



1 Code translatable

Un programme contenant du code translatable a été créé, en partant du principe qu'il serait chargé à l'adresse 0. Dans son code, le programme fait référence aux adresses suivantes : 50, 78, 150, 152, 154. Si le programme est chargé en mémoire en commençant à l'emplacement 250, comment doivent être ajustées ces adresses.

Adresse + Emplacement de départ : 300 328, 400, 402, 404.

Un programme contenant du code translatable a été créé, en partant du principe qu'il serait chargé à l'adresse 100. Dans son code, le programme fait référence aux adresses suivantes : 135, 160, 164, 220, 224. Si le programme est chargé en mémoire en commençant à l'emplacement 250, comment doivent être ajustées ces adresses.

Adresse + Emplacement de départ – adresse de départ : 536, 560, 564, 620, 624.

2 Multiprogrammation avec partitions fixes

Soit un système doté de 2^{24} octets de mémoire et de partitions fixes, toutes d'une taille de 65 536 octets.

- Quel est le nombre de bits nécessaires dans une entrée de la table de processus pour enregistrer la partition à laquelle a été allouée un processus ? $65536 = 2^{16}$ octets. $2^{24}/2^{16} = 2^8$ partitions.
- Combien de bits doit avoir le registre limite ? 16 bits

3 Multiprogrammation avec partitions variables

Soit un état de la zone mémoire

10 ko utilisé	10 Ko Libre	20 Ko Utilisé	30 Ko Libre	10 Ko utilisé	5 Ko libre	30 Ko utilisé	20 Ko libre	10 Ko utilisé	15 Ko libre	20 Ko utilisé	20 Ko libre
------------------	----------------	------------------	----------------	------------------	---------------	------------------	----------------	------------------	----------------	------------------	----------------

Les prochaines requêtes sont 20 Ko, 10 Ko, 5 Ko.

- Si le système utilisé l'allocation de la premier zone libre, à quelle adresse de départ vont être allouées les requêtes ?

10 ko utilisé	10 Ko Libre	40 Ko Utilisé	10 Ko Libre	10 Ko utilisé	5 Ko libre	30 Ko utilisé	20 Ko libre	10 Ko utilisé	15 Ko libre	20 Ko utilisé	20 Ko libre
------------------	----------------	--------------------------	------------------------	------------------	---------------	------------------	----------------	------------------	----------------	------------------	----------------

60 ko Utilisé		10 Ko Libre	10 Ko utilisé	5 Ko libre	30 Ko utilisé	20 Ko libre	10 Ko utilisé	15 Ko libre	20 Ko utilisé	20 Ko libre
--------------------------	--	----------------	------------------	---------------	------------------	----------------	------------------	----------------	------------------	----------------

65 ko Utilisé	5 Ko Libre	10 Ko utilisé	5 Ko libre	30 Ko utilisé	20 Ko libre	10 Ko utilisé	15 Ko libre	20 Ko utilisé	20 Ko libre
--------------------------	-----------------------	------------------	---------------	------------------	----------------	------------------	----------------	------------------	----------------

- o Si le système utilisé l'allocation du meilleur ajustement, à quelle adresse de départ vont être allouées les requêtes ?

10 ko utilisé	10 Ko Libre	20 Ko Utilisé	30 Ko Libre	10 Ko utilisé	5 Ko libre	60 Ko utilisé	15 Ko libre	20 Ko utilisé	20 Ko libre
------------------	----------------	------------------	----------------	------------------	---------------	--------------------------------	----------------	------------------	----------------

40 ko Utilisé	30 Ko Libre	10 Ko utilisé	5 Ko libre	60 Ko utilisé	15 Ko libre	20 Ko utilisé	20 Ko libre
--------------------------------	----------------	------------------	---------------	------------------	----------------	------------------	----------------

40 ko Utilisé	30 Ko Libre	75 Ko utilisé	15 Ko libre	20 Ko utilisé	20 Ko libre
------------------	----------------	--------------------------------	----------------	------------------	----------------

- o Si le système utilisé l'allocation du pire ajustement, à quelle adresse de départ vont être allouées les requêtes ?

