

# Compiladors: Examen parcial de laboratori.

17 d'abril de 2020

**ATENCIÓ:** Cal entregar l'examen en un fitxer `.tgz` pujat al Racó. Llegiu les instruccions del final de l'enunciat per veure com generar-lo.

**ATENCIÓ:** Al Racó trobareu els jocs de proves i codi necessari per a fer l'examen. Llegiu les instruccions del final de l'enunciat per veure d'on descarregar-lo.

**PUNTUACIÓ:** Els tres primers punts de la nota de laboratori s'obtenen amb els jocs de proves de la pràctica base. La resta s'obtenen superant els jocs de proves específics de l'examen. La correcció és **automàtica**, a través dels jocs de proves d'aquest enunciat, més un conjunt addicional de jocs de proves privats.

**IMPORTANT:** L'examen consta de dos exercicis independents. Podeu fer-los en qualsevol ordre. Es recomana fer cada exercici incrementalment, resolent cada joc de proves abans de passar al següent.

## 1 Funció predefinida max (4 punts)

Volem afegir a l'ASL la funció predefinida `max`, que retorna el valor més gran dels paràmetres que rep, amb les següents característiques:

- La funció es pot aplicar a qualsevol nombre de paràmetres.
- Els valors passats com a paràmetre poden ser de tipus numèric o de tipus caràcter.
- Els tipus numèrics `int` i `float` poden barrejar-se en la mateixa crida, però no els valors `char` (és a dir, si un dels paràmetres és de tipus caràcter, tots ho han de ser).
- El resultat de la funció `max` serà del mateix tipus que els seus paràmetres (i `float` en cas de tipus numèrics barrejats).

Per exemple:

```
func main()
  var i,z : int
  var c,d : char
  var x : float

  x = max(34, z+1, 25*i/100.0, x/4);
  while x <= max(z, max(i*2, x)) do
    x = 3*x - i;
  endwhile

  c = max('a', 'A', d, '\n');
endfunc
```

**Joc de proves 1 (1 punts).** Començarem modificant només la gramàtica per afegir la funció predifinida `max` al llenguatge, permetent que accepti un nombre qualsevol de paràmetres

Un cop fets els canvis, el primer joc de proves:

```
1 func main()
2   var i,z : int
3   var c,d : char
4
5   z = max(34, z+1);
6   while 0.1 <= max(z, max(i*2, z+1)) do
7     c = 3*x - i;
8   endwhile
9
10  if z+3 then
11    c = max('a', 'A', d, '\n');
12  endif
13 endfunc
```

hauria de produir la sortida:

Line 7:7 error: Assignment with incompatible types.  
Line 7:11 error: Identifier 'x' is undeclared.  
Line 10:2 error: Instruction 'if' requires a boolean condition.

**Joc de proves 2 (1 punts).** Ara farem el Typecheck de la nova funció, considerant **de moment** que el tipus del resultat és igual al tipus del primer argument.

Un cop fets els canvis, el primer joc de proves:

```
1 func main()
2   var i,z : int
3   var c,d : char
4
5   z = max(34, z+1, 25*i/100, d/4);
6   while 0.1 <= max(z, max(i*2, z+1)) do
7     c = 3*x - i;
8   endwhile
9
10  z = max('a', d);
11  if z+3 then
12    c = max('a', 'A', d, '\n');
13  endif
14 endfunc
```

hauria de produir la sortida:

Line 5:30 error: Operator '/' with incompatible types.  
Line 7:7 error: Assignment with incompatible types.  
Line 7:11 error: Identifier 'x' is undeclared.  
Line 10:4 error: Assignment with incompatible types.  
Line 11:2 error: Instruction 'if' requires a boolean condition.

**Joc de proves 3 (1 punt).** A continuació afegirem una nova comprovació semàntica: La funció `max` ha de tenir com a mínim dos paràmetres.

Així arribem al segon joc de proves:

```
1 func main()
2   var i,z : int
3   var a : array [10] of int
4
5   z = max(34);
6   z = 2.5 * max(z+1, 25*i/100, a[i-1]);
7   z = max();
8   z = max(10, max(i*z/2, a[0]+22));
9   z = max(max(12,x));
10
11 endfunc
```

que ha de produir els errors:

```
Line 5:6 error: Incorrect number of arguments in funtion 'max'.
Line 6:4 error: Assignment with incompatible types.
Line 7:6 error: Incorrect number of arguments in funtion 'max'.
Line 9:6 error: Incorrect number of arguments in funtion 'max'.
Line 9:17 error: Identifier 'x' is undeclared.
```

**Joc de proves 4 (1 punt).** El següent pas serà comprovar que els tipus dels paràmetres són correctes i donar un error si no és el cas.

