

# Guía de Ejercicios 8 - Argumentos en Línea de Comandos

## Advertencia

La resolución conjunta o grupal de los ejercicios aquí presentes no está permitida, excepto en la medida en que puedas pedir ayuda a tus compañeros de clase y a otras personas, y siempre que esa ayuda no se reduzca a que otro haga el trabajo por vos.

El código fuente entregado por un estudiante debe ser escrito en su totalidad por dicha persona.

## Condiciones de entrega:

¿Qué se entrega?	¿Qué no se entrega?
Archivos fuente/source (.c)	Archivos objeto (.o)
Archivos encabezado/header (.h)	Archivos ejecutables (programa, app, a.out, etc.)
Bibliotecas específicas (.a)	

Se deben entregar los ejercicios en un archivo zip (usar template como ayuda para el formato).

**Importante:** Recordá validar **siempre** que se reciben la cantidad de argumentos esperados. En caso contrario, el programa finalizará sin efectuar operación alguna y retornará el valor **1**.

## Ejercicio 8.1

Desarrollá un programa que imprima por pantalla todos los argumentos recibidos, uno por línea.

## Ejercicio 8.2

Desarrollá un programa que reciba como argumento un string e imprima su longitud, a la **strlen()**.

## Ejercicio 8.3

Desarrollá un programa que reciba como argumentos dos strings e imprima el resultado de su comparación lexicográfica, como lo haría la función **strcmp()** o **strcasecmp()**.

## Ejercicio 8.4

Desarrollá un programa que reciba como argumentos dos números enteros e imprima el resultado de la operación módulo entre ambos.

## Ejercicio 8.5

Desarrollá un programa que reciba como argumento un número entre el 30 y el 99 y presente por pantalla el número en letras. Recordá validar que el número se encuentre en ese rango.

## Ejemplo:

```
$ ./numbers 47
47: Cuarenta y siete
```

## Ejercicio 8.6

Desarrollá un programa que imprima los argumentos recibidos pero teniendo en cuenta que si el primer parámetro del programa es **-r**, entonces los deberá mostrar en orden inverso.

## Ejemplo:

```
$ ./printer Ser o "no ser," "esa es" "la cuestion"
Ser
o
no ser,
esa es
la cuestion
$ ./printer -r Ser o "no ser," "esa es" "la cuestion"
la cuestion
esa es
no ser,
o
Ser
```

## Ejercicio 8.7

Desarrollá un programa que al recibir como argumento el string **IP:PUERTO**, imprima los strings **IP** y **PUERTO** en líneas diferentes.

**Ejemplo:**

```
$ ./network 172.16.4.205:1234
La IP es: 172.16.4.205
El puerto es: 1234
```

## Ejercicio 8.8

Desarrollá un programa que genere muestras aleatorias en un intervalo dado por el usuario. El programa debe recibir como argumentos el inicio del intervalo,  **$X_i$** , el fin del intervalo,  **$X_f$** , y la cantidad de muestras a generar,  **$n$** .

**Ejemplo:**

```
$ ./sampler -3 2.65 10
2.0284
1.5767
0.4812
-1.1543
-0.7098
-0.1651
2.0144
0.2328
0.2302
1.3886
```

## Ejercicio 8.9

Desarrollá un programa que reciba como argumento un string e imprima por pantalla el string cifrado utilizando el algoritmo **ROT13**.

**Ejemplo:**

```
$ ./cipher hola
ubyn
```

## Ejercicio 8.10

En este ejercicio, la función de cada argumento estará dada por las banderas (**flags**) en la línea de comandos. Desarrollá una versión modificada del programa del ejercicio 8.8 para que reciba los argumentos utilizando las siguientes banderas:

- **-a** comienzo del intervalo
- **-b** fin del intervalo

- **-n** cantidad de muestras
- **-h** muestra un mensaje de ayuda, sin realizar ninguna otra función

Las banderas deben poder aparecer en cualquier orden, por lo que las siguientes invocaciones del programa son equivalentes:

```
$ ./sampler -a -3 -b 2.65 -n 10
2.0422
1.6900
-1.0428
-0.3802
-1.7582
-1.3972
-1.5386
-0.0568
-2.3125
-2.6120
$ ./sampler -b 2.65 -a -3 -n 10
...
$ ./sampler -n 10 -b 2.65 -a -3
...
$ ./sampler -h
NOMBRE
    sampler - genera lotes de números pseudo-aleatorios

SINOPSIS
    sampler [-a A] [-b B] [-n N] [-h]

DESCRIPCIÓN
    Genera un lote de N números pseudo-aleatorios en el intervalo
    [A,B). La distribución muestreada es U[A,B).

OPCIONES
    h
        muestra esta ayuda.
    a A
        comienzo del intervalo. Valor por omisión: 0.0.
    b B
        fin del intervalo. Valor por omisión: 1.0.
    n N
        cantidad de muestras a tomar. Valor por omisión: 100.

AUTORES
    Ada Lovelace, Alan Turing

LICENCIA
    GNU General Public License v3.0
```

## Referencias

Algunos ejercicios fueron obtenidos y adaptados de:

- Guía de Trabajos Prácticos 2011 - Informática I - Departamento de Electrónica - UTN FRBA
- Guía de Ejercicios - Algoritmos y Programación I - UBA FIUBA