

12.1 - PATH ¹

Los comandos se buscan en los directorios indicados en la variable `PATH`.

```
cd ; cd tso/01/progs
echo $PATH # 100
make fi01
ls -l fi01 # 120
fi01
./fi01 # 140
bash
export PATH=$PATH:. # 160
fi01
exit # 180
fi01
export PATH=$PATH:. # 200
fi01
```

Vemos que `fi01` se puede invocar sin anteponer `./` si nos ayuda la variable `PATH`.

Un poco mas:

```
cd
mkdir -p bin # 280
cp tso/01/progs/fi01 bin
ls # 300
fi01
export PATH=$PATH:$HOME/bin # 320
whereis fi01
fi01 # 340
```

12.2 - permisos ²

Preparamos el directorio `mio`, (hasta línea 400), observamos los permisos, y comprobamos que podemos leer el fichero `dias` y escribir en el mismo.

```
cd ; cd tso/01/mio ; rm *
cp ../palabras/* . # 400
ls -l
cat dias
echo ===== >> dias ; cat dias # 420
echo no tengo > dias ; cat dias
```

Cambiamos un permiso del fichero `meses`, escritura, lo vemos (línea 440), y comprobamos que podemos leer pero no podemos escribir.

```
chmod u-w meses
ls -l meses
cat meses # 440
echo ===== >> meses ; cat meses
echo no tengo > meses ; head -3 meses
```

¹apuntes SSAA, cap. 21, pag. 230-231

²apuntes SSAA, cap. 15.2, pag. 116-118

Preparamos un *escript* o programa del *intérprete de comandos*. (hasta línea 500). Le ponemos permiso de ejecución. Vemos que se puede invocar anteponiendo `./` a su nombre cuando estamos en su directorio.

```
echo head -3 dias    > di-mes
echo tail -3 meses  >> di-mes          # 500
ls -l di-mes
chmod u+x di-mes
ls -l di-mes
./di-mes
chmod u+x di-mes
./di-mes          # 540
di-mes
```

Si tenemos el directorio actual en la variable `PATH`, basta con el nombre.