Amb aixó, el joc de proves:

```
1 func main()
2   var i,z : int
3   var a : array [10] of int
4   var x : float
5   var c,d : char
6
7   z = max(34);
8   z = max(z+1, 25*i/100, a[i-1]);
9   x = max(a[i*2-1], x+1);
10  z = max(10, i*z/2, a[0]+22);
11  x = max(12, x, z);
12  c = max('A', 'a', d);
13  c = max('A', 'a', x+1, d);
14  z = max(23, d, x+1);
15 endfunc
```

produeix la sortida:

```
Line 7:6 error: Incorrect number of arguments in funtion 'max'.
Line 13:6 error: Incompatible argument types in funtion 'max'.
Line 14:6 error: Incompatible argument types in funtion 'max'.
```

**Joc de proves 5 (1 punt).** Finalment, caldra assegurar-se que el tipus del resultat de la funció `max` ara es calcula segons **tots** els paràmetres rebuts:

Així passarem el darrer joc de proves:

```
1 func main()
2   var i,z : int
3   var a : array [10] of int
4   var x : float
5   var c,d : char
6
7   z = max(34);
8   z = max(z+1, 25*i/100, a[i-1]);
9   x = max(a[i*2-1], x+1);
10  z = max(a[i*2-1], x+1);
11  z = max(10, max(i*z/2, a[0]+22));
12  z = max(max(12,x), z);
13  c = max('A', 'a', d);
14  z = max('A', 'a', d);
15  z = max('A', 'a', x+1, d);
16  c = max('A', 'a', x+1, d);
17 endfunc
```

que escriu la sortida:

```
Line 7:6 error: Incorrect number of arguments in funtion 'max'.
Line 10:4 error: Assignment with incompatible types.
Line 12:4 error: Assignment with incompatible types.
Line 14:4 error: Assignment with incompatible types.
Line 15:6 error: Incompatible argument types in funtion 'max'.
Line 16:6 error: Incompatible argument types in funtion 'max'.
```

## 2 Estructura repetitiva for (3 punts)

El segon exercici consisteix en dotar el llenguatge ASL amb una estructura similar al `for` de Python.

Un exemple de codi:

```
func main()
  var i,j : int
  var a : array [10] of char
  var c : char

  for i in range(10) do
    a[i] = a[i+1];
  endfor

  if j > 0 then
    for i in range(0,5) do
      write(a[i]*2);
    endfor
  else
    for i in range(10,6,-1) do
      write(a[i]/2);
    endfor
  endif
endfunc
```

**Joc de proves 6 (1 punt).** El primer pas és afegir l'estructura del `for` a la gramàtica, de forma que s'accepti com qualsevol altra estructura de control.

El primer joc de proves:

```
1 func main()
2   var i,j: int
3   var a,b: array [10] of char
4   var c: char
5
6   for j in range(10) do
7     a[j] = f(a,i);
8   endfor
9   i = j*c;
10  for j in range(1,i) do
11    a[j-1] = f(b,i+1);
12  endfor
13 endfunc
14
15 func f(a: array [10] of char, i: int): char
16   var m: char
17   var j,k: int
18
19   a[m] = 3;
20   for i in range(1, 10, 2*j) do
21     for k in range(i+1, j) do
22       if m>a[k+i] then m=a[i+1];
23       else return a[i-1];
24     endif
25   endfor
26 endfor
27   return j>a;
28 endfunc
```

genera els errors:

```
Line 9:7 error: Operator '*' with incompatible types.
Line 19:4 error: Array access with non integer index.
Line 19:7 error: Assignment with incompatible types.
Line 27:2 error: Return with incompatible type.
Line 27:10 error: Operator '>' with incompatible types.
```

**Joc de proves 7 (1 punt).** Ara cal que el TypeCheck accepti només els nombres de paràmetres vàlids per al `range` (és a dir: només poden aparèixer un, dos o tres paràmetres).

El primer joc de proves:

```
1 func main()
2   var i,j: int
3   var a: array [10] of char
4   var c: char
5
6   for j in range(10) do
7     a[j] = f(a,i);
8   endfor
9   for j in range(1,i,z-3,j+3) do
10     a[j-1] = f(a,i+1);
11   endfor
12 endfunc
13
14 func f(a: array [10] of char, i: int): char
15   var m: char
16   var j,k: int
17
18   for i in range(1, 10, 2*j) do
19     for k in range(i+1, j) do
20       if m>a[k+i] then m = a[i+1];
21       else return a[i-1];
22     endif
23     k = k/2.0;
24   endfor
25 endfor
26   return -1;
27 endfunc
```

genera els errors:

```
Line 9:2 error: Incorrect number of 'for-range' expressions.
Line 9:21 error: Identifier 'z' is undeclared.
Line 23:3 error: Assignment with incompatible types.
Line 26:2 error: Return with incompatible type.
```

