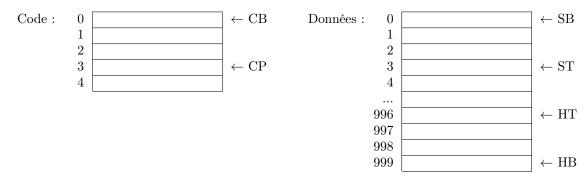
La machine TAM

Machine à pile. Pas de registre de donnée.



Instructions :

```
PUSH n
                    ST = ST + n
                    a = ST - d; ST = ST - d - n;
POP (d) n
                    Pour i de d à 0 Donnees(ST++) = Donnees[a++] fin pour
LOADL n
                    Donnees(ST) = n; ST = ST+1
LOADA d[r]
                    ST = d[r]; ST = ST + 1;
LOAD (n) d[r]
                    Pour i de 0 a n-1
                      Donnees(ST+i) = Donnees(val(r)+d+i)
                    fin pour;
                    ST = ST+n
LOADI (n)
                    Empile n mots lus à l'adresse précedemment empilée
STORE (n) d[r]
                    Pour i de 0 a n-1
                      Donnees(val(r)+d+i) = Donnees(ST+i-n);
                    fin pour;
                    ST = ST-n
STOREI (n)
                    Ecrit les n mots empilés, à l'adresse empilée
JUMP etiq
                    CP = val(etiq)
JUMP d[r]
                    CP = val(r) + d
JUMPI
                    ST = ST - 1; CP = data[ST]
                    si Donnees(ST -1) = n alors CP = val(etiq) fin si;
JUMPIF (n) etiq
                    ST = ST - 1
JUMPIF (n) d[r]
                    si Donnees(ST -1) = n alors CP = val(r) + d fin si;
                    ST = ST - 1
                    Appel de op (fonction pré-définie - cf verso)
SUBR op
                    consommation des arguments laissés en sommet de pile
CALL (r) op
                    Appel de op (fonction utilisateur)
                    Donnees(ST) = r
                    Donnees(ST+1) = LB
                                           | Enregistrement d'activation
                    Donnees(ST+2) = CP+1
                    LB = ST
                    ST = ST + 3
                    CP = op
CALLI
                    Appel de la fonction dont l'adresse est en sommet de pile
                    CP = Donnees(LB + 2)
RETURN (r) p
                    LB = Donnees(LB + 1)
                    Pour i de 0 a p-1
                      Donnees(LB-p+r-1) = Donnees(ST-i);
                    fin pour;
                    ST = LB-p+r -- dépile p paramètres et garde r résultats
HALT
                    Arret
```

Primitives prédéfinies de la machine TAM

BNeg	Nom	Paramètres	Résultat					
BOr	Fonctions Booléens							
BAnd 2	BNeg	1	1	Négation logique				
BOut 1	BOr		1	Ou logique				
BIn 0 1 Lit sur stdin un booléen (1 ou 0) B2C 1 1 Conversion vers un caractère (true = '1', false = '0') B2I 1 1 Conversion vers un entier (true = 1, false = 0) B2S 1 1 Conversion vers un entier (true = 1, false = 0) B2S 1 1 Conversion vers un caractère (empile l'adresse de la chaîne dans le tableau des chaînes) Fonctions Caractères COut 1 Lit sur stdin un caractère C2B 1 1 Conversion vers un booléen ('0' = false, sinon true). C2I 1 1 Conversion vers un entier (le code ASCII) C2S 1 1 Conversion vers un entien (le code ASCII) C2S 1 1 Conversion vers un entiène (le code ASCII) C2S 1 1 Négation entière INeq 1 1 Addition entière ISub 2 1 Addition entière ISub 2 1 Multiplication entière IEq 2 1	BAnd	2	1	Et logique				
B2C	BOut	1	0	Affiche sur stdout un booléen (true ou false)				
B2I	BIn	0	1	Lit sur stdin un booléen (1 ou 0)				
Conversion vers une chaîne de caractères (empile l'adresse de la chaîne dans le tableau des chaînes) Fonctions Caractères	B2C	1	1					
	B2I	1	1	Conversion vers un entier (true = 1, false = 0)				
