Caruso Davide 4 B 26/10/03

RELAZIONE DI SISTEMI

Testo del problema:
Creare una macro in Assembly che copi in ordine crescente gli elementi di un vettore di caratteri (o stinga) in un altro , attraverso il metodo della ricerca del minimo valore; richiamare , in fine, questa macro all' interno di un programma completo per verificare la sua efficienza.

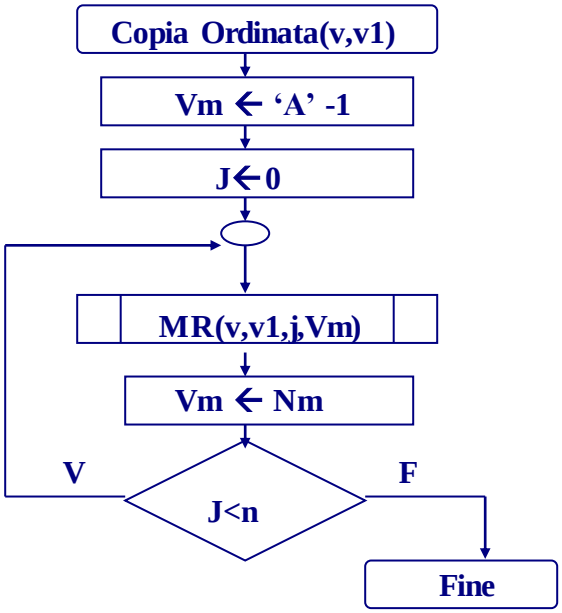
Risposta:
Dopo aver analizzato il problema proposto possiamo individuare i dati di input e di output dell'algoritmo riguardante la macro che chiameremo d'ora in avanti Copia Ordinata.

Dati di input:**v**(stringa da ordinare),**v1**(stringa in cui verranno copiati gli elementi ordinati della stringa **v**).
Dati di output:**v1**(ordinata).

Ricerca del algoritmo:
Attraverso l'utilizzo del modello di sviluppo Top-Down e in base a quanto chiesto dal problema , l'algoritmo sarà costituito , come vedremo in seguito dal diagramma di flusso , da un ciclo **for** in cui verrà richiamata ogni volta la macro **MR** che avrà il compito di cercare il minimo valore del vettore **v** e di copiarlo in **v1 ,** tenendo conto anche delle possibili ripetizioni di tale minimo , incrementando contemporaneamente **j**; inoltre , all'interno del ciclo , si assegnerà a **Vm** il minimo trovato , per aggiornarlo .

Diagramma di flusso

Variabili di lavoro:
Vm(vecchio minimo),**Nv**(nuovo minimo),**j**(indice della stringa **v1**),**i**(indice della stringa **v**),c(contatore delle possibili ripetizioni del nuovo minimo),**n**(dimensione dei due vettori).



