#2

Batch

Tutor: **La Muerte Blanca**

Taller de programación en Batch
Parte II

Comando SET

Operacones con Variables

Operaciones Combinadas

Ejercicios

Condiciones en Batch

Comandos: Prompt Title Color

Errores Típicos

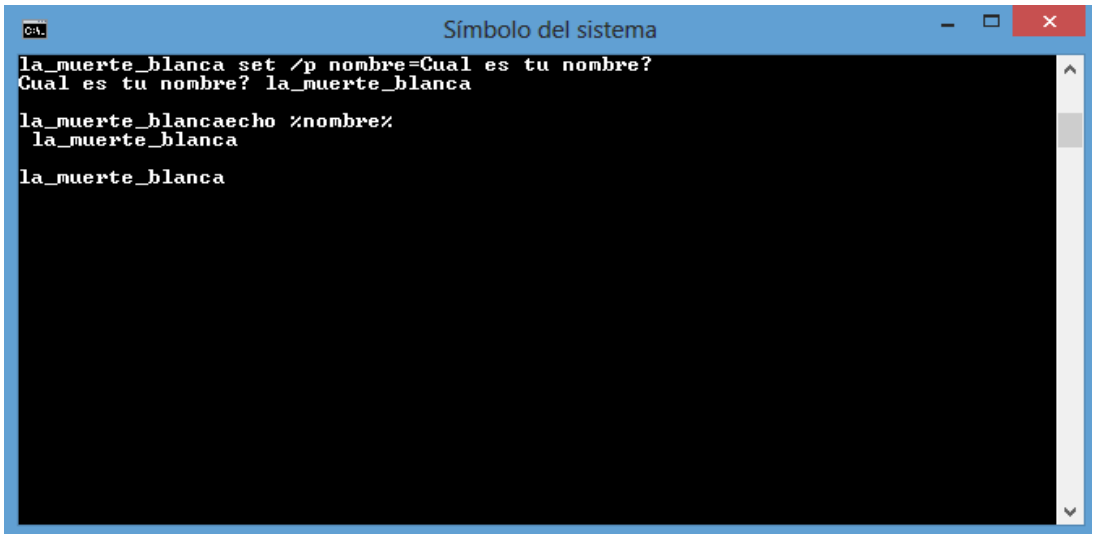
UNDERCODE

Esta es la segunda parte del taller de **Batch,** recomendamos leer la primera entrega debido a que es necesario saber/recordar algunas instrucciones allí dichas para poder seguir y comprender mejor al presente, comencemos:

COMANDO SET (continuación)

Como pudimos ver en el taller anterior, el comando **set** se utiliza para definir variables, pero imaginemos que creamos un programa y queremos que un usuario pueda asignar el valor que él quiera a una variable, por ejemplo, para preguntarle su nombre.

¿Cómo podríamos hacerlo? Pues es muy sencillo, simplemente debemos escribir **set /p "variable"=Cual es tu edad?**



Observemos la imagen y el comando que hemos escrito **/p** ¿para qué sirve esto? Pues **/p** después de **set,** nos permite pedir al usuario que escriba el valor de la variable, en este caso, el valor de la variable "nombre".

¿Para qué nos puede ser útil utilizar este comando? Este comando nos puede ser útil para múltiples usos, por ejemplo: si queremos crear un programa de felicitaciones, esto es, cuando un usuario introduzca su nombre, la consola devuelva: Feliz cumpleaños "Nombre".

Una duda que se suele plantear, es en caso de que escriba **set/p** sin el espacio ¿funcionaría? La respuesta, es afirmativa.

No importa si **/p** esta junto a **set** o separado por 900 espacios, seguirá funcionando. La razón de la separación con un espacio es para mayor legibilidad a la hora de examinar o leer ese código; pero de todas formas, les invito a probarlo en su consola de comandos o si prefieren, desde un documento de notas y ejecutar el .bat .

Dado que la finalidad de este taller es ayudar en la adquisición de conocimientos sobre Batch, dejaremos ejercicios para hacer por cada comando explicado.