COut	B2S	1	1	Conversion vers une chaîne de caractères (empile l'adresse de				
COut CIn 1 0 Affiche sur stdout un caractère CIn 0 1 Lit sur stdin un caractère C2B 1 1 Conversion vers un booléen ('0' = false, sinon true). C2I 1 1 Conversion vers un entier (le code ASCII) C2S 1 1 Conversion vers un entier (le code ASCII) C2S 1 1 Conversion vers un entière (le code ASCII) C2S 1 1 Conversion vers un entière (le code ASCII) C2S 1 1 Négation entière IAdd 2 1 Addition entière IAdd 2 1 Addition entière ISub 2 1 Multiplication entière IMul 2 1 Multiplication entière IMul 2 1 Diviseur dans division entière IMul 2 1 Reste dans division entière IMul 2 1 Test égalité entre 2 entières ILeq 2 1 Test égalité entre 2 entières				la chaîne dans le tableau des chaînes)				
CIn 0 1 Conversion vers un booléen ('0' = false, sinon true). C2I 1 1 1 Conversion vers un booléen ('0' = false, sinon true). C2S 1 Conversion vers un entier (le code ASCII) C2S 1 Conversion vers un etaîne de caractères (empile l'adresse de la chaîne dans le tableau des chaînes) Fonctions Entiers INeg 1 Addition entière IAdd 2 1 Addition entière ISub 2 1 Soustraction entière IMul 2 1 Multiplication entière IMul 2 1 Diviseur dans division entière IMod 2 1 Reste dans division entière IEq 2 1 Test égalité entre 2 entiers INeq 2 1 Test différence entre 2 entiers ILss 2 1 Test inférieur strictement entre 2 entiers ILeq 2 1 Test inférieur ou égal entre 2 entiers IGtr 2 1 Test supérieur ou égal entre 2 entiers IGtr 2 1 Test supérieur ou égal entre 2 entiers IGut 1 0 Affiche sur stdout un entier IIn 0 1 Lit sur stdoin un entier IZB 1 1 Conversion vers un booléen (0 = false, sinon true,) IZS 1 1 Conversion vers un caractère (le code ASCII). Non implémentée. IZS 1 1 Conversion vers un chaîne de caractères (empile l'adresse de	Fonctions Caractères							
C2B 1 1 Conversion vers un booléen ('0' = false, sinon true). C2I 1 1 Conversion vers un entier (le code ASCII) C2S 1 1 Conversion vers une chaîne de caractères (empile l'adresse de la chaîne dans le tableau des chaînes) Fonctions Entiers INeg 1 1 Négation entière IAdd 2 1 Addition entière ISub 2 1 Multiplication entière IMul 2 1 Multiplication entière IDiv 2 1 Diviseur dans division entière IMul 2 1 Diviseur dans division entière IMul 2 1 Reste dans division entière IMul 2 1 Reste dans division entière IEq 2 1 Test égalité entre 2 entières IEq 2 1 Test égalité entre 2 entières ILs 2 1 Test inférieur strictement entre 2 entières IGet 2 1 Test supérieur ou égal entre 2 entières	COut	1	0	Affiche sur stdout un caractère				
C2I 1 1 Conversion vers un entier (le code ASCII) C2S 1 1 Conversion vers une chaîne de caractères (empile l'adresse de la chaîne dans le tableau des chaînes) Fonctions Entiers INeg 1 1 Négation entière IAdd 2 1 Addition entière ISub 2 1 Multiplication entière IMul 2 1 Multiplication entière III Div 2 1 Diviseur dans division entière IMM 2 1 Reste dans division entière IMM 2 1 Reste dans division entière IMM 2 1 Test égalité entre 2 entière IMM 2 1 Test égalité entre 2 entière IMM 2 1 Test égalité entre 2 entières ILEQ 2 1 Test inférieur strictement entre 2 entières ILEQ 1 Test supérieur strictement entre 2 entières IGeq 2 1 Test supérieur ou égal entre 2 entières IOUt	CIn	0	1	Lit sur stdin un caractère				
C2S	C2B	1	1	Conversion vers un booléen ('0' = false, sinon true).				
