```
VAR
Fp : ARRAY[ 0..1 ] OF FILEP; (* File pointer *)
SktLsn : SysSktListen; (* FB socket listen *)
WMsgPulse : BOOL; (* Welcome message pulse *)
RxChars : ARRAY[ 0..1 ] OF INT; (* Auxiliary variable *)
RxString : STRING[ 256 ]; (* Rx command string *)
IChars : ARRAY[ 0..1 ] OF INT; (* Input characters *)
i : INT; (* Auxiliary variable *)
END_VAR
```

```
(* ************************************************************************** *)
(* GESTIONE PROTOCOLLO ASCII SU CONNESSIONE TCP                               *)
(* ************************************************************************** *)
(* Questo programma esegue gestione del protocollo ASCII su TCP.              *)
(* -------------------------------------------------------------------------- *)

    (* ---------------------------------------------------------------------- *)
    (* ESEGUO INIZIALIZZAZIONI                                                *)
    (* ---------------------------------------------------------------------- *)
    (* Eseguo apertura porta seriale esocket TCP. *)

    IF (SysFirstLoop) THEN
        Fp[0]:=Sysfopen('COM0', 'rw'); (* File pointer *)
        Fp[1]:=Sysfopen('TCPSKT', 'rw'); (* File pointer *)
    END_IF;

    (* ---------------------------------------------------------------------- *)
    (* GESTIONE CONTROLLO STATO SOCKET TCP                                    *)
    (* ---------------------------------------------------------------------- *)
    (* Forzo socket in condizione di listening. *)
    (* "LifeTm" piccolo, forza chiusura socket su mancanza comunicazione. *)
    (* "FlushTm" deve essere più grande del tempo di loop programma. *)

    SktLsn.File:=Fp[1]; (* Flusso dati stream *)
    SktLsn.MyPort:=2000; (* Porta in ascolto su socket *)
    SktLsn.LifeTm:=60; (* Tempo di vita socket (S) *)
    SktLsn.FlushTm:=10; (* Tempo di flush socket (mS) *)
    SktLsn.RxSize:=200; (* Dimensione buffer Rx dati socket *)
    SktLsn.TxSize:=200; (* Dimensione buffer Tx dati socket *)
    SktLsn(Enable:=TRUE); (* Forzo socket in listening *)

    (* ---------------------------------------------------------------------- *)
    (* GESTIONE CONNESSIONE TCP                                               *)
    (* ---------------------------------------------------------------------- *)
    (* Su connessione TCP il sistema invia un messaggio di Welcome. *)

    IF (SktLsn.Connect <> WMsgPulse) THEN
        WMsgPulse:=SktLsn.Connect; (* Welcome message pulse *)

        IF (SktLsn.Connect) THEN
            i:=SysVarfprintf(Fp[1], '%s$r$n', STRING_TYPE, ADR('Welcome !'));
        END_IF;
    END_IF;

    (* Se nessun client TCP è connesso esco. *)

    IF NOT(SktLsn.Connect) THEN RETURN; END_IF;
```

```
48
49     (* ------------------------------------------------------------------- *)
50     (* CONTROLLO RICEZIONE CARATTERI                                       *)
51     (* ------------------------------------------------------------------- *)
52     (* Ricevo caratteri da porta seriale e dal socket TCP. Se non ricevo più  *)
53     (* caratteri per almeno un loop di programma invio i caratteri ricevuti   *)
54     (* uno stream verso l'altro stream. Se caratteri ricevuti superano i 3/4  *)
55     (* della dimensione del buffer di ricezione viene gestito l'invio.        *)
56     (* ------------------------------------------------------------------- *)
57     (* Controllo se ricezione caratteri da porta seriale. *)
58
59     IF (((SysGetIChars(Fp[0]) = IChars[0]) AND (IChars[0] > 0)) OR (IChars[0] > 192)) THEN
60         RxChars[0]:=Sysfread(ADR(RxString), 1, IChars[0], Fp[0]); (* Received characters *)
61         RxChars[0]:=Sysfwrite(ADR(RxString), 1, RxChars[0], Fp[1]); (* Transmitted characters *)
62     END_IF;
63
64     (* Controllo se ricezione caratteri da socket TCP. *)
65
66     IF (((SysGetIChars(Fp[1]) = IChars[1]) AND (IChars[1] > 0)) OR (IChars[1] > 192)) THEN
67         RxChars[1]:=Sysfread(ADR(RxString), 1, IChars[1], Fp[1]); (* Received characters *)
68         RxChars[1]:=Sysfwrite(ADR(RxString), 1, RxChars[1], Fp[0]); (* Transmitted characters *)
69     END_IF;
70
71     (* Salvo numero caratteri in ricezione dai due streams. *)
72
73     IChars[0]:=SysGetIChars(Fp[0]); (* Input characters *)
74     IChars[1]:=SysGetIChars(Fp[1]); (* Input characters *)
75
76 (* [End of file] *)
77
78
```