NOTA: Si bien les invitamos a practicar e intentar realizar los ejercicios por ustedes mismos, todos ellos se encuentran resueltos al final del documento.

Ejercicio 1: Anteriormente, comentamos sobre crear un programa que pida al usuario su nombre. Este primer ejercicio consiste en pedir al usuario su nombre y que la consola devuelva: Feliz cumpleaños "nombre del usuario".

OPERACIONES CON VARIABLES

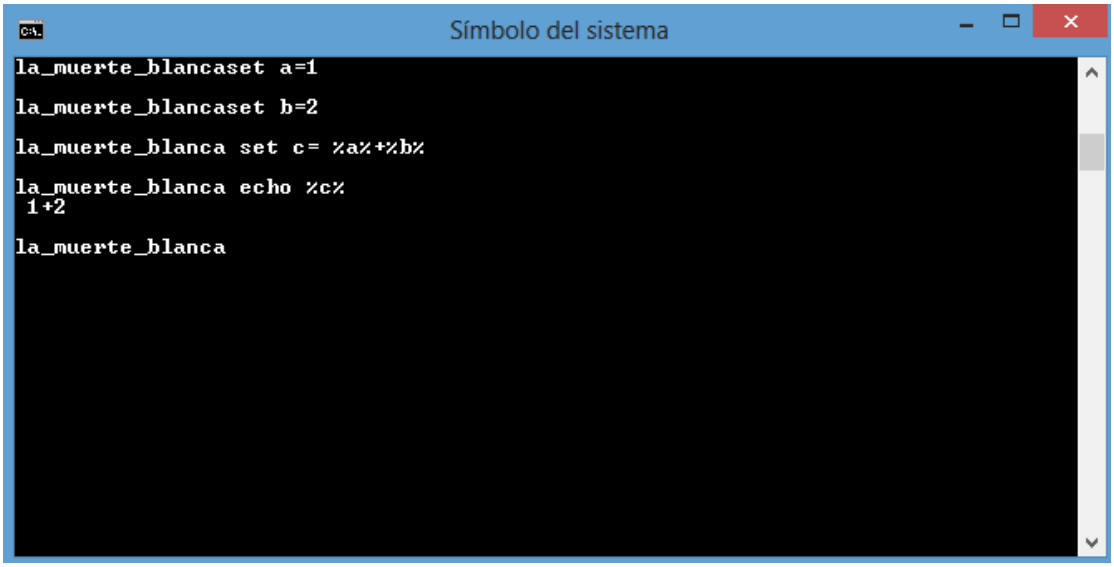
Seguramente, los que ya tendrán experiencia en programar pero que no han usado Batch, pensarán que sumar será igual o parecido a otros lenguajes, y si bien es cierto que existen similitudes, hay que evitar un error que se suele cometer y ahora mismo comentaremos; pero antes preparemos la operación:

Primero creamos una variable, en este caso le pondremos el nombre de "a" y le asignaremos el valor de 1.

Después creamos otra variable, en este caso con el nombre de "b" y le asignamos el valor de 2.

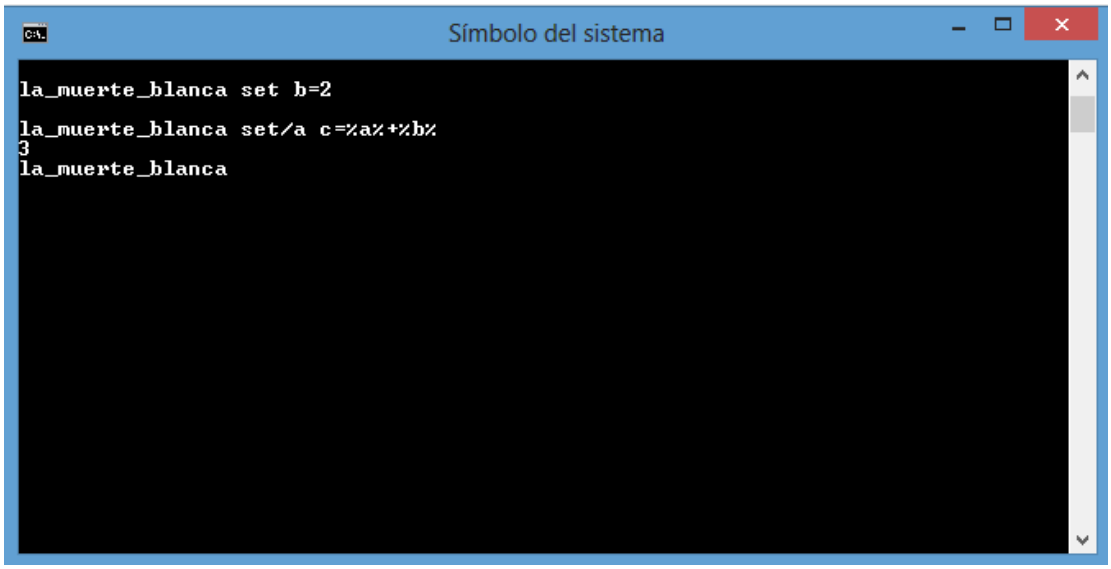
Ahora creamos una tercera variable, en este caso con el nombre de "c" y escribimos **%a% + %b%** y finalmente escribimos **echo %c%**

