Instrucțiuni pentru șiruri. Probleme „complexe” pe șiruri

* Au operanzi impliciti si care sunt folosite astfel:
  + Prelucreaza elementul curent al sirului
  + Trece la elementul urmator
* Initializari:
  + Se copiaza offset-ul sirului sursa in ESI
  + Se copiaza offset-ul sirului destinatie in EDI
  + Se reseteaza/seteaza DF (CLD => DF=0 si cu STD => DF=1) => se stabileste directia de parcurgere S->D (DF=0) sau D->S (DF=1)
* Clasificare
  + Instructiuni pentru transferul datelor
    - Din memorie in registrii (LODS – load string of ...)

LODSB

- AL<-[DS:ESI]

- daca DF=0 atunci ESI=ESI+1 altfel ESI=ESI-1

LODSW

- AX<-[DS:ESI]

- daca DF=0 atunci ESI=ESI+2 altfel ESI=ESI-2

LODSD

- EAX<-[DS:ESI]

- daca DF=0 atunci ESI=ESI+4 altfel ESI=ESI-4

* + - Din registrii in memorie (STOS – store string of ...)

STOSB

- [ES:EDI] <- AL

- daca DF=0 atunci EDI=EDI+1 altfel EDI=EDI-1

STOSW

- [ES:EDI] <- AX

- daca DF=0 atunci EDI=EDI+2 altfel EDI=EDI-2

STOSD

- [ES:EDI] <- EAX

- daca DF=0 atunci EDI=EDI+4 altfel EDI=EDI-4

* + - Din memorie in memorie (MOVS – move string of ...)

MOVSB

- [ES:EDI] <- byte [DS:ESI]

- daca DF=0 atunci ESI=ESI+1, EDI=EDI+1 altfel ESI=ESI-1, EDI=EDI-1

MOVSW

- [ES:EDI] <- word [DS:ESI]

- daca DF=0 atunci ESI=ESI+2, EDI=EDI+2 altfel ESI=ESI-2, EDI=EDI-2

MOVSD

- [ES:EDI] <- dword[DS:ESI]

- daca DF=0 atunci ESI=ESI+4, EDI=EDI+4 altfel ESI=ESI-4, EDI=EDI-4

* + Instructiuni pentru compararea datelor
    - Se compara date din memorie cu date din registrii (SCAS – scan string of ...)

SCASB

- CMP AL,[ES:EDI]

- daca DF=0 atunci EDI=EDI+1 altfel EDI=EDI-1

SCASW

- CMP AX,[ES:EDI]

- daca DF=0 atunci EDI=EDI+2 altfel EDI=EDI-2

SCASD

- CMP EAX,[ES:EDI]

- daca DF=0 atunci EDI=EDI+4 altfel EDI=EDI-4

* + - Se compara date din memorie cu date din memorie (CMPS – compare string of)

CMPSB

- CMP byte [DS:ESI], byte [ES:EDI]

- daca DF=0 atunci EDI=EDI+1, ESI=ESI+1 altfel EDI=EDI-1, ESI=ESI-1

CMPSW

- CMP word [DS:ESI], word [ES:EDI]

- daca DF=0 atunci EDI=EDI+2, ESI=ESI+2 altfel EDI=EDI-2, ESI=ESI-2

CMPSD

- CMP dword [DS:ESI], dword [ES:EDI]

- daca DF=0 atunci EDI=EDI+4, ESI=ESI+4 altfel EDI=EDI-4, ESI=ESI-4

Exercitii

1. Se da un sir de octeti care contine litere mici. Sa se obtina sirul literelor mari coresp. Literelor mici

|  |  |
| --- | --- |
| Data segment | Code segment |
| S db ‘a’,’b’, ‘c’,’d’  len equ $-S  D times len db 0 | MOV ECX,len  CLD  MOV ESI,S  MOV EDI,D  JECXZ Final  Repeta:  LODSB  SUB AL, ‘a’-’A’  STOSB  LOOP Repeta  Final: |

‘a’ - (‘a’-’A’)

‘a’ + (‘A’-’a’)

