Les programmes en langage d'assemblage nécessaires pour exécuter quelques structures algorithmiques classiques

<table>
<tr><th>La structure algorithmique</th><th>en langage d'assemblage</th></tr>
<tr><td><u>Si Alors Sinon ...</u>

Si (AX=1) alors
début
BX← 10;
DX←30 ;
Fin
Sinon
début
BX ← 0;
CX ←20;
Fin</td><td>Si: CMP AX,0001
JNZ Sinon
Alors: MOV BX,0010
MOV DX,0030
JMP FinSi
Sinon: XOR BX,BX ; ou bien MOV BX,0000
MOV CX,0020
FinSi:</td></tr>
<tr><td><u>La boucle Tant que…</u>

BX ← 5 ;
Tant que BX > 0 faire
début
BX ← BX-1 ;
AX ← 33 ;
Fin</td><td>MOV BX,0005
Tant que : CMP BX,0000
JBE FinTantque
DEC BX
MOV AX, 0033
JMP Tant que
FinTantque:</td></tr>
<tr><td rowspan="2"><u>La boucle pour</u> ...

AX←F9;
bx ← 0 ;
Pour K = 0 jusqu'à 10
bx ←bx + K ;</td><td>MOV AX,F9
XOR BX, BX
XOR CX,CX; ou bien MOV CX,00
For: CMP CX,0A
JA FinPour
ADD BX, CX
INC CX
JMP For
FinPour:</td></tr>
<tr><td>MOV AX,F9
XOR BX, BX
MOV CX, 0Ah
Repeat1: ADD BX, CX
LOOP Repeat1
EndRepeat1:</td></tr>
<tr><td><u>La boucle Répéter……</u>

BX← 10 ;
AX←0 ;
Répéter
début
AX←AX+BX ;
BX←BX – 1 ;
Fin
jusqu'à BX= 3</td><td>MOV BX,0010
XOR AX,AX; ou bien MOV AX,0000
Repeat2: ADD AX,BX
DEC BX
CMP BX,0003
JG Repeat2
FinRepeat2 :</td></tr>
<tr><td><u>Le test à plusieurs alternatives...</u>

Switch(BX)
début
case 1: AX ←1 ; break;
case 2: ax ← 5 ; break ;
case 3: ax ← 10; break;
default: ax ← 0;
fin</td><td>CMP BX,0001
JNZ case2
MOV AX,0001
JMP endswitch
case2: CMP BX,0002
JNZ case3
MOV AX,0005
JMP endswitch
case 3: CMP BX, 0003
JNZ default
MOV AX,000A
JMP endswitch
default: XOR AX,AX
endswitch:</td></tr>
</table>

Notez bien : Il n'y a pas de notion de variables mais des registres.
Le programme en langage d'assemblage n'est pas structuré.