¿Ven lo que ocurre? No se ha realizado la operación y pasamos a explicar las razones.



Como habrán podido observar en la imagen, el resultado ha sido "1+2". Se preguntarán porque no se han sumado los valores contenidos en las variables (a y b) y la respuesta es muy sencilla: no le hemos indicado a la consola que queríamos realizar una operación y por lo tanto ha tomado el **"+"** como un string (cadena).

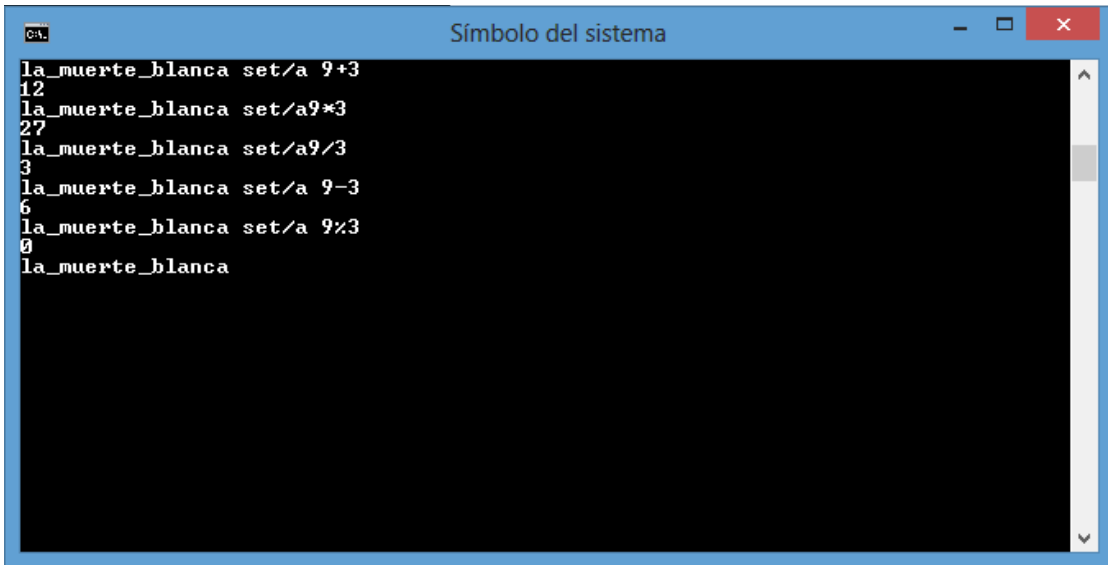
Ahora, se preguntarán: ¿Cómo le digo a la consola que quiero realizar una operación? La instrucción se la damos usando el comando **/a** de esta manera: **set/a c = %a% + %b%**



¡Mirad! Lo hemos conseguido, ahora la suma ha funcionado y todo gracias a la orden **/a** que se encarga de "instruir" a la consola que queremos realizar una operación.

Una duda que surge al ver esto es ¿qué puedo hacer si quiero realizar una operación sin utilizar variables?

