1.asm.

```
STACK1    SEGMENT  PARA STACK
STACK_AREA  DW     100H DUP(?)
STACK_BOTTOM EQU   $-STACK_AREA
STACK1    ENDS

DATA1     SEGMENT  PARA
          DB       20 DUP(?)
STRING1   DB       'TANLIDE', '$'
STRING1_LEN EQU    $-STRING1
          DB       20 DUP(?)
NEW_LINE  DB       0DH,0AH,'$'
DATA1     ENDS

CODE1     SEGMENT  PARA
          ASSUME   CS:CODE1,DS:DATA1
          ASSUME   ES:DATA1,SS:STACK1

PRINT_STRING PROC  FAR
          PUSH     AX
          MOV      AX,9
          INT      21H
          POP      AX
          RET
PRINT_STRING ENDP

PRINT_NEWLINE PROC FAR
          PUSH     DX
          MOV      DX, OFFSET NEW_LINE
          CALL     PRINT_STRING
          POP      DX
PRINT_NEWLINE ENDP

Memmove   PROC     FAR
          PUSH     SI
          PUSH     DI
          MOV      CX, STRING1_LEN
          CLD
          REP      MOVSB
          POP      DI
          POP      SI
          RET
Memmove   ENDP

DO_STH    PROC     FAR
          PUSH     SI
          PUSH     DI
          PUSH     DX
          LEA      SI, STRING1
          LEA      DI, DX
          MOV      DX, SI
          CALL     PRINT_STRING
```

```
        CALL    PRINT_NEWLINE
        CALL    Memmove
        MOV     DX , [SI]
        CALL    PRINT_STRING
        CALL    PRINT_NEWLINE
        MOV     PX , DI
        CALL    PRINT_STRING
        CALL    PRINT_NEWLINE
        POP     PX
        POP     DI
        POP     SI
DO-STH  ENDP

START:
MAIN    PROC    FAR
        MOV     AX , STACK1
        MOV     SS , AX
        MOV     SP , & STACK_BOTTOM
        MOV     AX , DATA1
        MOV     DS , AX
        MOV     ES , AX

        MOV     DX , 10H
        CALL    DO-STH
        MOV     DX , -5H
        CALL    DO-STH
        MOV     DX , 5H
        CALL    DO-STH

EXIT:
        MOV     AX , 4C00H
        INT     21H
MAIN    ENDP
DATA1   ENDS
        END     MAIN
```

2.asm

```
STACK1          SEGMENT     PARA    STACK
STACK_AREA      DW          100H    DUP(?)
STACK_BOTTOM    EQU         $-STACK_AREA
STACK1          ENDS

DATA1           SEGMENT     PARA
STRING1         DB          'TANLIPE','$'
STRING1_LEN     EQU         $-STRING1
STRING2_MAX     EQU         20H
STRING2_BUF     DB          STRING2_MAX-1
STRING2_LEN     DB          ?
STRING2         DB          STRING2_MAX DUP(?)
NEW_LINE        DB          0DH,0AH,'$'
DATA1           ENDS

CODE1           SEGMENET    PARA
                ASSUME      CS:CODE1,DS:DATA1
                ASSUME      ES:DATA1, SS:STACK1

PRINT_BYTE:     PROC        FAR
                PUSH        AX
                MOV         AH,2
                INT         21H
                POP         AX
                RET
PRINT_BYTE      ENDP

PRINT_STRING    PROC        FAR
                PUSH        AX
                MOV         AH,9
                INT         21H
                POP         AX
                RET
PRINT_STRING    ENDP

PRINT_NEWLINE   PROC        FAR
                PUSH        DX
                MOV         DX,OFFSET NEW_LINE
                CALL        PRINT_STRING
                POP         DX
                RET
PRINT_NEWLINE   ENDP

GET_STRING      PROC        FAR
                PUSH        AX
                MOV         AH,0AH
                INT         21H
                POP         AX
                RET
GET_STRING      ENDP
```

```
STR_CMP     PROC
            PUSH  ES
            PUSH  DS
            POP   ES

            MOV   CH, 0
            MOV   CL, STRING1_LEN
            CMP   CL, STRING2_LEN
            JA    STR_CMP_1
            MOV   CL, STRING2_LEN

STR_CMP_1:
            CLD
            REPZ  CMPSB
            JA    STR_ABOVE
            JB    STR_BELOW

            MOV   DL, '='
            JMP   STR_CMP_2

STR_ABOVE:
            MOV   DL, '>'
            JMP   STR_CMP_2

STR_BELOW:
            MOV   DL, '<'

STR_CMP_2:
            CALL  PRINT_BYTE
            POP   ES

            RET

STR_CMP     ENDP
```

```
START:
MAIN        PROC  FAR

            MOV   AX, STACK1
            MOV   SS, AX
            MOV   SP, STACK_BOTTOM
            MOV   AX, DATA1
            MOV   DS, AX

            MOV   DX, OFFSET STRING2_BUF
            CALL  GET_STRING
            MOV   BH, 0
            MOV   BL, STRING2_LEN
            MOV   BYTE PTR STRING2[BX], '$'
            CALL  PRINT_NEWLINE

            MOV   SI, OFFSET STRING1
            MOV   DI, OFFSET STRING2
            CALL  STR_CMP

            MOV   DX, OFFSET STRING2
            CALL  PRINT_STRING

            CALL  PRINT_NEWLINE

EXIT:
            MOV   AX, 4C00H
            INT   21H

MAIN        ENDP
RCODE1      ENDS
            END   MAIN
```