Fonctions Entiers INeg	C2I	1	1	Conversion vers un entier (le code ASCII)				
INeg	C2S	1	1	Conversion vers une chaîne de caractères (empile l'adresse de				
INeg1Négation entièreIAdd21Addition entièreISub21Soustraction entièreIMul21Multiplication entièreIDiv21Diviseur dans division entièreIMod21Reste dans division entièreIEq21Test égalité entre 2 entiersINeq21Test différence entre 2 entiersILss21Test inférieur strictement entre 2 entiersILeq21Test inférieur ou égal entre 2 entiersIGtr21Test supérieur strictement entre 2 entiersIGeq21Test supérieur ou égal entre 2 entiersIOut10Affiche sur stdout un entierIIn01Lit sur stdin un entierI2B11Conversion vers un booléen (0 = false, sinon true,)I2C11Conversion vers un caractère (le code ASCII). Non implémentée.I2S11Conversion vers une chaîne de caractères (empile l'adresse de				la chaîne dans le tableau des chaînes)				
IAdd21Addition entièreISub21Soustraction entièreIMul21Multiplication entièreIDiv21Diviseur dans division entièreIMod21Reste dans division entièreIEq21Test égalité entre 2 entiersINeq21Test différence entre 2 entiersILss21Test inférieur strictement entre 2 entiersILeq21Test supérieur ou égal entre 2 entiersIGtr21Test supérieur ou égal entre 2 entiersIGeq21Test supérieur ou égal entre 2 entiersIOut10Affiche sur stdout un entierIIn01Lit sur stdin un entierI2B11Conversion vers un booléen (0 = false, sinon true,)I2C11Conversion vers un caractère (le code ASCII). Non implémentée.I2S11Conversion vers une chaîne de caractères (empile l'adresse de	Fonctio	Fonctions Entiers						
ISub21Soustraction entièreIMul21Multiplication entièreIDiv21Diviseur dans division entièreIMod21Reste dans division entièreIEq21Test égalité entre 2 entiersINeq21Test différence entre 2 entiersILss21Test inférieur strictement entre 2 entiersILeq21Test inférieur ou égal entre 2 entiersIGtr21Test supérieur strictement entre 2 entiersIGeq21Test supérieur ou égal entre 2 entiersIOut10Affiche sur stdout un entierIIn01Lit sur stdin un entierI2B11Conversion vers un booléen (0 = false, sinon true,)I2C11Conversion vers un caractère (le code ASCII).Non implémentée.Non implémentée.I2S11Conversion vers une chaîne de caractères (empile l'adresse de	INeg	1	1	Négation entière				
IMul21Multiplication entièreIDiv21Diviseur dans division entièreIMod21Reste dans division entièreIEq21Test égalité entre 2 entiersINeq21Test différence entre 2 entiersILss21Test inférieur strictement entre 2 entiersILeq21Test inférieur ou égal entre 2 entiersIGtr21Test supérieur strictement entre 2 entiersIGeq21Test supérieur ou égal entre 2 entiersIOut10Affiche sur stdout un entierIDut1Lit sur stdin un entierI2B11Conversion vers un booléen (0 = false, sinon true,)I2C11Conversion vers un caractère (le code ASCII). Non implémentée.I2S11Conversion vers une chaîne de caractères (empile l'adresse de	IAdd	2	1	Addition entière				
IDiv 2 1 Diviseur dans division entière IMod 2 1 Reste dans division entière IEq 2 1 Test égalité entre 2 entiers INeq 2 1 Test différence entre 2 entiers ILss 2 1 Test inférieur strictement entre 2 entiers ILeq 2 1 Test inférieur ou égal entre 2 entiers IGtr 2 1 Test supérieur strictement entre 2 entiers IGtr 2 1 Test supérieur strictement entre 2 entiers IGeq 2 1 Test supérieur ou égal entre 2 entiers IOut 1 0 Affiche sur stdout un entier IIn 0 1 Lit sur stdin un entier I2B 1 1 Conversion vers un booléen (0 = false, sinon true,) I2C 1 Conversion vers un caractère (le code ASCII). Non implémentée. I2S 1 1 Conversion vers une chaîne de caractères (empile l'adresse de	ISub	2	1	Soustraction entière				
