

```

program test1 (input, output);

var
  i : integer;

begin
  i:= 0;
  if i=0 then begin writeln(i) end
else begin writeln(i+1)end
end.

```

```

-----

CASL      START   BEGIN
BEGIN     LAD      GR6, 0   ; reset write buffer
          LAD      GR7, LIBBUF   ; set output buffer address
          LAD      GR5, 0   ; set FP(GR5) null
          JUMP     LLMAIN   ; go to the main body
LLMAIN    NOP      ; entry of main body
          LAD      GR1,0    ; set integer constant
          ST       GR1,ZZi  ; complete assignment
          LD       GR1,ZZi  ; load global variable
          PUSH     0,GR1    ; Save GR1 for exp comparison
          LAD      GR1,0    ; set integer constant
          POP      GR2      ; Restore the 1st operant to GR1 for
comparison
          CPA      GR2,GR1
          LAD      GR1,1    ; Initilize GR1 with True
          JZE     LB1      ; jump EQUAL
          XOR      GR1,GR1  ; set GR1 to False (zero)
LB1       NOP
          ; End of Exp comparison
          AND      GR1,GR1  ; check if's condition
          JZE     LB2      ; jump if not true
          LD       GR1,ZZi  ; load global variable
          LD       GR2,GR1
          CALL     WRTINT
          CALL     WRTLN
          JUMP     LB3      ; exit of if statement
LB2       NOP
          ;
          LD       GR1,ZZi  ; load global variable
          PUSH     0,GR1    ; start +
          LAD      GR1,1    ; set integer constant
          LD       GR2,GR1
          POP      GR1
          ADDA    GR1,GR2   ; End of +
          LD       GR2,GR1
          CALL     WRTINT
          CALL     WRTLN
LB3       NOP
          ;
          RET
; start of global variable area
ZZi      DS      1
; end of global variable area

```

```

LIBBUF DS      256
      END
; lib.cas
;=====
; MULT: 掛け算を行うサブルーチン
; GR1 * GR2 -> GR2
MULT   START
      PUSH    0,GR1    ; GR1の内容をスタックに退避
      PUSH    0,GR3    ; GR3の内容をスタックに退避
      PUSH    0,GR4    ; GR4の内容をスタックに退避
      LAD     GR3,0    ; GR3を初期化
      LD      GR4,GR2
      JPL     LOOP
      XOR     GR4,=#FFFF
      ADDA    GR4,=1
LOOP   SRL     GR4,1
      JOV     ONE
      JUMP    ZERO
ONE    ADDL   GR3,GR1
ZERO  SLL     GR1,1
      AND     GR4,GR4
      JNZ     LOOP
      CPA     GR2,=0
      JPL     END
      XOR     GR3,=#FFFF
      ADDA    GR3,=1
END    LD      GR2,GR3
      POP     GR4
      POP     GR3
      POP     GR1
      RET
      END
;=====
; DIV 割り算を行うサブルーチン
; GR1 / GR2 -> 商は GR2, 余りは GR1
DIV    START
      PUSH    0,GR3
      ST      GR1,A
      ST      GR2,B
      CPA     GR1,=0
      JPL     SKIPA
      XOR     GR1,=#FFFF
      ADDA    GR1,=1
SKIPA  CPA     GR2,=0
      JZE     SKIPD
      JPL     SKIPB
      XOR     GR2,=#FFFF
      ADDA    GR2,=1
SKIPB  LD      GR3,=0
LOOP   CPA     GR1,GR2
      JMI     STEP
      SUBA    GR1,GR2

```

```

STEP      LAD      GR3,1,GR3
          JUMP     LOOP
          LD       GR2,GR3
          LD       GR3,A
          CPA      GR3,=0
          JPL      SKIPC
          XOR      GR1,=#FFFF
          ADDA     GR1,=1
SKIPC     XOR      GR3,B
          CPA      GR3,=0
          JZE      SKIPD
          JPL      SKIPD
          XOR      GR2,=#FFFF
          ADDA     GR2,=1
SKIPD     POP      GR3
          RET
A         DS      1
B         DS      1
          END

