Práctica 1

Definición de un lenguaje de programación básico: "miLenguaje"

1. Objetivos

1. Determinar los elementos fundamentales de un lenguaje de programación imperativo.

2. Definir un lenguaje propio de programación.

2. Contenidos

Como resultado de esta primera práctica (para ser desarrollada en una única sesión de laboratorio) cada grupo debe definir y proponer su propio lenguaje de programación, denominado en todos los casos "miLenguaje". El lenguaje será manejado y ampliado durante el resto de las prácticas.

Este lenguaje debe cumplir una serie de requisitos que se van a enumerar a continuación. Éstos no son muchos, pero son suficientes para trabajar con los conceptos fundamentales del diseño de un traductor.

Las características *mínimas* que debe cumplir "miLenguaje" son las siguientes:

1. Maneja, como mínimo, tres tipos de datos escalares básicos: carácter, enteros y booleanos. Más adelante se establecen cuáles son los operadores asociados que debe soportar.

2. Además de los anteriores, el lenguaje debe permitir el uso de constantes de tipo cadena, pero sólo a efectos de escritura por la salida estándar.

Las constantes de tipo cadena se representan entre comillas dobles: “hola, caracola”, “H”, etc. Cuando dentro de una cadena se necesite el carácter doble comilla, éste se indicará mediante dos dobles comillas seguidas, de manera que la ejecución de

```
put("Dijo ""No me lo creo""")
```

imprimiría por stdout

```
Dijo "No me lo creo"
```

3. El lenguaje debe disponer de un constructor de tipos indexados, análogo al ARRAY de Ada, cuyo rango se establece exclusivamente mediante la construcción `v1..v2`, siendo `v1` y `v2` constantes de tipo escalar.
4. El lenguaje dispone de una instrucción de asignación a una variable escalar.
5. El lenguaje dispone de procedimientos y funciones.
6. El lenguaje debe disponer de una operación de escritura de expresiones escalares hacia la salida estándar y de lectura de datos escalares desde la entrada estándar.
7. Como estructuras de control debe tener, además de la composición secuencial, al menos una instrucción de selección y una de iteración.
8. El lenguaje no hace distinción entre mayúsculas y minúsculas en identificadores o palabras reservadas, aunque sí en el caso de valores constantes de tipo cadena o carácter.
9. Los comentarios son igual que en el caso de Ada: empiezan con `--` y se terminarán al final de la misma línea.
10. Todas aquellas cuestiones de definición del lenguaje que no queden claras deben consultarse con el profesor.

3. Resultados

Durante la primera sesión, cada grupo tiene que definir las características de su propio lenguaje. Como resultado cada grupo deberá someter, desde la cuenta del alumno de menor NIP del grupo, el fichero denominado `pract1.tar`. Este fichero se obtendrá a partir de la instrucción `tar` de UNIX (ejecutar `man tar`), de manera que la ejecución de la instrucción `tar xvf pract1.tar` genere un directorio denominado `pract1` con:

- el fichero de texto `descipcionDeMiLenguaje` con un informe de los detalles concretos del lenguaje propuesto. Como aspectos fundamentales se deberán comentar los siguientes:
  - cómo son los identificadores permitidos
  - si el lenguaje permite o no el uso de tipos de datos sin nombre
  - si el procedimiento de salida permite una sola expresión o una lista de expresiones
  - si el procedimiento de entrada permite la lectura de una sola variable o de una lista de variables
  - si está permitida la lectura de variables (escritura de expresiones) de tipo booleanos
  - cómo son las formas de paso de parámetros permitidas, tanto para procedimientos como para funciones
  - si se permite que haya procedimientos y funciones anidados
  - cómo son las expresiones permitidas
  - describir cuáles podrían ser los errores léxicos más frecuentes y proponer para ellos políticas de recuperación
- el directorio `bancoDePruebas`, que debe contener 5 programas escritos en el lenguaje definido. Por unificar, un programa fuente escrito en "miLenguaje" llevará el sufijo `.ml`. La especificación de cada uno de los programas es la siguiente:
  - `eratostenes.ml`
    Pide por entrada estándar un natural `0<n<=100` y muestra por la salida estándar todos los números primos menores o iguales que `n`, aplicando el método denominado "criba de Eratóstenes"