Simplemente debes escribir **set/a** "operación". En la imagen siguiente, un pequeño ejemplo:



En la imagen precedente pudimos ver un ejemplo del uso de **/a** y las operaciones. Explicaremos sus signos, aunque seguramente ya los conocen todos y también pueden utilizarlos para hacer operaciones seguidas en Batch:

+ ➔Este signo sirve para sumar.

- ➔Este signo sirve para restar.

/ ➔Este signo, al igual que en la calculadora, sirve para dividir.

*** ➔**Este signo sirve para multiplicar.

% ➔Este signo, a diferencia de los demás, cabe la posibilidad de que no lo conozcan y su uso (aparte de identificar las variables) es para ver el resto de un número entre otro, fijándonos en la imagen **9%3=0** ya que la división de 9/3 da de resto 0.

Esta parte es muy sencilla, por lo tanto dejaremos unos ejercicios para que practiquen y lo comprueben ustedes mismos.

Ejercicio 2: Crear un programa que pida al usuario dos variables:"numero" y "otronumero" y muestre con el comando echo el resultado de la suma, la multiplicación y el resto del primero por el segundo.

Ejercicio 3: Para que repasen algunas cosas que aprendieron en el primer taller, este ejercicio tratará de crear un bucle que empieza por 1 y se sume 1 infinitamente.

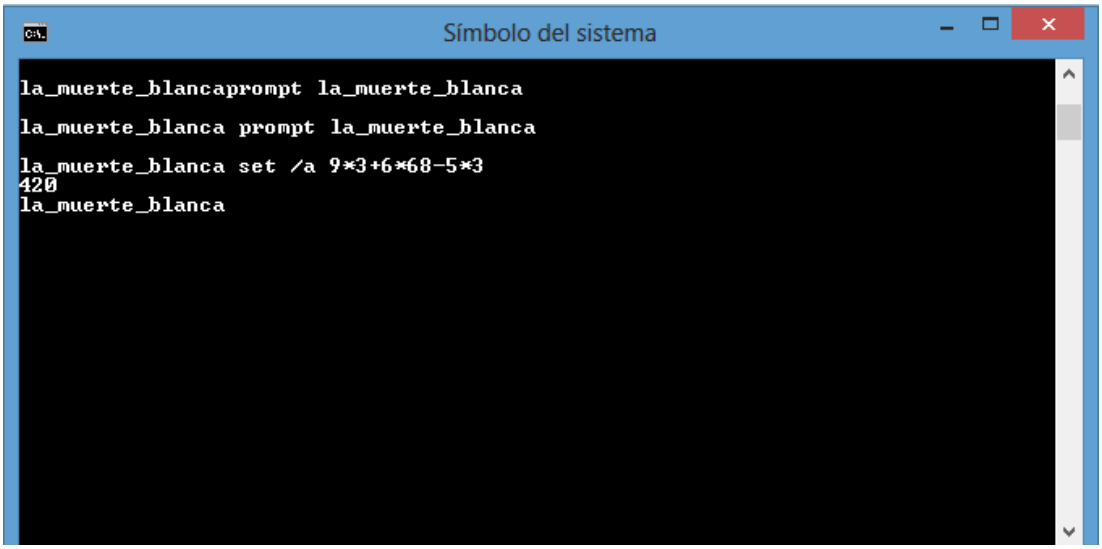
No se preocupen si no les sale el ejercicio, ya aprenderán con la práctica y verán que es muy fácil.

OPERACIONES SEGUIDAS EN BATCH

Posiblemente se preguntarán ¿puedo hacer operaciones seguidas en Batch, por ejemplo: 9+3*2?

La respuesta es sí...ya pueden ir tirando las calculadoras.

Comprobaremos su funcionamiento usando una operación larga, por ejemplo: 9*3+6*68-5/3



Fijaos, podría ser que realizara las operaciones según el orden en el que aparecen dispuestas; sin embargo, no es así. Este lenguaje realiza las operaciones en orden de prioridad: primero potencias y restos (%), después multiplicaciones y divisiones, y finalmente restas y sumas.

