Algoritmos y Estructuras de Datos I
Primer Cuatrimestre 2020

Guía Práctica 7
Ciclos a partir de invariantes

Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

Ejercicio 1.
Sea la siguiente especificación del problema de copiar una secuencia de enteros:

`proc copiarSecuencia` (in s: $seq\langle\mathbb{Z}\rangle$, out result: $seq\langle\mathbb{Z}\rangle$) {
`Pre` $\{True\}$
`Post` $\{s = result\}$
}

Sea la siguiente implementación incompleta de la función copiarSecuencia:

```
vector<int> copiarSecuencia(vector<int> s) {
    vector<int> r;
    int i = 0;
    while(i<s.size()) {
      ...
    }
    return r;
  }
}
```

Completar el programa (i.e. escribir el cuerpo del `while`) de forma que cumpla el siguiente invariante de ciclo:

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

Ejercicio 2.
Sea la siguiente especificación:

`proc incSecuencia` (inout s: $seq\langle\mathbb{Z}\rangle$) {
`Pre` $\{s = S_0\}$
`Post` $\{|s| = |S_0| \wedge_L (\forall i : \mathbb{Z})(0 \leq i < |s| \rightarrow_L s[i] = S_0[i] + 1)\}$
}

Sea la siguiente implementación incompleta de la función incSecuencia:

```
void incSecuencia(vector<int> &a) {
    int i = 0;
    while(...) {
      ...
    }
}
```

Completar el programa (i.e. escribir el cuerpo del `while` y su guarda) de forma que cumpla el siguiente invariante de ciclo:

$$I = 0 \leq i \leq |s| \wedge (\forall j : \mathbb{Z})(0 \leq j < i \rightarrow_L s[j] = S_0[j] + 1)$$

Ejercicio 3.
Sea la siguiente especificación del problema de retornar la cantidad de apariciones de un elemento en una secuencia de enteros:

`proc cantApariciones` (in s: $seq\langle\mathbb{Z}\rangle$, in e: $\mathbb{Z}$, out result: $\mathbb{Z}$) {
`Pre` $\{True\}$
`Post` $\{result = \#apariciones(s, e))\}$
}

Sea la siguiente implementación incompleta de la función cantApariciones:

```
int cantApariciones(vector<int> s, int e) {
    int r = 0;
    for(int i=0; ... ; ...) {
      ...
    }
    return r;
}
```

Completar el programa (i.e. escribir el cuerpo y la declaración del `for`) de forma que cumpla el siguiente invariante de ciclo:

$$I = 0 \leq i \leq |s| \wedge_L r = \#apariciones(subseq(s, 0, i), e))$$

Ejercicio 4.
Sea la siguiente especificación de un ciclo:

- $P_c : i = -1 \wedge s = S_0$
- $Q_c : |s| = |S_0| \wedge_L (\forall z : \mathbb{Z})(0 \leq z < |s| \rightarrow_L s[z] = S_0[z]^2)$

Dar dos implementaciones distintas que satisfagan la especificación del ciclo con el siguiente invariante:

$$I = (|s| = |S_0| \wedge -1 \leq i \leq |s| - 1) \wedge_L (\forall j : \mathbb{Z})(0 \leq j \leq i \rightarrow_L s[j] = S_0[j]^2) \wedge (\forall j : \mathbb{Z})(i < j < |s| \rightarrow_L s[j] = S_0[j])$$

Ejercicio 5.
Escribir un programa para el siguiente problema que respete la especificación y el invariante dado

`proc duplicarElementos` (inout s : $seq\langle\mathbb{Z}\rangle$) {
`Pre` $\{s = s_0 \wedge |s| \ mod \ 2 = 0\}$
`Post` $\{|s| = |s_0| \wedge_L$
$(\forall i : \mathbb{Z})(enRango(i, s) \longrightarrow_L s[i] = 2 * s_0[i])\}$

$I \equiv (|s| = |s_0| \wedge (0 \leq i \leq |s|/2)) \wedge_L$
$subseq(s, 0, |s| - 2 * i) = subseq(s_0, 0, |s_0| - 2 * i) \wedge$
$(\forall k : \mathbb{Z})(|s| - 2 * i \leq k < |s| \longrightarrow_L s[k] = 2 * s_0[k]))$