Traduzione in linguaggio assembly

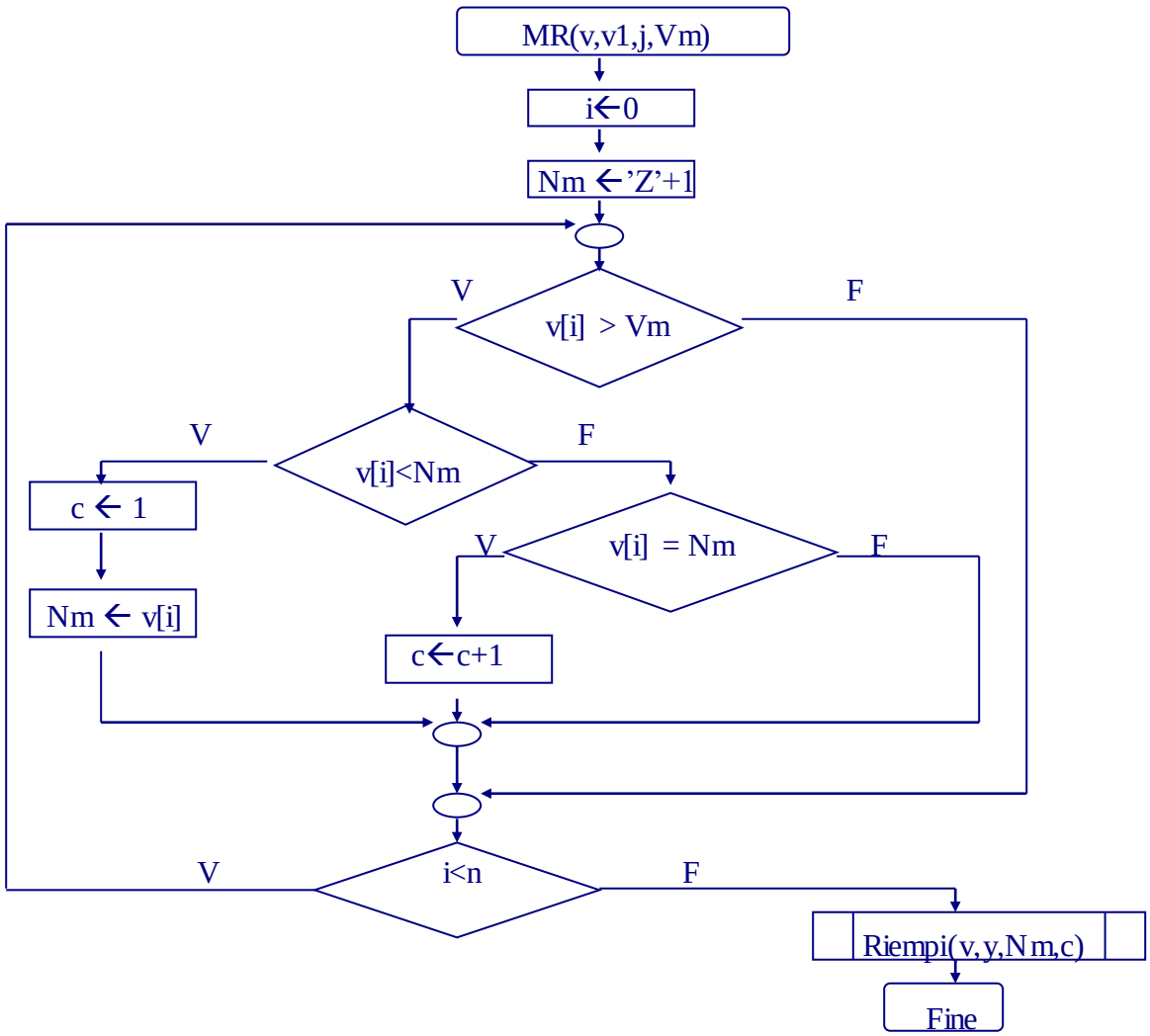
Prima di codificare l'algoritmo di questa macro in assembly occorre precisare che le due stringhe passate alla macro dovranno avere la stessa dimensione ed inoltre essendo i caratteri contenuti nelle stringhe appartenenti al codice ASCII (costituito da 256 caratteri) essi saranno considerati come vettori di byte.
Come potete dedurre della codifica seguente , l'uso dei registri per la variabili di lavoro è il seguente: **CL** per **Vm** , **CH** per **Nm** , **DI** per **j** , **BX** per **i** e **Al** per **c** .

```
COPIA_ORDINATA  MACRO V,V1,n
            MOV CL,0H
            MOV DI,0H
INIZIO_C_1:
           MR V,V1,DI,CL
            INC CL
            MOV CL,CH
           CMP DI,n
           JB INIZIO_C_1

ENDM COPIA_ORDINATA
```