A continuación, unos ejercicios que podéis hacer para ver el buen funcionamiento de Batch en operaciones

matemáticas:

Ejercicio 4: Tendrán que crear una operación matemática con una multiplicación, tres divisiones, cinco sumas y ejecutarla para comprobar que Batch lo resuelve correctamente.

Ejercicio 5: Consiste en pedirle al usuario tres números, uno para cada variable (la variable numero1, numero2 y numero3) y que multiplique numero1 por numero2 y divida el resultado por numero3.

CONDICIONALES EN BATCH (continuación)

En el anterior taller vimos una introducción al uso de **if** y **else**, pero en esta parte vamos a seguir profundizando su uso.

If y **else** no solamente sirven para determinar si existe un archivo o no y ejecutar algún comando, sino que se pueden utilizar como comparadores.

Comenzaremos aprendiendo la comparación con números:

Primeramente utilizaremos **==** (No confundir con el signo igual, estos son dos iguales seguidos y se utilizan para ver si un número es igual a otro).

Escribiremos para empezar dos comparaciones:

Primera comparación: if 9==3 echo hola

Como podemos comprobar no devolverá "hola", debido a que el numero 9 no es igual al número 3.

Segunda comparación: if 9==9 echo hola

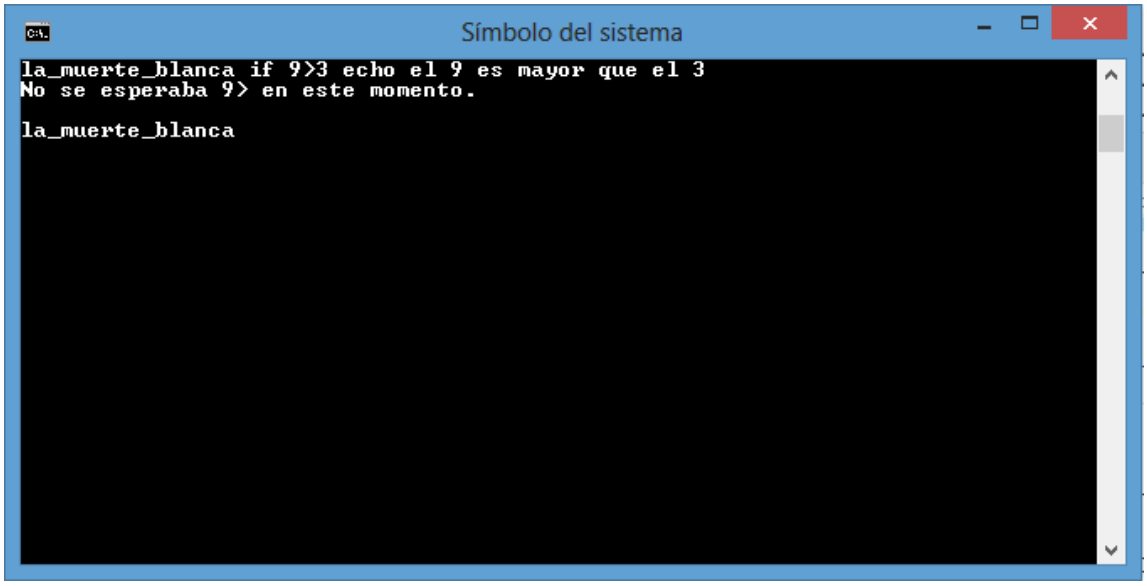
En este caso, sí devolverá “hola”, ya que el número 9 es igual a 9.

Ahora viene el gran fallo que suele tenerse si se ha programado en otros lenguajes. Para poder verlo y que no les pase cuando programen en Batch ¿qué mejor que probarlo?

Probaremos con una comparación de mayor:

if 9>3 echo El 9 es mayor que el tres

¿Qué ha pasado? Lo vemos en la siguiente imagen.



La respuesta es muy simple, en Batch los signos **>** (mayor que), **<** (menor que), **>=** (mayor o igual que), **<=** (menor o igual que) **NO** se utilizan.

¿Entonces, cómo puedo hacer este tipo de comparaciones en Batch? Vamos a tener que aprender algunas instrucciones:

== ➔Se encarga de verificar si dos números son iguales.

