Nome, cognome, matricola ….....……..……..……

Esercizio di programmazione

sino a 12 punti – è possibile consultare solamente l'instruction set Intel - tempo: 60 minuti

Un problema comune per compilatori ed editor di testo consiste nel determinare se le parentesi all'interno di una stringa sono bilanciate e correttamente annidate. Ad esempio:

1. (((1+2)*3)-(4-8)) OK: parentesi bilanciate e correttamente annidate
2. (1+2)+)4-1(NO: parentesi non correttamente annidate
3. (1+2*(5-3) NO: parentesi non bilanciate

Si scriva in linguaggio Assembly 8086 una procedura **verificaParentesi** che controlli la correttezza delle parentesi all'interno di una stringa. La lunghezza della stringa DIM è definita tramite costante. La procedura riceve l'offset della stringa tramite stack. Non è ammesso l'uso di variabili.

La procedura **verificaParentesi** deve restituire attraverso il registro BX:
- il valore DIM se le parentesi sono bilanciate e correttamente annidate
- altrimenti, la posizione della prima parentesi che non soddisfa i requisiti di bilanciamento e annidamento.

Negli esempi proposti, il valore restituito è:

1. (((1+2)*3)-(4-8)) BX = DIM
2. (1+2)+)4-1(BX = 6
3. (1+2*(5-3) BX = 0

Una possibile realizzazione dell'algoritmo in pseudocodice è la seguente:

```
char s[DIM];
int par = 0;
for (i=0; i<DIM; i++)
  {
  if (s[i] == '(')
    {
    par++;
    push(i);
    }
  else if (s[i] == ')')
    {
    par--;
    if (par < 0) return i;
    else pop();
    }
  }
if (par!=0)
  return pop();
return DIM;
```

Di seguito un esempio di programma chiamante:

```
DIM EQU 17
.MODEL small
.STACK
.DATA
string DB "(((1+2)*3)-(4-8))"

.CODE
.STARTUP
…
PUSH OFFSET string
CALL verificaParentesi
ADD SP, 2
…
.EXIT
```

Soluzioni proposte

```
verificaParentesi PROC
        MOV BP, SP
        MOV AX, 0
        MOV BX, [BP + 2]
        MOV CX, DIM
        MOV SI, 0

while:  CMP [BX][SI], '('
        JNE close
        INC AX
        PUSH SI
        JMP next
close:  CMP [BX][SI], ')'
        JNE next
        DEC AX
        JS error1
        POP DI
next:   INC SI
        LOOP while

        CMP AX, 0
        JNE error2
; parentesi corrette
        MOV BX, DIM
        JMP return

; errore annidamento
error2: POP BX
        DEC AX
        SHL AX, 1
        ADD SP, AX
        JMP return

; errore bilanciamento
error1: MOV BX, SI

return:
        RET
verificaParentesi ENDP
end
```