_Table est POINTEUR de T_Cellule Type T_IP est ENTIER modulo 2**32	
R1: Comment « Réaliser un routeur simple » ?	
Lire des paramètres d'exécution Initialiser la table de routage Gérer les paquets	
R2: Comment « Lire des paramètres d'exécution » ?	
Mettre les paramètres aux valeurs par défaut i <- 1 -- Indice de l'argument de la commande **TantQue** i <= Argument_Count **Faire** -- i est modifié dans la boucle (cf R3: Lire le paramètre i) Exploiter le paramètre i **FinTQ**	Afficher_Stats: out; Nom_Fichier_Table: out; Nom_Fichier_Paquet: out; Nom_Fichier_Resultats: out i: in out; Afficher_Stats: in out; Nom_Fichier_Table: in out; Nom_Fichier_Paquet: in out; Nom_Fichier_Resultats: in out
R2: Comment « Initialiser la table de routage »	
Lire la table de routage dans le fichier Initialiser la liste chaînée	

R2: Comment « Lire la table de routage dans le fichier » ?	
Ouvrir(Fichier_Entree, Nom_Fichier_Table) **Si** exception Nema_Error **Alors** -- Cette exception arrête le programme Lever Ouverture_Impossible **Sinon** Ne rien faire **FinSi** Initialiser(Table) **Répéter** Texte <- Lire_Ligne(Fichier_Entree) Exploiter l'adresse Ajouter l'adresse à la table **JusquA** fin du fichier atteinte Fermer(Fichier_Entree)	 Table: out; Texte: in out; Adresse: out; Interface: out; Table: in out; Adresse: in; Interface: in;
R2: Comment « Gérer les paquets » ?	
Créer(Fichier_Sortie, Nom_Fichier_Resultats) Ouvrir(Fichier_Entree, Nom_Fichier_Paquet) **Si** exception Nema_Error **Alors** -- Cette exception arrête le programme Lever Ouverture_Impossible **Sinon** Ne rien faire **FinSi** **Répéter** Paquet <- Lire_Ligne(Fichier_Entree) **Si** Paquet est un paquet **alors** Convertir Paquet en adresse IP Déterminer l'interface avec la table de routage Ecrire le paquet et la sortie dans le fichier résultat **Sinon** Effectuer la commande **FinSi** **JusquA** fin du fichier atteinte Fermer(Fichier_Entree)	 Paquet: in; Adresse: out; Table: in; Adresse: in; Interface: out; Interface: in; Adresse: in; Nom_Fichier_Resultat: in; Adresse: in; Table: in; Cache: in;
R3: Comment « mettre les paramètres aux valeurs par défaut » ?	

```
-- Capacité du cache spécifiée par l'utilisateur
Capacite_Cache <- 0
-- Décision d'afficher ou non les statistiques
Afficher_Stats <- Vrai
-- Nom par défaut du fichier où est écrite la table
Nom_Fichier_Table <- "table.txt"
-- Nom par défaut du fichier où sont écrits les paquets
Nom_Fichier_Paquet <- "paquets.txt"
-- Nom par défaut du fichier où sont écrites les tables
Nom_Fichier_Resultats <- "resultats.txt"
```

R3: Comment « Exploiter le paramètre i » ?

```
Cle <- Lire_Paramètre
i <- i + 1
Si (i > Argument_Count) et alors (Cle = "-p" ou "-t" ou "-r") Alors
-- Cette exception ignore le dernier argument
        Lever Erreur_Dernier_Argument
SinonSi Cle = "-p" Alors
        Nom_Fichier_Paquet <- Lire_Paramètre
SinonSi Cle = "-S" Alors
        Afficher_Stats <- Faux
        i <- i - 1
SinonSi Cle = "-s" Alors
        Afficher_Stats <- Vrai
        i <- i - 1
SinonSi Cle = "-t" Alors
        Nom_Fichier_Table <- Lire_Paramètre
SinonSi Cle = "-r" Alors
        Nom_Fichier_Resultats <- Lire_Paramètre
Sinon
-- Cette exception ignore le paramètre courant
        Lever Parametre_Inconnu
FinSelon
```

R3: Comment « Convertir Paquet en adresse IP » ?

```
UN_OCTET <- 2**8
Adresse <- Entier(Paquet)
Adresse <- Adresse * UN_OCTET + Entier(Paquet)
Adresse <- Adresse * UN_OCTET + Entier(Paquet)
Adresse <- Adresse * UN_OCTET + Entier(Paquet)
```

R3: Comment « Ecrire le paquet et la sortie fichier résultat » ?