Ejercicio 6.
Escribir un programa para el siguiente problema que respete la especificación y contenga un ciclo el invariante dado

`proc dividirPorPromedio` (inout s : $seq\langle\mathbf{R}\rangle$) {
`Pre` $\{s = s_0 \wedge |s| \ mod \ 2 = 0 \wedge |s| > 0\}$
`Post` $\{|s| = |s_0| \wedge_L$
$(\forall i : \mathbb{Z})(enRango(i, s) \longrightarrow_L s[i] = \frac{s_0[i]}{promedio(s_0)})\}$
`aux promedio` (s : $seq\langle\mathbf{R}\rangle$) : $\mathbf{R} \ = \frac{\sum_{i=0}^{|s|-1} s[i]}{|s|}$;

$I \equiv (|s| = |s_0| \wedge 0 \leq i \leq \frac{|s|}{2}) \wedge_L$
$subseq(s, i, |s| - i) = subseq(s_0, i, |s_0| - i) \wedge$
$(\forall k : \mathbb{Z})(0 \leq k < i \longrightarrow_L s[k] = \frac{s_0[k]}{promedio(s_0)}) \wedge$
$(\forall k : \mathbb{Z})(|s| - i - 1 < k < |s| \longrightarrow_L s[k] = \frac{s_0[k]}{promedio(s_0)})$

Ejercicio 7.
Dar un programa que satisfaga la especificación y tenga un ciclo con el invariante:

`proc armarPiramide` (in v: $\mathbb{Z}$, inout l: $seq\langle\mathbb{Z}\rangle$) {
`Pre` $\{l = L_0\}$
`Post` $\{|L_0| = |l| \wedge esPiramide(l, v)\}$
`pred esPiramide` (l: $seq\langle\mathbb{Z}\rangle$, v: $\mathbb{Z}$) {

$(\forall j : \mathbb{Z})(0 \leq j < |l|/2 \Rightarrow_L l[j] = v + j) \wedge$
$(\forall j : \mathbb{Z})(|l|/2 \leq j < |l| \Rightarrow_L l[j] = v + |l| - j - 1)$

}
}

a. **Invariante:** $|l| = |L_0| \wedge |l|/2 \leq i \leq |l| \wedge_L ((i = |l|/2 \wedge l = L_0) \vee_L$
$(\exists p : seq\langle\mathbb{Z}\rangle)(esPiramide(p, v) \wedge |p| = |l| \wedge_L subseq(p, |l| - i, i) = subseq(l, |l| - i, i)))$

b. **Invariante:** $|l| = |L_0| \wedge 0 \leq i \leq |l| \wedge_L piramideHastaI(l, i, v)$
`pred piramideHastaI` (l: $seq\langle\mathbb{Z}\rangle$, i: $\mathbb{Z}$, v: $\mathbb{Z}$) {
$(\exists p : seq\langle\mathbb{Z}\rangle)esPiramide(p, v) \wedge |p| = |l| \wedge_L subseq(p, 0, i) = subseq(l, 0, i)$
}

Ejercicio 8.
Escribir un programa para el siguiente problema que respete la especificación y el invariante dado

`proc multiplicar` (inout s : $seq\langle\mathbb{R}\rangle$) {
`Pre` $\{s = s_0 \wedge |s| \ mod \ 4 = 0\}$
`Post` $\{|s| = |s_0| \wedge_L$
$(\forall i : \mathbb{Z})(0 \leq i < |s| \longrightarrow_L s[i] = 10 * s_0[i])\}$
}

$I \equiv (|s| = |s_0| \wedge \frac{|s|}{2} \leq i \leq |s|) \wedge_L esPar(i) \wedge$
$subseq(s, i, |s|) = subseq(s_0, i, |s_0|) \wedge subseq(s, 0, |s| - i) = subseq(s_0, 0, |s_0| - i) \wedge$
$(\forall k : \mathbb{Z})(\frac{|s|}{2} \leq k < i \longrightarrow_L s[k] = 10 * s_0[k]) \wedge (\forall k : \mathbb{Z})(|s| - i \leq k < \frac{|s|}{2} \longrightarrow_L s[k] = 10 * s_0[k])$