10 ko utilisé	10 Ko Libre	40 Ko Utilisé	10 Ko Libre	10 Ko utilisé	5 Ko libre	30 Ko utilisé	20 Ko libre	10 Ko utilisé	15 Ko libre	20 Ko utilisé	20 Ko libre
------------------	----------------	--------------------------------	------------------------------	------------------	---------------	------------------	----------------	------------------	----------------	------------------	----------------

10 ko utilisé	10 Ko Libre	40 Ko Utilisé	10 Ko Libre	10 Ko utilisé	5 Ko libre	40 Ko utilisé	10 Ko libre	10 Ko utilisé	15 Ko libre	20 Ko utilisé	20 Ko libre
------------------	----------------	------------------	----------------	------------------	---------------	--------------------------------	------------------------------	------------------	----------------	------------------	----------------

10 ko utilisé	10 Ko Libre	40 Ko Utilisé	10 Ko Libre	10 Ko utilisé	5 Ko libre	40 Ko utilisé	10 Ko libre	10 Ko utilisé	15 Ko libre	25 Ko utilisé	15 Ko libre
------------------	----------------	------------------	----------------	------------------	---------------	------------------	----------------	------------------	----------------	--------------------------------	------------------------------

- o Si le système utilisé l'allocation du pire ajustement, à quelle adresse de départ vont être allouées les requêtes ?

10 ko utilisé	10 Ko Libre	40 Ko Utilisé	10 Ko Libre	10 Ko utilisé	5 Ko libre	30 Ko utilisé	20 Ko libre	10 Ko utilisé	15 Ko libre	20 Ko utilisé	20 Ko libre
------------------	----------------	--------------------------------	------------------------------	------------------	---------------	------------------	----------------	------------------	----------------	------------------	----------------

10 ko utilisé	10 Ko Libre	60 Ko Utilisé	5 Ko libre	30 Ko utilisé	20 Ko libre	10 Ko utilisé	15 Ko libre	20 Ko utilisé	20 Ko libre
------------------	----------------	--------------------------------	---------------	------------------	----------------	------------------	----------------	------------------	----------------

10 ko utilisé	10 Ko Libre	95 Ko Utilisé	20 Ko libre	10 Ko utilisé	15 Ko libre	20 Ko utilisé	20 Ko libre
------------------	----------------	--------------------------------	----------------	------------------	----------------	------------------	----------------

Soit un état de la zone mémoire

10 ko utilisé	30 Ko Libre	20 Ko Utilisé	15 Ko Libre	10 Ko utilisé	20 Ko libre	30 Ko utilisé	50 Ko libre	10 Ko utilisé	45 Ko libre	20 Ko utilisé	30 Ko libre
------------------	----------------	------------------	----------------	------------------	----------------	------------------	----------------	------------------	----------------	------------------	----------------

Les prochaines requêtes sont 10 Ko, 25 Ko, 20 Ko.

- o Soit un système utilisant l'allocation de la premier zone de libre, a quelle adresse de départ vont être alloués chacune des requêtes ? 10 Ko, 135 Ko, 20 Ko

- Soit un système utilisant l'allocation du meilleur ajustement, a quelle adresse de départ vont être alloués chacune des requêtes ? *60 Ko, 10 Ko, 85 Ko*
- Soit un système utilisant l'allocation du pire ajustement, a quelle adresse de départ vont être alloués chacune des requêtes ? *135 Ko, 195Ko, 145 Ko*
- Soit un système utilisant l'allocation de la zone de libre suivante, a quelle adresse de départ vont être alloués chacune des requêtes ? *10 Ko, 135 Ko, 160 Ko*

4 Système de zones siamoises (buddy system)

Soit un système doté de 1Mo de mémoire et qui utilise le système de zones siamoises. Dessinez un schéma qui illustre l'allocation après chaque événement.

- Processus A, requête 50 Ko.
- Processus B requête 150 Ko.
- Processus C, requête 60 Ko.
- Processus D, requête 60 Ko.
- Processus E, requête 60 Ko.
- Processus D, sortie.
- Processus C, sortie.
- Processus E, sortie.
- Processus A, sortie.
- Processus F, requête 125 Ko.
- Processus G, requête 150 Ko.
- Processus F, sortie.
- Processus G, sortie.
- Processus B, sortie.

1024					
A	64	128	256	512	
A	64	128	B	512	
A	C	128	B	512	
A	C	D	64	B	512
A	C	D	E	B	512
A	C	64	E	B	512
A	64	64	E	B	512
A	64	128	B	512	
256			B	512	
F	128	B		512	
F	128	B		G	256
256			B	G	256
256			B	512	
1024					

Soit un système doté de 1Mo de mémoire et qui utilise le système de zones siamoises. Dessinez un schéma qui illustre l'allocation après chaque événement.