GTR ➔Esta vez no es un signo raro, si no letras...Se encarga de comprobar que el primer número es mayor que el segundo.

LSS ➔Se encarga de comprobar que el primer número sea menor que el segundo.

GEQ ➔Se encarga de comprobar que el primer número es mayor o igual

LEQ ➔ Se encarga de comprobar que el primer número es menor o Igual al segundo.

A continuación, verificamos que todos funcionan correctamente con una comparación verdadera y otra falsa:

Mayor que:

if 3 GTR 1 echo 3 es mayor que 1

if 3 GTR 15 echo 3 es mayor que 15

Menor que:

if 1 LSS 7 echo 1 es menor que 7

if 6 LSS 1 echo 6 es menor que 1

Mayor o igual que:

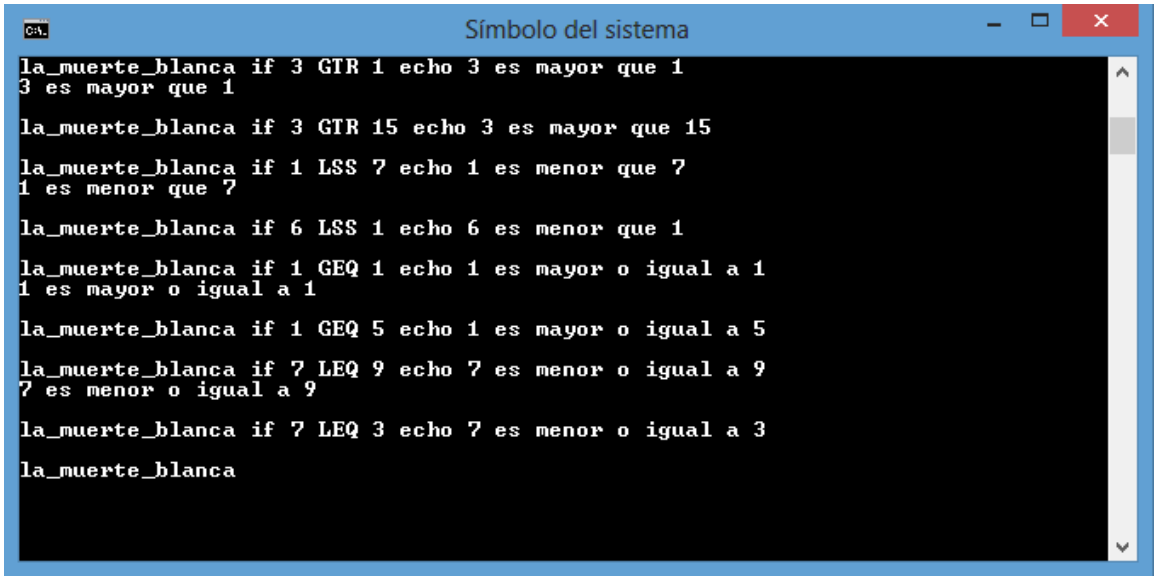
if 1 GEQ 1 echo 1 es mayor o igual a 1

if 1 GEQ 5 echo 1 es mayor o igual a 5

Menor o igual que:

if 7 LEQ 9 echo 7 es menor o igual a 9

if 7 LEQ 3 echo 7 es menor o igual a 3



¡Excelente! Todas las comparaciones han sido correctas. Por cierto, **LSS, GEQ** y **LEQ** se pueden poner en minúsculas, pero se suelen poner en mayúsculas para mejor legibilidad del código.

Pero si queremos crear una condición, que en caso de que no se cumpla (la condición) se ejecute, ¿qué debo hacer?