Ejercicio 9.
Escribir un programa para el siguiente problema que respete la especificación y el invariante dado

`proc cerearYsumar` (inout s : $seq\langle\mathbb{Z}\rangle$, inout suma : $\mathbb{Z}$) {

`Pre` $\{s = s_0 \wedge |s| \ mod \ 8 = 0\}$

`Post` $\{|s| = |s_0| \wedge_L ((\forall i : \mathbb{Z})(enRango(i, s) \longrightarrow_L s[i] = 0) \wedge suma = \sum_{i=0}^{|s|-1} s_0[i])\}$

}

$I \equiv (|s| = |s_0| \wedge 0 \leq i \leq |s|/4) \wedge_L$
$(subseq(s, 2 * i, |s| - 2 * i) = subseq(s_0, 2 * i, |s_0| - 2 * i) \wedge (\forall k : \mathbb{Z})(0 \leq k < 2 * i \longrightarrow_L s[k] = 0) \wedge$
$(\forall k : \mathbb{Z})(|s| - 2 * i \leq k < |s| \longrightarrow_L s[k] = 0) \wedge suma = \sum_{j=0}^{2*i-1} s_0[j] + \sum_{j=|s|-2*i}^{|s|-1} s_0[j])$

RESOLUCIONES.

Ejercicio 1 .

```
vector<int> copiarSecuencia(vector<int> s){
   vector<int> r;
   int i=0;
   while(i<s.size()){
      r.push_back(s[i]);
      i++;
   }
   return r;
}
```

.

Ejercicio 2 .

```
vector<int> incSecuencia(vector<int> &s){
   int i=0;
   while(i<s.size()){
      s[i]++;
      i++;
   }
}
```

.

Ejercicio 3 .

```
int cantApariciones(vector<int> s, int e){
   int r=0;
   for(int i=0; i<s.size();i++){
      r=r+(s[i]==e);
   }
   return r;
}
```

.

Ejercicio 4 .

```
void raizConEsei(vector<int> &s){
   int i=-1;
   vector<int> s0=s;
   while( i<s.size()-1){
      i++;
      s[i]=sqrt(s0[i]);
   }
   return;
}
```

.

Ejercicio 5 .

```
void duplicarElementos(vector<int> &s){
   int i=0;
   vector<int> s0=s;
   while( i<s.size()/2){
```

```
5        s[s.size()-2*i-1]=2*s0[s0.size()-2*i-1];
6        s[s.size()-2*i-2]=2*s0[s0.size()-2*i-2];
7        i++;
8     }
9     return;
10 }
```

.

Ejercicio 6 .

```
1  void dividirPorPromedio(vector<float> &s){
2     int i=0;
3     vector<float> s0=s;
4     int p=promedio(s0);
5     while( i<s.size()/2){
6        s[i]=s0[i]/p;
7        s[s.size()-i-1]=s0[s0.size()-i-1]/p;
8        i++;
9     }
10    return;
11 }
12
13 float promedio(vector<float> &s){
14    float res=0;
15    for(int i=0;i<s.size();i++){
16       res+=s[i];
17    }
18    res/=s.size();
19    return res;
20 }
```

.

Ejercicio 7

a) primer I.

```
1  void armarPiramide(int v,vector<int> &s){
2     int i=0;
3     vector<float> s0=s;
4     while( i<s.size()/2){
5        s[i]=i+v;
6        s[s.size()-i-1]=v+s.size()-i-1;
7        i++;
8     }
9     return;
10 }
```

b) segundo I.

```
1  void armarPiramide(int v,vector<int> &s){
2     int i=0;
3     while(i<s.size()/2){
4        s[i]=i+v;
5        i++;
6     }
```

```
    while(i<s.size()){
       s[i]=v+s.size()-i-1;
       i++;
    }
    return;
}
```