```

```

;=====

```

```

; 入力装置から数値データを読み込み,

```

```

; その内容をGR2が指すアドレスに格納するサブルーチン

```

```

RDINT     START
          PUSH     0,GR1      ; GR1の内容をスタックに退避
          PUSH     0,GR3      ; GR3の内容をスタックに退避
          PUSH     0,GR4      ; GR4の内容をスタックに退避
          PUSH     0,GR5      ; GR5の内容をスタックに退避
          PUSH     0,GR6      ; GR6の内容をスタックに退避
          LD       GR5,GR2    ; GR2が指す番地をGR5にコピー
          LD       GR2,=0     ; GR2を初期化
          LD       GR3,=0     ; GR3を初期化
          IN       INAREA,INLEN ; 入力を受け取る
          ; 入力がnullかどうかのチェック
          CPA      GR3,INLEN
          JZE      ERROR
          ; 最初の文字が'-'かどうかのチェック
          LD       GR4,INAREA,GR3
          LAD      GR3,1,GR3
          LD       GR6,GR4    ; GR6に入力された先頭の文字を保存
          CPL      GR4,=#002D ; '-'かどうか
          JZE      LOOP
          CPL      GR4,='0'   ; 数値かどうかのチェック
          JMI      ERROR
          CPL      GR4,='9'
          JPL      ERROR
          XOR      GR4,=#0030 ; 数値だったら変換
          ADDA     GR2,GR4
          ; 「すでに読み込んだ数値を10倍して、新しく読み込んだ数値と足す」を繰り返す

```

返す

```

LOOP      CPA      GR3,INLEN
          JZE      CODE      ; 入力された文字数とGR3が同じであればループを抜ける
          LD       GR1,=10
          CALL    MULT      ; GR2の値を10倍する
          LD       GR4,INAREA,GR3
          CPL     GR4,='0' ; 数値かどうかのチェック
          JMI     ERROR
          CPL     GR4,='9'
          JPL     ERROR
          XOR     GR4,=#0030      ; GR4の内容を数値に変換
          ADDA   GR2,GR4 ; GR2にGR1の内容を足す
          LAD    GR3,1,GR3      ; GR3(ポインタ)をインクリメント
          JUMP   LOOP
          ; 最初の文字が '-' であった場合は-1倍する
CODE      CPL     GR6,=#002D
          JNZ     END
          XOR     GR2,=#FFFF
          LAD    GR2,1,GR2
          JUMP   END
          ; エラーを出力する
ERROR     OUT     ERRSTR,ERRLEN
END       ST      GR2,0,GR5      ; GR2の内容をGR5が指す番地に格納する
          LD     GR2,GR5 ; GR5が指す番地をGR2に戻す
          POP   GR6
          POP   GR5
          POP   GR4
          POP   GR3
          POP   GR1
          RET
ERRSTR   DC     'illegal input'
ERRLEN   DC     13
INAREA   DS     6
INLEN    DS     1
          END

;=====
; 入力装置から文字を読み込み,
; その内容をGR2が指すアドレスに格納するサブルーチン
RDCH     START
          IN     INCHAR,INLEN
          LD     GR1,INCHAR
          ST     GR1,0,GR2
          RET
INCHAR   DS     1
INLEN    DS     1
          END

;=====
; 入力装置から, GR1の文字数を読み込む.
; 読み込んだ文字列は, GR2 が指すアドレスから順に格納される
RDSTR    START

```

```

        PUSH    0,GR3    ; GR3の内容をスタックに退避
        PUSH    0,GR4    ; GR4の内容をスタックに退避
        PUSH    0,GR5    ; GR5の内容をスタックに退避
        LAD     GR4,0     ; GR4を初期化
        IN      INSTR,INLEN
LOOP    CPA     GR4,GR1
        JZE     END      ; GR1で指定された文字数を越えたら終わり
        CPA     GR4,INLEN
        JZE     END      ; 入力された文字数を越えたら終わり
        LD      GR5,GR2
        ADDA    GR5,GR4  ; 文字の格納先番地を計算
        LD      GR3,INSTR,GR4
        ST      GR3,0,GR5
        LAD     GR4,1,GR4
        JUMP    LOOP
END     POP     GR5
        POP     GR4
        POP     GR3
        RET
INSTR   DS     256
INLEN   DS     1
        END

```

```

;=====
; 入力装置からの文字列を改行まで読み飛ばすサブルーチン

```

```

RDLN   START
        IN      INAREA,INLEN
        RET
INAREA  DS     256
INLEN   DS     1
        END

```

```

;=====
; GR2の内容（数値データ）を出力装置に書き出すサブルーチン
; このサブルーチンが呼ばれたとき、
; GR7には、出力用番地の先頭アドレスが、
; GR6には、現在出力用番地に入っている文字数が、
; それぞれ格納されている。