Pues la respuesta a esta pregunta es muy simple: debes poner "**not**", como pueden observar en los siguientes unos ejemplos:

if not 9==4 echo 9 no es igual a 4, por eso me ejecuto igual

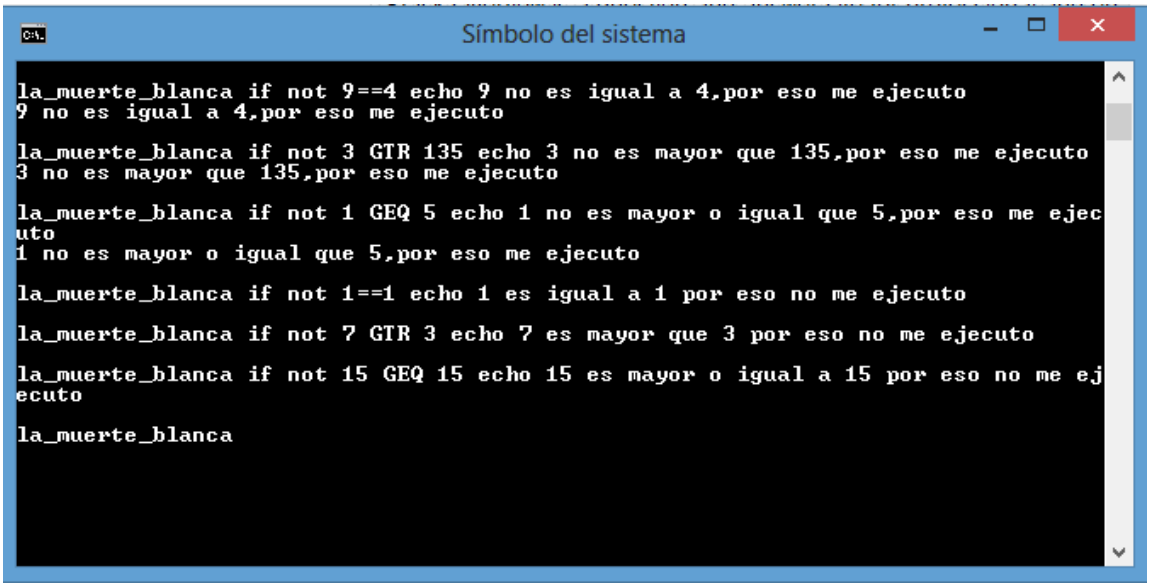
if not 3 GTR 135 echo 3 no es mayor que 135, por eso me ejecuto igual

if not 1 GEQ 5 echo 1 no es mayor o igual que 5, por eso me ejecuto

if not 1==1 echo 1 es igual a 1, por eso no me ejecuto

if not 7 GTR 3 echo 7 es mayor que 3, por eso no me ejecuto

if not 15 GEQ 15 echo 15 es mayor o igual a 15,por eso no me ejecuto



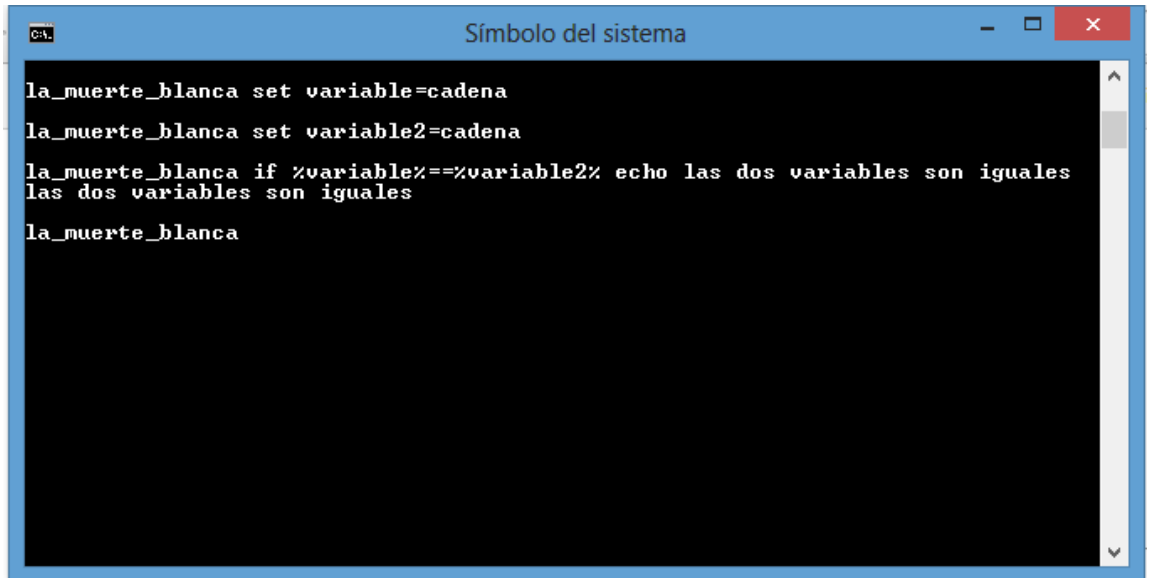
¿A qué no es tan difícil? El **not** es tan simple que si se cumple la comparación, pues no se ejecuta, pero en caso de que no se cumpla, entonces va a ejecutarse.