1. Se da un sir de octeti. Sa se obtina sirul oglindit

A db 1,2,3,4,5 => B db 5,4,3,2,1

|  |  |
| --- | --- |
| Data segment | Code segment |
| Sir db 1,2,3,4  Len equ $-sir  Ogl times len db 0 | MOV ECX, len  JECXZ final  MOV ESI, sir+len-1  MOV EDI, ogl  Repeta:  STD  LODSB  CLD  STOSB  Loop repeta  Final: |
|  |  |

1. Se dau 2 siruri de cuvinte. Se cere sa se concateneze sirul octetilor inferiori ai cuvintelor primului sir cu sirul octetilor superiori ai cuvintelor din cel de-al doilea sir. Sa se sorteze crescator sirul rezultat in interpretarea cu semn

S1 dw 2345h, 0a5h, 368h, 3990h

S2 dw 4h, 2655h, 10

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 45h | 23h | a5h | 00h | 68h | 03h | 90h | 39h | 4h | 00h |
| 55h | 26h | 0ah | 00h |  |  |  |  |  |  |

|  |  |
| --- | --- |
| Data segment | Code segment |
| S1 dw 2345h, 0a5h, 368h, 3990h  L1 equ ($-s1)/2  S2 dw 4h, 2655h, 10  L2 equ ($-s2)/2  S3 times L1 + L2 db 0 | MOV ECX, L1  JECXZ final1  MOV ESI, S1  MOV EDI, S3  Repeta:  LODSB  STOSB  LODSB  Loop Repeta  final1:  MOV ESI, S2  MOV ECX, L2  JECXZ final2  Repeta2:  LODSB  LODSB  STOSB  Loop Repeta2  final2:  MOV BL, 1  Sort:  CMP BL, 1  JNE final    MOV BL, 0  MOV ECX, L1+L2-1  JECXZ final  MOV ESI, S3  For:  MOV AL, [ESI]  MOV DL, [ESI+1]  CMP AL, DL  JLE next  MOV [ESI], DL  MOV [ESI+1], AL MOV BL, 1  next:  INC ESI  Loop For  JMP Sort  final |

Ok=1

While(ok==1){

Ok=0

For(I=1;I<=len-1;I++)

If(a[I+1]<a[I]){

Aux=a[I];

A[I]=a[I+1];

A[I+1]=aux

Ok=1

}

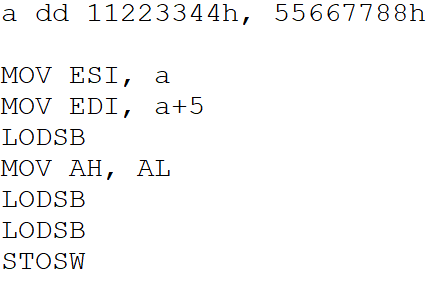
}

1. Se dă un şir de valori numerice întregi reprezentate pe dublucuvinte. Să se construiască şirul corespunzător al octeţilor din reprezentarea în memorie a dublucuvintelor date care contin doar octetii negativi.

|  |  |
| --- | --- |
| Data segment | Code segment |
| S1 dd 1, 2, 3, 4  Len equ $-S1  S2 resb Len | MOV ECX, Len  MOV ESI, S1  MOV EDI, S2  JECXZ final  loop1:  LODSB  CMP AL, 0  JGE next  STOSB  next:  LOOP loop1    Final: |
|  |  |

1. Se da un sir de dublu cuvinte. Sa se calculeze suma octetilor low ai cuvintelor high multiplii de 10.

|  |  |
| --- | --- |
| Data segment | Code segment |
| S1 dd 1, 2, 3, 4  Len equ ($-S1)/4  Sum db 0 | MOV ECX, LEN  JECXZ FINAL  MOV ESI, S1  CLD  REPETA:  LODSW  LODSW  MOV AH, 0  MOV BL, 10  MOV DX, AX  DIV BL  CMP AH, 0  JNE REIA  ADD [SUM],DL  REIA:  LOOP REPETA  FINAL: |



Care e memory layout la inceput?

Care e memory layout la final?

Ce face fiecare instructiune?

44h|33h|22h|11h|88h|77h|66h|55h

LODSB; AL=44h; ESI=a+1

MOV AH,AL; AH=44h;

LODSB; AL=33h; ESI=a+2

LODSB; AL=22h; ESI=a+3

STOSW; AX=4422h

44h|33h|22h|11h|88h|22h|44h|55h