Ejercicio 3. Considere el problema `sumaDivisores`, dado por la siguiente especificación:

proc `sumaDivisores` (in n: $\mathbb{Z}$) : $\mathbb{Z}$
    requiere $\{n \geq 1\}$
    asegura $\{res = \sum_{j=1}^{n} (\text{if } n \ mod \ j = 0 \text{ then } j \text{ else } 0 \text{ fi})\}$

a) Escribir un programa en SmallLang que satisfaga la especificación del problema y que contenga exactamente un ciclo.

b) Escribir la pre y post condición del ciclo y su invariante.

c) Considere el siguiente invariante para este problema

$$I \equiv 1 \leq i \leq n/2 \wedge res = \sum_{j=1}^{i} (\text{if } n \ mod \ j = 0 \text{ then } j \text{ else } 0 \text{ fi})$$

Si no coincide con el propuesto en el inciso anterior, ¿qué cambios se le deben hacer al programa para que lo represente este invariante? ¿Deben cambiar la pre y post condición?

Va desde j=1 hasta n incluido

```
j := 1
res := 0
WHILE (j ≤ n) DO
    if (n MOD j = 0) THEN
        res := res + j
    ELSE
        SKIP
    ENDIF
j := j + 1
```

ENDWHILE

b) $P_C: \{ M \geq 1 \wedge j = 1 \wedge res = 0 \}$

$Q_C: \{ j = M \wedge res = \sum_{j=1}^{M} \text{if } M \text{ MOD } J = 0 \text{ THEN } J \text{ ELSE } 0 \text{ fi} \}$

$I: \{ \underbrace{1 \leq j \leq m}_{\text{CONTROLO } j.} \wedge_L \; res = \sum_{j=1}^{m} \text{if } m \text{ MOD } J = 0 \text{ THEN } J \text{ ELSE } 0 \text{ fi} \}$

c) El invariante con $1 \leq i \leq \frac{n}{2}$ me dice que va a i tomará valores pero solo la MITAD de ellos.

El programa, con ese invariante sería

$j := 1$

$res := 0$

WHILE $(j \leq \frac{m}{2})$ DO

if $(M \text{ MOD } J = 0)$ THEN

$res := res + j$

ELSE

SKIP

ENDIF

j := j + 1

ENDWHILE

No, no debe CAMBIAR Pc, pero Qc sí. (en la memotoria) pues solo se modificó el rango q tomará el ciclo y no se habla de ellos.

Ejercicio 4. Considere la siguiente especificación e implementación del problema `copiarSecuencia`, y la pre y post condiciones del ciclo

Especificación

proc copiarSecuencia (in s: $array < \mathbb{Z} >$, inout r: $array < \mathbb{Z} >$)
  requiere $\{|s| = |r| \land r = r_0\}$
  asegura $\{|s| = |r| \land_L (\forall j : \mathbb{Z})(0 \leq j < |s| \rightarrow_L s[j] = r[j])\}$

Implementación en SmallLang

```
i := 0;
while (i < s.size()) do
  r[i]:=s[i];
  i:=i+1
endwhile
```

$$P_c \equiv |s| = |r| \land i = 0$$

$$Q_c \equiv (\forall j : \mathbb{Z})(0 \leq j < |r| \rightarrow_L s[j] = r[j])$$

a) ¿Qué variables del programa deben aparecer en el invariante?

b) Proponer un invariante e indicar qué clausula del mismo es necesario para cada paso de la demostración.

c) Proponer una función variante que permita demostrar que el ciclo termina.

d) Comparar la solución propuesta con la que ofrecemos al final de la guía.

Los dos arrays tienen misma longitud.