IMod21Reste dans division entièreIEq21Test égalité entre 2 entiersINeq21Test différence entre 2 entiersILss21Test inférieur strictement entre 2 entiersILeq21Test inférieur ou égal entre 2 entiersIGtr21Test supérieur strictement entre 2 entiersIGeq21Test supérieur ou égal entre 2 entiersIOut10Affiche sur stdout un entierIIn01Lit sur stdin un entierI2B11Conversion vers un booléen (0 = false, sinon true,)I2C11Conversion vers un caractère (le code ASCII).Non implémentée.I2S11Conversion vers une chaîne de caractères (empile l'adresse de	IMul	2	1	Multiplication entière				
IEq21Test égalité entre 2 entiersINeq21Test différence entre 2 entiersILss21Test inférieur strictement entre 2 entiersILeq21Test inférieur ou égal entre 2 entiersIGtr21Test supérieur strictement entre 2 entiersIGeq21Test supérieur ou égal entre 2 entiersIOut10Affiche sur stdout un entierIIn01Lit sur stdin un entierI2B11Conversion vers un booléen (0 = false, sinon true,)I2C11Conversion vers un caractère (le code ASCII).Non implémentée.I2S11Conversion vers une chaîne de caractères (empile l'adresse de	IDiv	2	1	Diviseur dans division entière				
INeq 2 1 Test différence entre 2 entiers ILss 2 1 Test inférieur strictement entre 2 entiers ILeq 2 1 Test inférieur ou égal entre 2 entiers IGtr 2 1 Test supérieur strictement entre 2 entiers IGeq 2 1 Test supérieur ou égal entre 2 entiers IOut 1 0 Affiche sur stdout un entier IIn 0 1 Lit sur stdin un entier I2B 1 1 Conversion vers un booléen (0 = false, sinon true,) I2C 1 Conversion vers un caractère (le code ASCII). Non implémentée. I2S 1 Conversion vers une chaîne de caractères (empile l'adresse de	IMod	2	1	Reste dans division entière				
ILss 2 1 Test inférieur strictement entre 2 entiers ILeq 2 1 Test inférieur ou égal entre 2 entiers IGtr 2 1 Test supérieur strictement entre 2 entiers IGeq 2 1 Test supérieur ou égal entre 2 entiers IOut 1 0 Affiche sur stdout un entier IIn 0 1 Lit sur stdin un entier I2B 1 1 Conversion vers un booléen (0 = false, sinon true,) I2C 1 1 Conversion vers un caractère (le code ASCII). Non implémentée. I2S 1 1 Conversion vers une chaîne de caractères (empile l'adresse de	IEq	2	1	Test égalité entre 2 entiers				
ILeq21Test inférieur ou égal entre 2 entiersIGtr21Test supérieur strictement entre 2 entiersIGeq21Test supérieur ou égal entre 2 entiersIOut10Affiche sur stdout un entierIIn01Lit sur stdin un entierI2B11Conversion vers un booléen (0 = false, sinon true,)I2C11Conversion vers un caractère (le code ASCII).Non implémentée.I2S11Conversion vers une chaîne de caractères (empile l'adresse de	INeq	2	1	Test différence entre 2 entiers				
IGtr 2 1 Test supérieur strictement entre 2 entiers IGeq 2 1 Test supérieur ou égal entre 2 entiers IOut 1 0 Affiche sur stdout un entier IIn 0 1 Lit sur stdin un entier I2B 1 1 Conversion vers un booléen (0 = false, sinon true,) I2C 1 1 Conversion vers un caractère (le code ASCII). Non implémentée. I2S 1 1 Conversion vers une chaîne de caractères (empile l'adresse de	ILss	2	1	Test inférieur strictement entre 2 entiers				