**Joc de proves 8 (1 punt).** A continuació cal comprovar que la variable de control del bucle sigui de tipus enter

Usarem el següent codi:

```
1 func main()
2   var i,j: int
3   var a,b: array [10] of char
4   var z: float
5
6   for z in range(10) do
7     a[j] = f(b, a[j+2*i]);
8   endfor
9 endfunc
10
11 func f(a: array [10] of char, i: int): int
12   var m: char
13   var j,k: int
14
15   for m in range(1, 10, 2*j) do
16     for k in range(i+1, j) do
17       if m>a[k+i] then m=a[i+1];
18       else return a[m-1];
19     endif
20   endfor
21 endfor
22   return k;
23 endfunc
```

que ha de donar la sortida:

```
Line 6:6 error: Integer type required in 'for' control variable.
Line 7:11 error: Assignment with incompatible types.
Line 7:18 error: Parameter #2 with incompatible types in call to 'f'.
Line 15:6 error: Integer type required in 'for' control variable.
Line 18:13 error: Return with incompatible type.
Line 18:23 error: Operator '-' with incompatible types.
```



**Joc de proves 9 (1 punt).** El darrer pas serà comprovar que els paràmetres del **range** són també enters.

El codi d'aquest joc de proves:

```
1 func main()
2   var i,j: int
3   var a: array [10] of char
4   var z: float
5
6   for z in range(10, a[j]) do
7     a[j] = f(c,i);
8   endfor
9 endfunc
10
11 func f(a: array [10] of char, i: int): int
12   var m: char
13   var j,k: int
14
15   for m in range(1, a, 2*j) do
16     for q in range(i+1, 10) do
17       if m>a[k+i] then m=a[i+1];
18       else return a[m-1];
19     endif
20   endfor
21 endfor
22   return k;
23 endfunc
```

ha de generar els errors:

```
Line 6:6 error: Integer type required in 'for' control variable.
Line 6:21 error: Integer types required in 'for-range' expressions.
Line 7:11 error: Assignment with incompatible types.
Line 7:15 error: Identifier 'c' is undeclared.
Line 15:6 error: Integer type required in 'for' control variable.
Line 15:20 error: Integer types required in 'for-range' expressions.
Line 16:9 error: Identifier 'q' is undeclared.
Line 18:13 error: Return with incompatible type.
Line 18:23 error: Operator '-' with incompatible types.
```

## Informació important

**FITXERS PER A L'EXAMEN:** Al Racó (`examens.fib.upc.edu`) trobareu un fitxer `examen.tgz` amb el següent contingut:

- `parcial-lab-CL-2020.pdf`: Aquest document, amb l'enunciat i les instruccions.
- `jps`: Subdirectori amb jocs de proves (`jp_chkt_XX.asl`) i la seva corresponent sortida esperada (`jp_chkt_XX.err`)
- `common/SemError.*`: Mòdul d'errors semàntics ampliat amb els errors necessaris per a l'examen.
- `avalua.sh`: Script que executa tots els jocs de proves i diu si se superen o no.
- `empaqueta.sh`: Script que crea un fitxer `examen-nom.cognom.tgz` amb la vostra solució. Aquest és el fitxer que cal pujar al Racó.

### PASSOS A SEGUIR:

- Feu una còpia de les carpetes `asl` i `common` de la vostra pràctica a un directori `examen`.  

```
mkdir examen
cp -r practica/asl practica/common examen/
```
- Canvieu al nou directori `examen`, i descomprimiu-hi el fitxer `examen.tgz` del Racó:  

```
cd examen
tar -xzf examen.tgz
```

Això extreurà el contingut del paquet, **afegint** al vostre directori `examen` els fitxers llistats anteriorment.

**IMPORTANT:** Feu-ho en l'ordre especificat (primer una còpia de la vostra pràctica i després descomprimir el `.tgz`). Fer-ho en l'ordre invers causarà que us falti codi necessari a `common` i que els JPs no siguin els adequats.
- Treballeu normalment a la carpeta `examen/asl`.  

```
cd asl
make antlr
make
...
```
- Per veure les diferències entre la sortida del vostre `asl` i la sortida esperada en un joc de proves, podeu fer:  

```
./asl ../jps/jp_chkt_XX.asl | diff -y - ../jps/jp_chkt_XX.err
```

(Podeu ignorar la línia "There are semantic errors: no code generated" que genera el `main`)
- Per executar tots els jocs de proves i veure si els passeu, executeu `../avalua.sh`.
- Executeu `../empaqueta.sh` per crear el fitxer d'entrega `../examen-USERNAME.tgz` que cal pujar al Racó. Els paquets creats sense usar aquest script seran qualificats com **NO PRESENTAT**.