Ouvrir(Fichier_Sortie, Nom_Fichier_Resultat) -- & est le 'et' bit à bit Ecrire(Fichier_Sortie, Adresse_Destination & " " & Interface) Nouvelle_Ligne(Fichier_Sortie)	
R3: Comment « Effectuer la commande » ?	
Numero_Ligne = Ligne_Courante(Fichier_Entree) **Si** Texte = "table" **Alors** Ecrire("table (ligne ", Numero_Ligne, ")") Afficher(Table) **SinonSi** Texte = "stats" **Alors** Ecrire("stats (ligne ", Numero_Ligne, ")") Ecrire("Au cours du programme, “, i, “demandes ont étés effectuées.") **SinonSi** Texte = "fin" **Alors** Ecrire("fin (ligne ", Numero_Ligne, ")") **Sinon** -- Cette exception ignore la ligne courante de Fichier_Paquets Lever EXCEPTION_COMMANDE_INCONNUE **FinSelon**	Table: in;
R3: Comment « Déterminer l’interface avec la table de routage » ?	
Table_Routage <- Table Interface <- "" Continuer <- Vrai **TantQue** non FinTable(Table_Routage) et Continuer **Faire** -- & est le 'et' bit à bit **Si** (Paquet & Table_Routage.Masque) = … (Table^.Adresse & Table_Routage^.Masque) **Alors** Interface <- Table_Routage^.Interface Continuer <- Faux **SinonSi** Table_Routage^.Suivant = Null **Alors** Interface <- Table_Routage^.Interface Continuer <- Faux **Sinon** Ne rien faire **FinSi** Table <- Table^.Suivant **FinTQ**	

Raffinage des opérations d'un routeur LL (politique = FIFO, LRU, LFU)

Modifications mineures au routeur simple

```
Type T_Cache est POINTEUR de T_Cellule_Cache

-- type utilisé si cache LFU
Type T_Cellule_Cache est ENREGISTREMENT (
        Adresse : T_IP;
        Masque : T_IP;
        Interface_Nom : Unbounded_String;
        -- type utilisé si cache LFU
        Frequence : Entier;
)
```

R3: Comment « Exploiter le paramètre i » ?

```
Cle <- Lire_Paramètre
i <- i + 1
Si (i > Argument_Count) et alors (Cle = “-p” ou “-t” ou “-r”) Alors
-- Cette exception ignore le dernier argument
        Lever Erreur_Dernier_Argument
SinonSi Cle = "-c" Alors
        Capacite_Cache <- Lire_Paramètre
SinonSi Cle = "-p" Alors
        Nom_Fichier_Paquet <- Lire_Paramètre
SinonSi Cle = “-P” Alors
        Si Lire_Paramètre != “FIFO”, “LRU” ou “LFU” Alors
                Lever Erreur_Politique_Incorrecte
        Sinon
                Politique_Cache <- Lire_Paramètre
        FinSi
SinonSi Cle = "-S" Alors
        Afficher_Stats <- Faux
        i <- i - 1
SinonSi Cle = "-s" Alors
        Afficher_Stats <- Vrai
        i <- i - 1
SinonSi Cle = "-t" Alors
        Nom_Fichier_Table <- Lire_Paramètre
SinonSi Cle = "-r" Alors
        Nom_Fichier_Resultats <- Lire_Paramètre
```