IGeq 2 1 Test supérieur ou égal entre 2 entiers IOut 1 0 Affiche sur stdout un entier IIn 0 1 Lit sur stdin un entier I2B 1 1 Conversion vers un booléen (0 = false, sinon true,) I2C 1 1 Conversion vers un caractère (le code ASCII). Non implémentée. I2S 1 1 Conversion vers une chaîne de caractères (empile l'adresse de	ILeq	2	1	Test inférieur ou égal entre 2 entiers				
IOut 1 0 Affiche sur stdout un entier IIn 0 1 Lit sur stdin un entier I2B 1 1 Conversion vers un booléen (0 = false, sinon true,) I2C 1 1 Conversion vers un caractère (le code ASCII). Non implémentée. Non implémentée de caractères (empile l'adresse de	IGtr	2	1	Test supérieur strictement entre 2 entiers				
IIn 0 1 Lit sur stdin un entier I2B 1 1 Conversion vers un booléen (0 = false, sinon true,) I2C 1 1 Conversion vers un caractère (le code ASCII). Non implémentée. I2S 1 1 Conversion vers une chaîne de caractères (empile l'adresse de	IGeq	2	1	Test supérieur ou égal entre 2 entiers				
I2B 1 1 Conversion vers un booléen (0 = false, sinon true,) I2C 1 1 Conversion vers un caractère (le code ASCII). Non implémentée. Non implémentée. I2S 1 1 Conversion vers une chaîne de caractères (empile l'adresse de	IOut	1	0	Affiche sur stdout un entier				
I2C 1 Conversion vers un caractère (le code ASCII). Non implémentée. Non implémentée. I2S 1 Conversion vers une chaîne de caractères (empile l'adresse de	IIn	0	1	Lit sur stdin un entier				
I2C 1 Conversion vers un caractère (le code ASCII). Non implémentée. Non implémentée. I2S 1 Conversion vers une chaîne de caractères (empile l'adresse de	I2B	1	1	Conversion vers un booléen $(0 = false, sinon true,)$				
I2S 1 Non implémentée. Conversion vers une chaîne de caractères (empile l'adresse de	I2C	1	1					
\ \ \				Non implémentée.				
	I2S	1	1	Conversion vers une chaîne de caractères (empile l'adresse de				

Nom	Paramètres	Résultat				
Fonctions Mémoires						
MVoid	0	1	Renvoie la valeur « adresse non initialisée »			
MAlloc	1	1	Alloue un bloc mémoire et renvoie son adresse			
MFree	1	0	Libère un bloc mémoire			
			Non implémentée.			
MCompare	2	1	Test égalité entre le contenu de 2 blocs mémoire dont les			
			adresses sont en sommet de pile			
MCopy	3	0	Copie le contenu d'un bloc mémoire dans le second bloc			
			mémoire. Dans l'ordre d'empilage : taille, adr destina-			
			tion, adr source			
Fonctions Chaînes de caractères						
SAlloc	1	1	Crée une chaîne vide et empile son adresse dans le ta-			
			bleau des chaînes.			
SFree	1	0	Libère l'espace de la chaîne			
SCopy	1	1	Duplique la chaîne et empile l'adresse de la chaîne du-			
			pliquée			
SConcat	2	1	Concatène deux chaînes (résultat dans le premier argu-			
			ment - gardé en sommet de pile)			
SOut	1	0	Affiche sur stdout une chaîne			
SIn	0	1	Lit sur stdin une chaine			
S2B	1	1	Conversion vers un booléen ("false", "f" ou "0" = false,			
			sinon true,)			
S2C	1	1	Conversion vers un caractère (le code ASCII du première			
			caractère de la chaîne)			
S2I	1	1	Conversion vers un entier (ne fait rien si la chaîne n'est			
			pas un entier)			