- `factorial.ml`
  Pide por entrada estándar un natural `0<n` y muestra por la salida estándar `n!`
- `intercambio.ml`
  Pide por entrada estándar dos números enteros, invoca al procedimiento `intercambia`, y escribe el resultado por la salida estándar. El procedimiento `intercambia` toma dos parámetros enteros y permuta los valores de las variables con que se invoca.
- `seleccionDirecta.ml`
  Toma un secuencia de datos de la entrada estándar, los ordena mediante el método de selección directa, y los escribe ordenados por la salida estándar
- `meLoHeInventado.ml`
  Este quinto ejemplo debe ser de cosecha propia (puede ser vacío) para destacar aquellos aspectos "especiales" del lenguaje que merezcan la pena ser resaltados

4. Plazo de entrega del trabajo

Esta práctica será revisada en la segunda sesión de laboratorio.

5. Cómo entregar el material pedido en las prácticas

Las prácticas de esta asignatura se harán sobre le máquina Merlin. El directorio `/users2/COMPI` será utilizado para las prácticas de esta asignatura. Dentro de él hay dos directorios, denominados `entradas` y `salidas`.

5.1. El directorio entradas

Contiene un directorio por cada grupo de prácticas de la asignatura, coincidiendo con el *username* en *Merlin* con menor NIP. Es allí donde quedarán depositados los ficheros que cada alumno someta al realizar sus prácticas. La forma de someter un fichero se describe más adelante. El estudiante cuyo

username coincida con el subdirectorio podrá únicamente hacer `ls` con el fin de comprobar cuál es el contenido de dicho subdirectorio, pero nada más.

5.2. El directorio salidas

En el directorio `salidas` los profesores irán depositando ficheros necesarios para el desarrollo de las prácticas (información, bibliotecas necesarias, enunciados de las prácticas en formato PostScript o pdf, etc.).

5.3. Para entregar los resultados de las prácticas

Existe una utilidad, denominada `someter`, que es la encargada de depositar los ficheros a entregar en el lugar adecuado. La invocación requiere dos parámetros: el primero, el nombre correspondiente a la asignatura ("COMPI" en este caso); el segundo, el nombre del fichero a someter. Un ejemplo de uso sería el siguiente. Supongamos que hay que entregar el fichero `pract1.tar`. La invocación sería:

```
$someter COMPI pract1.tar
```

El programa se encarga de depositar una copia del fichero `pract1.tar` en el directorio de entrega (dentro de `entradas`) del alumno que lo invoca (es decir, en el directorio cuyo nombre coincida con el username de la cuenta desde que se invoca). La invocación sin parámetros recuerda cómo ha de ser la invocación correcta. Tened presente que, una vez sometido un fichero, no puede volver a ser "resometido". En caso de necesidad, contactad con el profesor.

Es preciso tener presentes las siguientes consideraciones:

- Los nombres de los ficheros que se entregan deben coincidir exactamente con los nombres que se piden en los enunciados de las prácticas (tened presente que UNIX distingue mayúsculas de minúsculas).
- Es aconsejable mirar el contenido de `salidas` cuando se vaya a trabajar en alguna práctica; es posible que se deje información de última hora. Por ejemplo, para la práctica 1 puede haber información adicional en un fichero denominado `pract1LEEME`.

- Cualquier duda sobre el lenguaje debe ser preguntada a los profesores de la asignatura. Es interesante que para cuestiones que queráis comentar con algún profesor enviéis un mensaje a la dirección de correo electrónico correspondiente (aunque, preferentemente, usad la interacción directa en las clases de prácticas y en las tutorias: cuesta menos hablar que escribir).

- Los ficheros que tengan el sufijo `ps` corresponden al formato PostScript, y deben ser visualizados utilizando una aplicación específica. En *Merlin* podéis usar el visualizador "ghostview", ejecutando:

```
$/usr/local/bin/X11/ghostview nombreArchivo&
```

- Los ficheros que tengan el sufijo "pdf" corresponden al formato PDF, y deben ser visualizados utilizando una aplicación específica. En *Merlin* podéis usar el visualizador "acroread", ejecutando:

```
$/opt/acroread/bin/acroread nombreArchivo&
```

5.4. Un último comentario

Una vez que una práctica ha sido entregada, y salvo justificadas excepciones, no podrá volver a ser sometida. Esto hace necesario que el material que se someta esté cuidadosamente revisado (programas funcionando correctamente en *Merlin*, fuentes adecuadamente comentados, ficheros `Make` con los *paths* correctos, comentarios sin faltas de ortografía, etc.).