Adesso dobbiamo risolvere il sottoproblema della ricerca dell'elemento di valore minimo in un vettore rappresentato dalla macro **MR**. L'algoritmo sarà costituito , come vedremo , da un'altro ciclo **for** , in cui verranno confrontati tutti gli elementi della stringa **v1** con il valore minimo di volta in volta trovato ed inoltre , per evitare di incontrare ogni volta lo stesso minimo , verrà inserita anche la condizione in cui l'elemento in questione dovrà essere maggiore del vecchio minimo;infine, sempre all'interno dello stesso ciclo ,tramite la variabile c , verrà contato il numero di ripetizioni del valore minimo. Una volta confrontati tutti gli elementi si uscirà dal ciclo e si chiamerà la macro **riempi** , che riempirà il vettore **v1** , partendo dall'elemento di posto **j** ,con il nuovo minimo , **c** volte.

Diagramma di flusso

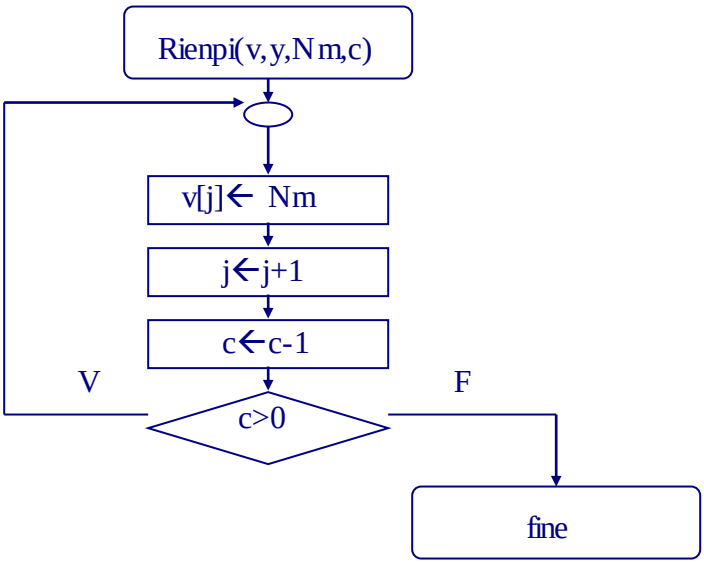


Traduzione in linguaggio assembly

```
MR MACRO V,V1,DI,CL,n
      MOV BX,0H
      MOV CH,0FFH
INIZIOC:
      CMP V[BX],CL
      JBE FINE_IF
      CMP V[BX],CH
      JB IF_2_VERO
      JE IF_3_VERO
      JMP FINE_IF
IF_2_VERO:
      MOV AL,01H
      MOV CH,V[BX]
      JMP FINE_IF
IF_3_VERO:
      INC AL
FINE_IF:
      INC BX
      CMP BX,n
      JB INIZIOC
      RIEMPI V1,DI,CH,AL
ENDM MR
```

Per completare la nostra macro dobbiamo soltanto definire la macro Riempi , la quale inserirà nel vettore **v1** il nuovo minimo nel modo precedentemente descritto. L'algoritmo della macro sarà costituito , semplicemente , da un ciclo **for** in cui a **v[j]** verrà assegnato **Nm** ,fin quando **c** , decrementato ad ogni ciclo , sarà uguale a zero.