Sinon -- Cette exception ignore le paramètre courant Lever Parametre_Inconnu **FinSelon**	
R3: Comment « mettre les paramètres aux valeurs par défaut » ?	
-- Capacité du cache spécifiée par l'utilisateur Capacite_Cache <- 0 Initialiser(Cache) -- Politique du cache spécifiée par l'utilisateur Politique_Cache <- FIFO -- Décision d'afficher ou non les statistiques Afficher_Stats <- Vrai -- Nom par défaut du fichier où est écrite la table Nom_Fichier_Table <- "table.txt" -- Nom par défaut du fichier où sont écrits les paquets Nom_Fichier_Paquet <- "paquets.txt" -- Nom par défaut du fichier où sont écrites les tables Nom_Fichier_Resultats <- "resultats.txt"	Cache: out
R3: Comment « Effectuer la commande » ?	
Numero_Ligne = Ligne_Courante(Fichier_Entree) **Si** Texte = "table" **Alors** Ecrire("table (ligne ", Numero_Ligne, ")") Afficher(Table) **SinonSi** Texte = "cache" **Alors** Ecrire("cache (ligne ", Numero_Ligne, ")") Afficher(Cache) **SinonSi** Texte = "stats" **Alors** Ecrire("stats (ligne ", Numero_Ligne, ")") Ecrire("Au cours du programme, ", i, "demandes ont étés effectuées.") **SinonSi** Texte = "fin" **Alors** Ecrire("fin (ligne ", Numero_Ligne, ")") **Sinon** -- Cette exception ignore la ligne courante de Fichier_Paquets Lever EXCEPTION_COMMANDE_INCONNUE **FinSelon**	Table: in; Cache: in;
R0: Utiliser le cache LL	
-- Pour Comment « Déterminer l'interface avec la table de routage » cf Routeur Simple **R1: Comment** « Déterminer l'interface pour une adresse IP » ?	

Déterminer l'interface avec le cache **Si** non Est_Trouve **Alors** Déterminer l'interface avec la table de routage Ajouter Adresse au cache **FinSi**	Cache: in; Adresse: in; Interface: out Est_Trouve: out Table: in; Adresse: in; Interface: out; Table: in; Adresse:in; Cache:in
R2: Comment « Déterminer l'interface avec le cache » ?	
Déterminer l'interface avec une liste chaînée **Si** Politique = LFU et Est_Trouve **Alors** Element_Trouve^.Frequence <- … Element_Trouve^.Frequence + 1 Supprimer Element_Trouve de Cache Insérer nouvelle adresse selon ordre fréquence **SinonSi** Politique = LRU et Est_Trouve **Alors** Placer Element_Trouve en premiere position **Sinon** Ne rien faire **FinSi**	Cache: in; Adresse: in; Interface: out Element_Trouve, Est_Trouve: out Cache: in-out; Element_Trouve: in Cache: in-out; Element_Trouve^.Frequence: in Element_Trouve^.Masque: in; Element_Trouve^.Adresse: in
R3: Comment « Déterminer l'interface avec une liste chaînée » ?	
Table_Routage <- Cache Interface <- "" Continuer <- Vrai **TantQue** non FinTable(Table_Routage) et Continuer **Faire** -- & est le 'et' bit à bit **Si** (Paquet & Table_Routage.Masque) = … (Table^.Adresse & Table_Routage^.Masque) **Alors** Interface <- Table_Routage^.Interface Continuer <- Faux **SinonSi** Table_Routage^.Suivant = Null **Alors** Interface <- Table_Routage^.Interface Continuer <- Faux **Sinon** Ne rien faire **FinSi** Table <- Table^.Suivant **FinTQ**	