Empieza a copiar desde i = 0.

a) Las que sean relevantes durante todo el programa. En este caso acotan el rango de copia. Las longitudes no tanto xq ya son Pc y son ARRAYS => la long no cambia.

b) $I \equiv (0 \leq i \leq |N| \wedge_L (\forall j : \mathbb{Z})(0 \leq j < i \rightarrow_L N[j] = r[j]))$

c) $F_V = |N| - i$ pues como i crece, la F_V es decreciente.

d) ✓

Ejercicio 5. Sea el siguiente ciclo con su correspondiente precondición y postcondición:

```
while (i >= s.size() / 2) do
        suma := suma + s[s.size()-1-i];
    i := i - 1
endwhile
```

$$P_c : \{|s| \ mod \ 2 = 0 \wedge i = |s| - 1 \wedge suma = 0\}$$

$$Q_c : \{|s| \ mod \ 2 = 0 \wedge i = |s|/2 - 1 \ \wedge_L \ suma = \sum_{j=0}^{|s|/2-1} s[j]\}$$

a) Proponer un invariante e indicar qué clausula del mismo es necesario para cada paso de la demostración.

b) Proponer una función variante que permita demostrar que el ciclo termina.

c) Comparar la solución propuesta con la que ofrecemos al final de la guía.

i VA DECRECIENDO. PARA CUANDO $i = \frac{|N|}{2}$

$i = |N| - 1 \qquad |i| \leq \frac{|N|}{2}$

En el invariante sólo nos importa el rango del cálculo, no hay otro factor q' debe valer antes, durante y después del ciclo.

Fijó $|v| \bmod 2 = 0$ pero v No es múltiple

a) $$I \equiv \left(\overbrace{\frac{|v|}{2} - 1}^{Qc} \leq i \leq \overbrace{|v|-1}^{Pc} \wedge_2 \sum_{j=\frac{|v|}{2}-1}^{|v|-1} v[j] \right)$$

$|v| = 4 \qquad \frac{4}{2} = 2$

$v[4-1-3] = 0 \qquad i = 3$

$v[4-1-2] = 1 \qquad i = 2$

$v[4-1-1] = 2 \qquad i = 1$

Acá termina pues $1 \ngtr 2$.

$$\left(\overbrace{\frac{|v|}{2} - 1}^{Qc} \leq i \leq \overbrace{|v|-1}^{Pc} \wedge_2 \sum_{j=\frac{|v|}{2}-1}^{|v|-1} v[|v|-1-j] \right.$$

$\rightarrow$ solución profes

b) Fv: i: $\quad |v| =$ ——— $\quad \sum_{j=0}^{|v|-2-i} v[j]$

$\frac{|N|}{2} - 1 + i$ ¿llega a 0?

$|N| = 4 \quad i = 3 \quad \frac{4}{2} - 1 = 1 \quad i = 3 \quad i = 0$ mínimo

$3 \geq 1$

$2 \geq 1$

$1 \geq 1$

$0 \not\geq 1$

ent: $1 + 3 = 4$

$1 + 2 = 3$

$1 + 1 = 2$

$1 + 0 = 1$

me falla en 1.

$$i - \left(\frac{|N|}{2} - 1\right) = i - \frac{|N|}{2} + 1$$

$3 - 2 + 1 = 2$

$2 - 2 + 1 = 1$

$1 - 2 + 1 = 0$

$0 - 2 + 1 = -1$ me paré

Ejercicio 6. Dado el siguiente problema

proc `sumarElementos` (in s: $array < \mathbb{Z} >$) : $\mathbb{Z}$

requiere $\{|s| \geq 1\}$

asegura $\{res = \sum_{j=0}^{|s|-1} s[j]\}$

a) Corregir las siguientes implementaciones

b) Dar un invariante y función variante para cada una de estas implementaciones

a)
```
res := 0
i := 0      <
while (i > s.size()) do
        res := res + s[i];
        i := i + 1
endwhile
```

b)
```
res := 0
i := 0      <
while (i > s.size()) do
        res := res + s[s.size() - i];
        i := i + 1                      -i-1
endwhile
```

c)
```
res := 0
i := s.size() - 1 ]
while (i ≥ 0) do
        res := res + s[i];
        i := i - 1
endwhile
```

d)
```
res := 0
i := 0
while (i > s.size() / 2) do
        res := res + s[i] + s[s.size() - i];
        i := i + 1
endwhile
```

a) El error es el > xq si i empieza en 0

0 nunca es mayor a s.size().