Diagramma di flusso



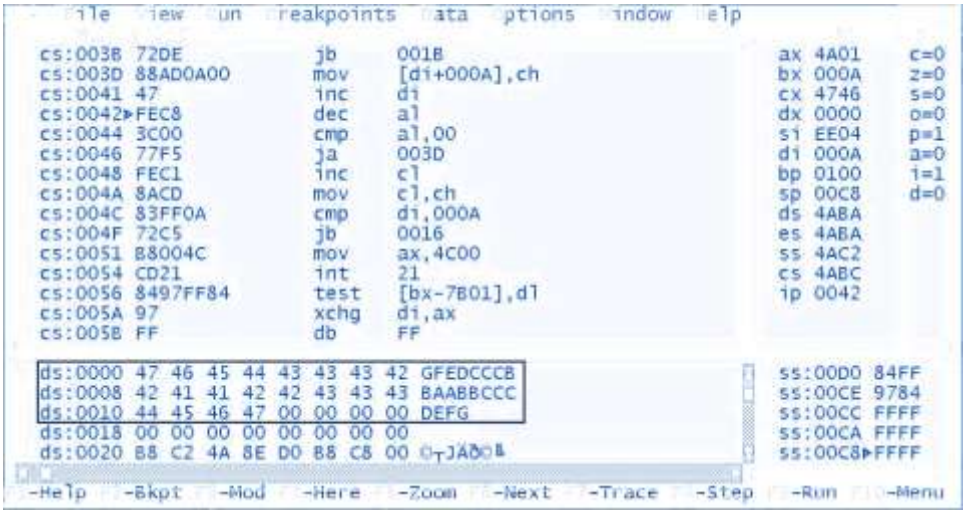
Traduzione in linguaggio assembly

```
RIEMPI MACRO V1,DI,CH,AL
INIZIO_CICLO:
      MOV V1[DI],CH
      INC DI
      DEC AL
      CMP AL,00H
      JA INIZIO_CICLO
ENDM RIEMPI
```

Adesso , per verificare l'effettiva efficienza della macro , come richiesto , dobbiamo costruire un programma del tipo :

```
Dati SEGMENT
          vOri DB 0AH DUP(77H)
          vCop DB 0AH DUP(77H)
Dati ENDS
SISTEMA SEGMENT STACK "STACK"
   ……….
SISTEMA ENDS
Codice SEGMENT
         ;definizione dei segmenti
         ASSUME CS:Codice ,SS:Sistema,DS:DATI,ES:Dati
inizio:          ;inizializzazione dei registri di segmento
            ……….……
            ;PROGRAMMA
            COPIA_ORDINATA vOri,vCop,0Ah  ;Chiamata alla Macro COPIA_ORDINATA
                                          ;e passaggio dei parametri formali vOri , v Cop e 0Ah
                                          ;che al momento della compilazione , tramite
                                          ;l'assemplatore(Tasm.exe) , verranno sostituiti ai
                                          ; parametri formali della macro costruita.
            ;ritorno al sistema operativo
            MOV AX,  4C00h
            INT 21h
codice ENDS
END Inizio
```

E inserire in testa al programma tutte le macro precedentemente realizzate.
Successivamente , dobbiamo compilare e "lincare" il programma . Se in queste due fasi non saranno riscontrati degli errori, verrà creato il file eseguibile . Per verificare che il programma ordini effettivamente la stinga **vOri** in **vCop ,** dopo aver aperto questo programma eseguibile tramite il Turbo Debugger **,** dobbiamo inserire nel vettore **v** , situato tra DS:0000 e DS:000A , dei caratteri casuali (in sostituzione al carattere 'W') e successivamente dobbiamo eseguire tutte le istruzioni del codice , tramite la pressione del tasto [F7] .
Otterremo alla fine, in caso positivo, un risultato simile a quello illustrato nella pagina seguente , cioè con il vettore **vCop** (DS:000B-DS:0014) costituito degli elementi di **vOri** ordinati.



Commento:

Il programma costruito ordina soltanto i caratteri maiuscoli , per ordinare i caratteri minuscoli occorre , o ricostruire le macro Copia Ordinata e RM utilizzando anziché il codice ASCII del carattere ‘A’ e ‘Z’ quello di ‘a’ e ’z’ ,oppure ,più semplicemente , passare ad entrambe le macro un altro parametro che chiameremo car (carattere) che avrà il valore del simbolo ‘A’ o ‘a’ , in base al tipo di caratteri che si vuole ordinare (maiuscoli o minuscoli) ed inserire dove vi è il confronto con la lettera <z > :car +21h.