

**Programación 1**  
**Segundo Parcial**  
**Instituto de Computación - Facultad de Ingeniería**  
**Julio 2020**

**Soluciones**

**Ejercicio 1**

---

```
function palabraMasLarga (fra: TipoFrase): integer;
var i, largo, max : Integer;
begin
  i := 1; max := 0;
  repeat
    largo := 0;
    while (i <= N) and (fra[i] <> separa) do
      begin
        largo := largo + 1;
        i := i + 1
      end;
    if (largo > max) then max := largo;
    i := i + 1;
  until (i >= N - max);
  palabraMasLarga := max
end;
```

---

**Ejercicio 2**

**Parte a)**

---

```
(* si es vacia (i es 1 y tope es 0) es ordenada *)
function esSecuenciaOrdenada (sec: Secuencia): boolean;
var i : integer;
begin
  i := 1;
  while (i <= sec.tope - 1) and (sec.valores[i] < sec.valores[i+1]) do
    i := i+1;
  esSecuenciaOrdenada := i >= sec.tope
end;
```

---

**Parte b)**

---

```
procedure agregarValor (val: natural; var sec: Secuencia);
var i, j : integer;
begin
  i := 1;
  while (i <= sec.tope) and (sec.valores[i] < val) do
    i := i + 1;
  sec.tope := sec.tope + 1; (* hay espacio para agregar al menos uno *)
  for j := sec.tope downto i+1 do
    sec.valores[j] := sec.valores[j-1];
  sec.valores[i] := val
end;
```

---

---

```
procedure agregarValor (val: natural; var sec: Secuencia);
var i: integer;
begin
  sec.tope := sec.tope + 1; (* hay espacio para agregar al menos uno *)
  i := sec.tope - 1;
  while (i > 0) and (sec.valores[i] > val) do
    begin
      sec.valores[i+1] := sec.valores[i];
      i := i - 1
    end;
  sec.valores[i + 1] := val
end;
```

---

### Ejercicio 3

---

```
procedure buscarMinimo (lst: Lista; var resu: Resultado);
var min: Integer;
begin
  if lst = NIL then resu.vacia := true
  else
    begin
      min := lst^.elemento;
      lst := lst^.siguiente;
      while (lst <> NIL) do
        begin
          if (lst^.elemento < min) then
            min := lst^.elemento;
            lst := lst^.siguiente
          end;
          resu.vacia := false;
          resu.min := min
        end
      end
    end;
end;
```

---

### Ejercicio 4

<b>Entrada</b>	<b>Salidas</b>
----------------	----------------

0	8 0 4
1	10 1 5
2	12 2 6
3	14 3 7
4	16 4 8
5	18 5 9
6	20 6 10
7	22 7 11
8	24 8 12
9	26 9 13