- Processus A, requête 150 Ko.
- Processus B requête 40 Ko.
- Processus C, requête 80 Ko.
- Processus D, requête 210 Ko.
- Processus B, sortie.
- Processus E, requête 60 Ko.
- Processus C, sortie.
- Processus A, sortie.
- Processus E, sortie.
- Processus F, requête 115 Ko.
- Processus G, requête 150 Ko.
- Processus F, sortie.
- Processus G, sortie.
- Processus D, sortie.

1024					
A		256		512	
A	B	64	128	512	
A	B	64	C	512	
A	B	64	C	D	256
A	128		C	D	256
A	E	64	C	D	256
A	E	64	128	D	256
256		E	64	128	D
512				D	256
F	128	256		D	256
F	128	G		D	256
256		G		D	256
512				D	256
1024					

Soit un système doté de 1Mo de mémoire et qui utilise le système de zones siamoises. Quelle va être la première requête à échouer en raison d'un manque de mémoire.

Requete : 50 Ko, 150 Ko, 90 Ko, 130 Ko, 70 Ko, 80 Ko, 120 Ko, 180 Ko, 60 Ko.

1024				
X	64	128	256	512
X	64	128	X	512

X	64	128	X	512		
X	64	X	X	512		
X	64	X	X	X	256	
X	64	X	X	X	X	128
X	64	X	X	X	X	X

Requête 120 Ko

Soit un système doté de 1Mo de mémoire et qui utilise le système de zones siamoises. Quelle va être la première requête à échouer en raison d'un manque de mémoire.

Requete : 150 Ko, 70 Ko, 260 Ko, 40 Ko, 70 Ko, 180 Ko, 50 Ko, 120 Ko, 60 Ko.

1024					
X	256			512	
X	X	128		512	
X	X	128		X	
X	X	X	64	X	

5 Pagination simple

Sur un système de pagination simple de 2^{24} octets de mémoire physiques, 256 pages d'espace d'adressage logique et une taille de pages de 2^{10} octets.

Combien de bits se trouvent dans une adresse logique ? $2^8 + 2^{10} = 2^{18} \Rightarrow 18 \text{ bits}$

Combien d'octets se trouvent dans le cadre de page ? 2^{10}

Combien de bits physiques spécifient un cadre de page ? $2^{24} - 2^{10} = 2^{14} \Rightarrow 14 \text{ bits}$

Combien d'entrées se trouvent dans la table de page ? 2^8

Combien de bits sont nécessaires pour stocker une entrée de table de pages (avec 1 bit de validité) ? *cadre de page + bit de validité* $\Rightarrow 14 + 1 \Rightarrow 15$

Sur un système de pagination simple de 2^{32} octets de mémoire physiques, 2^{12} pages d'espace d'adressage logique et une taille de pages de 512 octets.

Combien de bits se trouvent dans une adresse logique ? $2^{12} + 2^9 = 2^{21} \Rightarrow 21 \text{ bits}$

Combien d'octets se trouvent dans le cadre de page ? 2^9

Combien de bits physiques spécifient un cadre de page ? $2^{32} - 2^9 = 2^{23} \Rightarrow 23 \text{ bits}$

Combien d'entrées se trouvent dans la table de page ? 2^{12}

Combien de bits sont nécessaires pour stocker une entrée de table de pages (avec 1 bit de validité) ? *cadre de page + bit de validité* $\Rightarrow 23 + 1 \Rightarrow 24$

Sur un système de pagination simple avec une table de pages contenant 64 entrées de 11bits (avec un bit de validité compris), et une taille de page de 512 octets.

- Combien de bits dans l'adresse logique spécifient le numéro de page ? $2^6 \Rightarrow 6 \text{ bits}$
- Combien de bits dans l'adresse logique spécifient l'offset à l'intérieur d'une page ? $2^9 \Rightarrow 9 \text{ bits}$

- Combien de bits se trouvent dans l'adresse logique ? $2^6+2^9 = 2^{15} \Rightarrow 15 \text{ bits}$
- Quelle est la taille de l'espace d'adressage logique ? $2^6+2^9 = 2^{15} = 32768 \text{ bits}$
- Combien de bits de l'adresse physique spécifient le numéro de cadre ? $11 \text{ bits de la table} - 1 \Rightarrow 10 \text{ bits}$
- Combien de bits de l'adresse physique spécifient l'offset au sein d'un cadre de page ? $2^9 \Rightarrow 9 \text{ bits}$
- Combien de bits se situe dans l'adresse physique ? $9+10 \Rightarrow 19 \text{ bits}$
- Quelle est la taille de l'espace d'adressage physique ? 2^{19} bits

Sur un système de pagination simple avec une table de pages contenant 512 entrées de 16bits (avec un bit de validité compris), et une taille de page de 1024 octets.

