```
#################################################################
# Scrivere un programma Assembler che si comporta come segue:
# 1. Chiede in ingresso un numero INTERO x in base 10,
#    rappresentato in modulo e segno, supponendo che il modulo stia su 8 bit
# 2. aggiunge il numero x digitato, ad un accumulatore s, che sta su *10 bit*
#    (1 bit di segno e 9 di modulo). s Ã¨ inizialmente pari a zero.
#    Se l'operazione di somma algebrica ritorna un numero non rappresentabile
#    su 10 bit in modulo e segno, stampa "overflow" e termina.
#    Altrimenti stampa s in modulo e segno.
#
#  Esempio:
#  ?+255
#  +255
#  ?-20
#  +235
#  ?+255
#  +490
#  ?+255
#  overflow
#####################################################################

.GLOBAL _main

.DATA
modulo:         .WORD 0x0000
segno:          .BYTE 0x00
message:        .ASCII "overflow"

.TEXT
_main:          NOP
                CALL newline
                MOV $'?',%AL
                CALL outchar
in_sgn:         CALL inchar                             #segno
                CMP $'-',%AL
                JE ok
                CMP $'+',%AL
                JE ok
                JMP in_sgn
ok:             CALL outchar
                MOV %AL, segno
                CALL indecimal_tiny             #modulo
                CALL newline

                MOV $0,%AH
                CMPB $'+', segno
                JNE sottrai
                ADD %AX, modulo
                JMP check
sottrai:        SUB %AX, modulo
check:          CMPW $-512, modulo
                JLE overflow
                CMPW $+512, modulo
                JGE overflow

                CMPW $+0, modulo                #stampa
                JL  negativo
                MOV $'+', %AL
                CALL outchar
                MOV modulo, %AX
                JMP stampa_m
negativo:       MOV $'-', %AL
                CALL outchar
                MOV modulo, %AX
                NEG %AX
stampa_m:       CALL outdecimal_short
                JMP _main
overflow:       MOV $8, %CX
                LEA message, %EBX
                CALL outmess
                CALL newline
                RET

.INCLUDE "C:/amb_GAS/utility"
```