R3: Comment « Ajouter Adresse au cache » ?	
Déterminer masque le plus long qui discrimine la route **Si non** Est_Routable **Faire** Lever Route_Pas_Dans_Cache **SinonSi** (Politique_Cache = LRU) **ou** (Politique_Cache = LFU) **Faire** Extraire le début du cache Frequence <- Frequence +1 Insérer la route temporaire à la fin du cache **Sinon** Rien **FinSi**	Masque_Cache, Est_Routable: out; Adresse,Masque, Table: in; Cache: in-out; Frequence, Masque_Cache, Adresse: out; Cache: in-out; Masque_Cache, Frequence, Adresse: in;
R4: Comment « Insérer nouvelle adresse selon ordre fréquence » ?	
Si Cache = Null **Alors** Enregistrer(Cache, Adresse, Frequence, … MasqueCache, Null) **SinonSi** Cache^.Frequence < Frequence **Alors** Enregistrer(Cache, Adresse, Frequence, … MasqueCache, Cache) **Sinon** Insérer nouvelle adresse selon ordre fréquence **FinSi**	 Cache^.Suivant: in-out; Frequence, Masque_Cache, Adresse: in;

Raffinage des opérations d'un routeur LA (politique = LRU)

Modifications mineures au routeur simple

```
Type T_Arbre est ENREGISTREMENT (
        t : Entier;
        Masque : Entier;
        Adresse: Entier;
        Interface: String;
        Fils_Droit: Pointeur;
        Fils_Gauche: Pointeur;
)

{variable globale initialement à 0}
IdGlobal : Entier <- 0
```

R3: Comment « Exploiter le paramètre i » ?

```
Cle <- Lire_Paramètre
i <- i + 1
Si (i > Argument_Count) et alors (Cle = “-p” ou “-t” ou “-r”) Alors
-- Cette exception ignore le dernier argument
        Lever Erreur_Dernier_Argument
SinonSi Cle = "-c" Alors
        Capacite_Cache <- Lire_Paramètre
SinonSi Cle = "-p" Alors
        Nom_Fichier_Paquet <- Lire_Paramètre
SinonSi Cle = “-P” Alors
        Si Lire_Paramètre != “LRU” Alors
                Lever Erreur_Politique_Incorrecte
        Sinon
                Politique_Cache <- Lire_Paramètre
        FinSi
SinonSi Cle = "-S" Alors
        Afficher_Stats <- Faux
        i <- i - 1
SinonSi Cle = "-s" Alors
        Afficher_Stats <- Vrai
        i <- i - 1
SinonSi Cle = "-t" Alors
        Nom_Fichier_Table <- Lire_Paramètre
SinonSi Cle = "-r" Alors
```

Nom_Fichier_Resultats <- Lire_Paramètre **Sinon** -- Cette exception ignore le paramètre courant Lever Parametre_Inconnu **FinSelon**	
R3: Comment « mettre les paramètres aux valeurs par défaut » ?	
-- Capacité du cache spécifiée par l'utilisateur Capacite_Cache <- 0 Initialiser(Cache) -- Politique du cache spécifiée par l'utilisateur Politique_Cache <- LRU -- Décision d'afficher ou non les statistiques Afficher_Stats <- Vrai -- Nom par défaut du fichier où est écrite la table Nom_Fichier_Table <- "table.txt" -- Nom par défaut du fichier où sont écrits les paquets Nom_Fichier_Paquet <- "paquets.txt" -- Nom par défaut du fichier où sont écrites les tables Nom_Fichier_Resultats <- "resultats.txt"	Cache: out
R3: Comment « Effectuer la commande » ?	
Numero_Ligne = Ligne_Courante(Fichier_Entree) **Si** Texte = "table" **Alors** Ecrire("table (ligne ", Numero_Ligne, ")") Afficher(Table) **SinonSi** Texte = "cache" **Alors** Ecrire("cache (ligne ", Numero_Ligne, ")") Afficher(Cache) **SinonSi** Texte = "stats" **Alors** Ecrire("stats (ligne ", Numero_Ligne, ")") Ecrire("Au cours du programme, ", i, "demandes ont étés effectuées.") **SinonSi** Texte = "fin" **Alors** Ecrire("fin (ligne ", Numero_Ligne, ")") **Sinon** -- Cette exception ignore la ligne courante de Fichier_Paquets Lever EXCEPTION_COMMANDE_INCONNUE **FinSelon**	Table: in; Cache: in;
R0: Utiliser le cache LA avec la politique LRU	
R1: Comment « Déterminer l'interface avec le cache » ?	