- Combien de bits dans l'adresse logique spécifient le numéro de page ? $2^9 \Rightarrow 9 \text{ bits}$
- Combien de bits dans l'adresse logique spécifient l'offset à l'intérieur d'une page ? $2^{10} \Rightarrow 10 \text{ bits}$
- Combien de bits se trouvent dans l'adresse logique ? $2^9+2^{10} = 2^{19} \Rightarrow 19 \text{ bits}$
- Quelle est la taille de l'espace d'adressage logique ? $2^9+2^{10} = 2^{15}$
- Combien de bits de l'adresse physique spécifient le numéro de cadre ? $16 \text{ bits de la table} - 1 \Rightarrow 15 \text{ bits}$
- Combien de bits de l'adresse physique spécifient l'offset au sein d'un cadre de page ? $2^{10} \Rightarrow 10 \text{ bits}$
- Combien de bits se situe dans l'adresse physique ? $15+10 \Rightarrow 25 \text{ bits}$
- Quelle est la taille de l'espace d'adressage physique ? 2^{25} bits

6 Pagination sur plusieurs niveaux

Un système utilisant une table de pages de deux niveaux possède des pages de 2^{12} octets et des adresses virtuelles de 32 bits. Les 8 premiers bits de l'adresse servent d'index dans la table de pages de premier niveau.

- Combien de bits spécifient l'index de second niveau ? $32-8-12 = 12 \text{ bits}$
- Combien y a-t-il d'entrées dans la table de pages du 1er niveau ? 2^8
- Combien y a-t-il d'entrées dans la table de pages du 2eme niveau ? 2^{12}
- Combien y a-t-il de pages dans l'espace d'adressage virtuelle ? $2^8+2^{12} = 2^{20}$

Un système utilisant une table de pages de deux niveaux possède des adresses virtuelles de 32 bits. Les 8 premiers bits de l'adresse servent d'index dans la table de pages de premier niveau.

Les 10 prochains bits de l'adresse servent d'index dans la table de pages de deuxième niveau.

- Combien y a-t-il d'octets dans une page ? $32-8-10 = 14 \Rightarrow 2^{14} \text{ octets}$
- Combien y a-t-il d'entrées dans la table de pages du 1er niveau ? 2^8
- Combien y a-t-il d'entrées dans la table de pages du 2eme niveau ? 2^{10}
- Combien y a-t-il de pages dans l'espace d'adressage virtuelle ? $2^8+2^{10} = 2^{18}$

Un ordinateur possède un espace d'adressage virtuel de 32 bits et des pages de 1024 octets. Une entrée de table prend 4 octets. Une table de pages de plusieurs niveaux est utilisée car chaque doit être contenue dans une page. Combien faut-il de niveau ?

$$1024/4 = 256 = 8 \text{ entrées / Page}$$

$$2^{32} - 2^{10} = 2^{22} \text{ pages} \Rightarrow 22 \text{ bits}$$

$$22/8 = 2,75 \Rightarrow 3 \text{ niveaux}$$

Un ordinateur possède un espace d'adressage virtuel de 64 bits et des pages de 2048 octets. Une entrée de table prend 4 octets. Une table de pages de plusieurs niveaux est utilisée car chaque doit être contenue dans une page. Combien faut-il de niveau ?

$$2048/4 = 512 = 9 \text{ entrées / Page}$$

$$2^{64} - 2^{11} = 2^{53} \text{ pages} \Rightarrow 53 \text{ bits}$$

$$53/9 = 5.8 \Rightarrow 6 \text{ niveaux}$$

7 Segmentation simple

Sur un système utilisant la segmentation simple, calculez l'adresse physique de chacune des adresses logiques. Indiquer si elle génère un défaut de segment.