Solución: < => suma todos los elem.

b) Hay dos errores, en vez de > en el ciclo es <.

Después, si i=0 s[s.size()] explota.

Sol: Reemplazar > por < y después

v.size() - i - 1.

c) $i = |v| - 1$ ✓

[1,2,3] $i = 2$

$0 = 0 + 3$ $i = 1$

$3 = 3 + 2$ $i = 0$

Acá termina, falta el 1.

El error es el >. Debería ser ≥

d) Como i::0 no entra nunca al ciclo pues

$0 \not> \frac{|v|}{2}$.

Reemplazar > por ≤ ya al ser la mitad de la lista no debería indefinirse.

Luego, en i=0 explota.

Está, v[v.size() - i - 1].

[1,2,3] Max $i = 1.5 \Rightarrow i = 1$

$0 = 0 + 1 + 3 = 4$ $i = 0$

$4' = 4' + \boxed{2+2} =$ $i = 1$

le salta

b) a) <

$I \equiv \{0 \leq i \leq |v| \wedge \sum_{j=0}^{i-1} v[j]\}$ $F_v = |v| - i$

b) b) El < y el -1.

$I \equiv \{0 \leq i \leq |v| \wedge \sum_{j=0}^{|v|-1} v[|v|-1+j]\}$

$F_v = |v| - i$

b) c) $I \equiv \{0 \leq i \leq |v| \wedge \sum_{j=0}^{|v|-1} v[|v|-1+j]\}$

$F_v = |v| + i$

b) d) NO NOS SALIÓ

Ejercicio 7. Considerando el siguiente Invariante:

$I \equiv \{0 \leq i \leq |s| \wedge (\forall j : \mathbb{Z})(0 \leq j < i \rightarrow_L ((j \text{ mod } 2 = 0 \wedge s[j] = 2 \times j) \vee (j \text{ mod } 2 \neq 0 \wedge s[j] = 2 \times j + 1)))\}$

- Escribir un programa en SmallLang que se corresponda al invariante dado.
- Defina las P_c, B y Q_c que correspondan a su programa.
- Dar una función variante para que se pueda completar la demostración.

a)

```
j := 0

WHILE (j < S.SIZE()) DO:

    IF (J MOD 2 = 0) THEN
        S[j] := 2 * j
    ELSE
        S[j] := 2 * j + 1
    ENDIF

    j := j + 1

ENDWHILE
```

b) $P_c = \{|N| > 0 \wedge i = 0\}$

$B = \{i < |N|\}$

$Q_c = \{i = |N| \wedge_L (\forall j : \mathbb{Z})(0 \leq j < i \rightarrow_L ((j \text{ MOD } 2 = 0 \wedge N[j] = 2 * j) \vee (j \text{ MOD } 2 \neq 0 \wedge N[j] = 2 * j + 1))$

$F_V = |N| - i$

Ejercicio 8. Considerando el siguiente Invariante:

$$I \equiv \{0 \leq i \leq |s|/2 \wedge (\forall j : \mathbb{Z})(0 \leq j < i) \rightarrow_L (s[j] = 0 \wedge s[|s| - j - 1] = 0)\}$$

- Escribir un programa en SmallLang que se corresponda al invariante dado.
- Defina las P_c, B y Q_c que correspondan a su programa.
- Dar una función variante para que se pueda completar la demostración.

```
i := 0

WHILE( i< S.SIZE()/2 ) DO:

    IF (N[i] = 0) THEN:
       N [S.SIZE() - i - 1] = 0
    ELSE
          SKIP
    ENDIF

    i := i + 1
 ENDWHILE
```

$P_C = \{ i = 0 \}$

$B = \{ i < |N|/2 \}$

$Q_C = \{ i = \frac{|N|}{2} \wedge (\forall j : \mathbb{Z}) ((0 \leq j < i) \rightarrow_L N[j] = 0 \wedge N[|N| - j - 1] = 0) \}$

$F_V = \frac{|N|}{2} - i$