```

```

WRTINT  START
        PUSH    0,GR1    ; GR1の内容をスタックに退避
        PUSH    0,GR2    ; GR2の内容をスタックに退避
        PUSH    0,GR3    ; GR3の内容をスタックに退避
        PUSH    0,GR2    ; 数値データをもう一度スタックに退避
        LD      GR3,=0    ; GR3はインデックスとして用いる
        ; 数値データが負数である場合は、正の数に変換
        CPA     GR2,=0
        JPL     LOOP1
        XOR     GR2,=#FFFF
        ADDA    GR2,=1
        ; 数値データを変換しながら、バッファに格納
LOOP1   LD      GR1,GR2

```

```

LD      GR2,=10
CALL   DIV
XOR    GR1,=#0030
ST     GR1,BUFFER,GR3
LAD    GR3,1,GR3
CPA    GR2,=0
JNZ    LOOP1
; 数値データが負数であれば, '-'を追加
POP    GR2
CPA    GR2,=0
JZE    LOOP2
JPL    LOOP2
LD     GR1,='- '
ST     GR1,BUFFER,GR3
LAD    GR3,1,GR3
; BUFFERを逆順にたどりながら, 出力用バッファに格納
LOOP2  LAD    GR3,-1,GR3
LD     GR1,BUFFER,GR3
LD     GR2,GR7
ADDA   GR2,GR6
ST     GR1,0,GR2
LAD    GR6,1,GR6
CPA    GR3,=0
JNZ    LOOP2
END    POP    GR3
      POP    GR2
      POP    GR1
      RET
BUFFER DS    6
      END

```

```

;=====
; GR2の内容(文字)を出力装置に書き出すサブルーチン
; このサブルーチンが呼ばれたとき,
; GR7には, 出力用番地の先頭アドレスが,
; GR6には, 現在出力用番地に入っている文字数が,
; それぞれ格納されている.

```

```

WRTCH  START
      PUSH   0,GR1 ; GR1の内容をスタックに退避
      LD     GR1,GR7
      ADDA   GR1,GR6 ; GR1に次の文字を格納する番地を代入
      ST     GR2,0,GR1
      LAD    GR6,1,GR6
      POP    GR1
      RET
      END

```

```

;=====
; GR2の指すメモリ番地から, 長さGR1の文字列を出力装置に書き出すサブルーチン
; このサブルーチンが呼ばれたとき,
; GR7には, 出力用番地の先頭アドレスが,
; GR6には, 現在出力用番地に入っている文字数が,
; それぞれ格納されている.

```

```

WRTSTR  START
        PUSH    0,GR3    ; GR3の内容をスタックに退避
        PUSH    0,GR4    ; GR4の内容をスタックに退避
        PUSH    0,GR5    ; GR5の内容をスタックに退避
        LAD     GR3,0     ; GR3は制御変数として用いる
LOOP     CPA     GR3,GR1
        JZE     END
        LD      GR4,GR2
        ADDA    GR4,GR3   ; 出力する文字の格納番地を計算
        LD      GR5,0,GR4 ; 出力する文字をレジスタにコピー
        LD      GR4,GR7
        ADDA    GR4,GR6   ; 出力先の番地を計算
        ST      GR5,0,GR4 ; 出力装置に書き出し
        LAD     GR3,1,GR3
        LAD     GR6,1,GR6
        JUMP    LOOP
END      POP     GR5
        POP     GR4
        POP     GR3
        RET
        END

```

```

; =====
; 改行を出力装置に書き出すサブルーチン
; 実質的には、GR7で始まるアドレス番地から長さGR6の文字列を出力する

```

```

WRTLN   START
        PUSH    0,GR1
        PUSH    0,GR2
        PUSH    0,GR3
        ST      GR6,OUTLEN
        LAD     GR1,0
LOOP     CPA     GR1,OUTLEN
        JZE     END
        LD      GR2,GR7
        ADDA    GR2,GR1
        LD      GR3,0,GR2
        ST      GR3,OUTSTR,GR1
        LAD     GR1,1,GR1
        JUMP    LOOP
END      OUT     OUTSTR,OUTLEN
        LAD     GR6,0     ; 文字列を出力して、GR6を初期化
        POP     GR3
        POP     GR2
        POP     GR1
        RET
OUTSTR   DS      256
OUTLEN   DS      1
        END

```