330	124
902	211
111	99
498	302

- $0,99 \Rightarrow 99 < 124 \text{ ok} \Rightarrow 330 + 99 = 429$
- $2,78 \Rightarrow 78 < 99 \text{ ok} \Rightarrow 111 + 78 = 189$
- $1,265 \Rightarrow 265 > 211 \text{ nok}$
- $3,222 \Rightarrow 222 < 302 \text{ ok} \Rightarrow 498 + 222 = 720$
- $0,111 \Rightarrow 111 < 124 \text{ ok} \Rightarrow 330 + 111 = 441$

Sur un système utilisant la segmentation simple, calculez l'adresse physique de chacune des adresses logiques. Indiquer si elle génère un défaut de segment.

1100	500
2500	1000
200	600
4000	1200

- $0,300 \Rightarrow 300 < 500 \text{ ok} \Rightarrow 1100 + 300 = 1400$
- $2,800 \Rightarrow 800 > 600 \text{ nok}$
- $1,600 \Rightarrow 600 < 1000 \text{ ok} \Rightarrow 600 + 2500 = 3100$
- $3,1100 \Rightarrow 1100 < 1200 \text{ ok} \Rightarrow 1100 + 4000 = 5100$
- $1,1111 \Rightarrow 1111 > 1000 \text{ nok}$

8 Segmentation et pagination

Sur un système utilisant la pagination et la segmentation, l'espace d'adressage virtuel se compose au maximum de 8 segment, chacun d'entre eux pouvant avoir une longueur qui atteint 2^{29} . Le matériel pagine chaque segment en pages de 256 octets. Combien de bits de l'adresse virtuelle spécifient :

- Le numéro de segment ? $2^3 = 8 \Rightarrow 3 \text{ bits}$

- Le numéro de pages ? $2^{29}-2^8 = 2^{21} \Rightarrow 21 \text{ bits}$
- L'offset au sein de la page ? $2^8 \Rightarrow 8 \text{ bits}$
- L'adresse virtuelle entière ? $3+21+8 \Rightarrow 32 \text{ bits}$

Sur un système utilisant la pagination et la segmentation, l'espace d'adressage virtuel se compose au maximum de 16 segment, chacun d'entre eux pouvant avoir une longueur qui atteint 2^{16} . Le matériel pagine chaque segment en pages de 512 octets. Combien de bits de l'adresse virtuelle spécifient :

- Le numéro de segment ? $2^4=16 \Rightarrow 4 \text{ bits}$
- Le numéro de pages ? $2^{16}-2^9 = 2^7 \Rightarrow 7 \text{ bits}$
- L'offset au sein de la page ? $2^9 \Rightarrow 9 \text{ bits}$
- L'adresse virtuelle entière ? $4+7+9 \Rightarrow 20 \text{ bits}$

9 Algorithmes de remplacement

Soit un système avec quatre cadres de pages.

Page	Temps de chargement	Dernière référence	Bit de modification	Bit de référence
0	167	374	1	1
1	321	321	0	0
2	254	306	1	0
3	154	331	0	1

Quelle page va remplacer l'algorithme FIFO ? la première $\Rightarrow 3$

Quelle page va remplacer l'algorithme LRU ? la dernière utilisé $\Rightarrow 2$

Quelle page va remplacer l'algorithme NRU ? Bit de référence à 0 puis bit de mod à 0 $\Rightarrow 1$

Quelle page va remplacer l'algorithme de la seconde chance ? la plus ancienne avec bit de référence à 0 $\Rightarrow 2$

Soit un système avec quatre cadres de pages.

Page	Temps de chargement	Dernière référence	Bit de modification	Bit de référence
0	227	327	1	0
1	345	367	1	1
2	101	331	1	1
3	234	382	0	1

Quelle page va remplacer l'algorithme FIFO ? la première $\Rightarrow 2$

Quelle page va remplacer l'algorithme LRU ? la dernière utilisé $\Rightarrow 0$

Quelle page va remplacer l'algorithme NRU ? Bit de référence à 0 puis bit de mod à 0 $\Rightarrow 0$

Quelle page va remplacer l'algorithme de la seconde chance ? la plus ancienne avec bit de référence à 0 $\Rightarrow 0$

Soit les références aux pages suivantes : 0,9,0,1,8,1,8,7,8,7,1,2,8,2,7,8,2,3,8,3

Combien de défaut de page vont se produire si le programme possède 3 cadres de pages avec :