{On ne raffine que la politique LRU} {IdGlobal est une variable globale initialement égale à 0} **Si** cache = Null **Alors** Est_Trouve = Faux **SinonSi** Est_Feuille(Cache) and (Paquet & Cache^.Masque) = … (Cache^.Adresse & Cache^.Masque) **Alors** Est_Trouve = Vrai Cache^.t <- IdGlobal + 1 IdGlobal <- IdGlobal + 1 **Sinon** **Si** Valeur du i-ème bit de Cache^.Adresse **Alors** Determiner l'interface … avec le cache = Cache^.Fils_Droit **Sinon** Determiner l'interface … avec le cache = Cache^.Fils_Gauche **FinSi** Cache^.t = Trouver le minimum des temps de … (Cache^.Fils_Gauche, Cache^.Fils_Droit) **FinSi**	 Adresse: in ; Cache^.Fils_Droit: in out Adresse: in ; Cache^.Fils_Gauche: in out Cache^.t : out Cache^.Fils_Gauche : in Cache^.Fils_Droit : in
R2: Comment « Ajouter Adresse au cache » ?	
Déterminer masque le plus long qui discrimine la route **Si non** Est_Routable **Faire** Lever Route_Pas_Dans_Cache **SinonSi** (Politique_Cache = LRU) **…** **ou** (Politique_Cache = LFU) **Faire** **Si** Taille_Cache >= Capacite_Cache **Alors** Min <- Cache^.t Extraire élément avec t minimum **Sinon** Rien **Fin** Ajouter Adresse, Masque dans le cache **Sinon** Rien **FinSi**	Cache: in out; Adresse: in; Arbre: in out; Adresse, Masque in; Arbre: in out;
R3: Comment « Trouver le minimum des temps de (Arbre1, Arbre2) » ?	
Si (Arbre1 = Null) and (Arbre2 = Null) **Alors** Minimum <- Min(Arbre1^.t, Arbre2^.t) **SinonSi** Arbre1 = Null Alors Minimum <- Arbre2^.t	 Arbre: in;

Sinon Minimum <- Arbre2^.t **FinSi**	Adresse: in;
R3: Comment « Trouver la valeur du i-ème bit de Adresse » ?	
Retourne (Adresse & (2**(32 - i)))	Adresse: in;
R3: Comment « Extraire élément avec t minimum » ?	
Si Est_Feuille(Cache) and (Cache^.t = Min) **Alors** Liberer (Cache) **SinonSi** Cache^t = Min **Alors** **Si** Valeur du i-ème bit de Cache^.Adresse **Alors** Extraire élément avec t minimum dans … Cache^.Fils_Droit **Sinon** Extraire élément avec t minimum dans … Cache^.Fils_Gauche **FinSi** Supprimer le Noeud “Cache” si inutile **Sinon** Rien **FinSi**	 Min: in; Cache^.Fils_Droit : in out; Min: in; Cache^.Fils_Gauche : in out; Cache : in out

		Évaluation Étudiant (I/P/A/+)	**Justification / commentaire**
Forme	Respect de la syntaxe Ri : Comment "... une action complexe ..." ? des actions combinées avec des structures de contrôle Rj : ...	+	
	Verbe à l'infinitif pour les actions complexes	+	
	Nom ou équivalent pour expressions complexes	+	
	Tous les Ri sont écrits contre la marge et espacés	+	
	Les flots de données sont définis	A	
	Une seule décision ou répétition par raffinage	A	
	Pas trop d'actions dans un raffinage (moins de 6)	A	Plus que 6 actions lors de l'utilisation inévitable de structures type Selon par exemple
	Bonne présentation des structures de contrôle	+	
Fond	Le vocabulaire est précis	A	
	Le raffinage d'une action décrit complètement cette action	A	
	Le raffinage d'une action ne décrit que cette action	A	
	Les flots de données sont cohérents	A	
	Pas de structure de contrôle déguisée	+	
	Qualité des actions complexes	A	