- FIFO

Avant	Page	Défaut
-,-,-	0	*
0,-,-	9	*
9,0,-	0	
9,0,-	1	*
1,9,0	8	*
8,1,9	1	
8,1,9	8	
8,1,9	7	*
7,8,1	8	
7,8,1	7	
7,8,1	1	
7,8,1	2	*
2,7,8	8	
2,7,8	2	
2,7,8	7	
2,7,8	8	
2,7,8	2	
2,7,8	3	*
3,2,7	8	*
8,3,2	3	

8 défauts

- LRU

Avant	Page	Défaut
-,-,-	0	*
0,-,-	9	*
9,0,-	0	
0,9,-	1	*
1,0,9	8	*
8,1,0	1	
1,8,0	8	
8,1,0	7	*
7,8,1	8	
8,7,1	7	
7,8,1	1	

1,7,8	2	*
2,1,7	8	*
8,2,1	2	
2,8,1	7	*
7,2,8	8	
8,7,2	2	
2,8,7	3	*
3,2,8	8	
8,3,2	3	

9 défauts

- Optimal

Avant	Page	Défaut
-,-,-	0	*
0,-,-	9	*
9,0,-	0	
9,0,-	1	*
1,9,0	8	*
8,1,9	1	
8,1,9	8	
8,1,9	7	*
7,8,1	8	
7,8,1	7	
7,8,1	1	
7,8,1	2	*
2,7,8	8	
2,7,8	2	
2,7,8	7	
2,7,8	8	
2,7,8	2	
2,7,8	3	*
3,2,8	8	
3,2,8	3	

7 défauts

Soit les références aux pages suivantes : 0,1,4,2,0,2,6,5,1,2,3,2,1,2,6,2,1,3,6,2

Combien de défaut de page vont se produire si le programme possède 3 cadres de pages avec :

- FIFO

Avant	Page	Défaut
-,-,-	0	*

0,-,-	1	*
1,0,-	4	*
4,1,0	2	*
2,4,1	0	*
0,2,4	2	
0,2,4	6	*
6,0,2	5	*
5,6,0	1	*
1,5,6	2	*
2,1,5	3	*
3,2,1	2	
3,2,1	1	
3,2,1	2	
3,2,1	6	*
6,3,2	2	
6,3,2	1	*
1,6,3	3	
1,6,3	6	
1,6,3	2	*

8 défauts

- LRU

Avant	Page	Défaut
-,-,-	0	*
0,-,-	1	*
1,0,-	4	*
4,1,0	2	*
2,4,1	0	*
0,2,4	2	
2,0,4	6	*
6,2,0	5	*
5,6,2	1	*
1,5,6	2	*
2,1,5	3	*
3,2,1	2	
2,3,1	1	
1,2,3	2	
2,1,3	6	*
6,2,1	2	

2,6,1	1	
1,2,6	3	*
3,1,2	6	*
6,3,1	2	*

14 défauts

- Optimal

Avant	Page	Défaut
-, -, -	0	*
0, -, -	1	*
1, 0, -	4	*
4, 1, 0	2	*
2, 1, 0	0	
2, 1, 0	2	
2, 1, 0	6	*
6, 2, 1	5	*
5, 2, 1	1	
5, 2, 1	2	
5, 2, 1	3	*
3, 2, 1	2	
3, 2, 1	1	
3, 2, 1	2	
3, 2, 1	6	*
6, 2, 1	2	
6, 2, 1	1	
6, 2, 1	3	*
3, 6, 2	6	
3, 6, 2	2	

9 défauts

10 Mise en pratique en C

Ecrivez la fonction `Trans`, qui prend une adresse virtuelle segmentée et qui la translate en une adresse physique, en renvoyant l'adresse physique comme un `int`. La fonction, que l'on suppose déjà écrite, gère les fautes de segments. La variable « `segTable` » contient la table de segment.

`Struct SegTableType`

```
{
    int loc ; /* adresse du segment de départ */
    int len ; /* longueur du segment en octets */
};
```

`Struct VirtualAdressType`

```
{
    int seg ; /* portion d'adresse spécifiant le segment */
```

```
        Int off ; /* portion d'adresse spécifiant l'offset */
};
Struct SegTableType segTable[NUMBER_SEGMENTS];
Void Fault();
Int Trans(struct VirtualAdressType virtAddr)
{
    Int physAddr = -1 ;
    If (virtAddr.off >= segTable[virtAddr.seg].len)
        Fault();
    Else
        physAddr = segTable[virtAddr.seg].loc + virtAddr.off;
    return physAddr;
}
```