Ahora que hemos visto las comparaciones de números, les comento que también se pueden comparar variables. Solamente tenemos que darle un valor a las variables y comparar, en este ejemplo, compararemos dos cadenas:

set variable=cadena

set variable2=cadena

if %variable1%==%variable2% echo las dos variables son iguales



Perfecto, ya habrán comprobado que también se pueden comparar cadenas.

Practiquemos con un ejercicio...

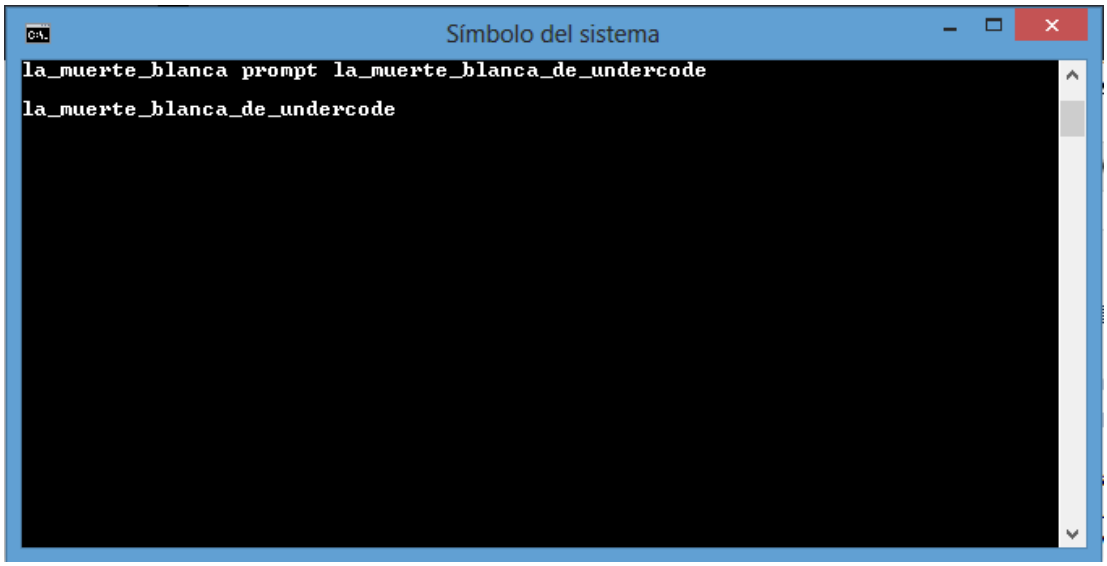
Ejercicio 6: Tendrán que crear un programa que le pregunte al usuario por dos variables: variable1 y variable2; y en cada una de ellas que ingrese un número. Luego que compare ambas y si son iguales, que devuelva son iguales y si son diferentes, que devuelva son diferentes.

COMANDOS PROMPT, COLOR Y TITLE:

Vamos a aprender unos comandos, que a la vez de simples son muy interesantes. Seguramente, habrán observado que las imágenes, en lugar de poner el directorio en el que nos encontrábamos, ponía el nick: La_muerte_blanca y se preguntarán cómo hacerlo. Para explicároslo, les presento el comando **Prompt**.

Prompt: Este comando se utiliza para cambiar el nombre del directorio en Batch. El comando se utiliza de esta manera: **prompt "nombre".**

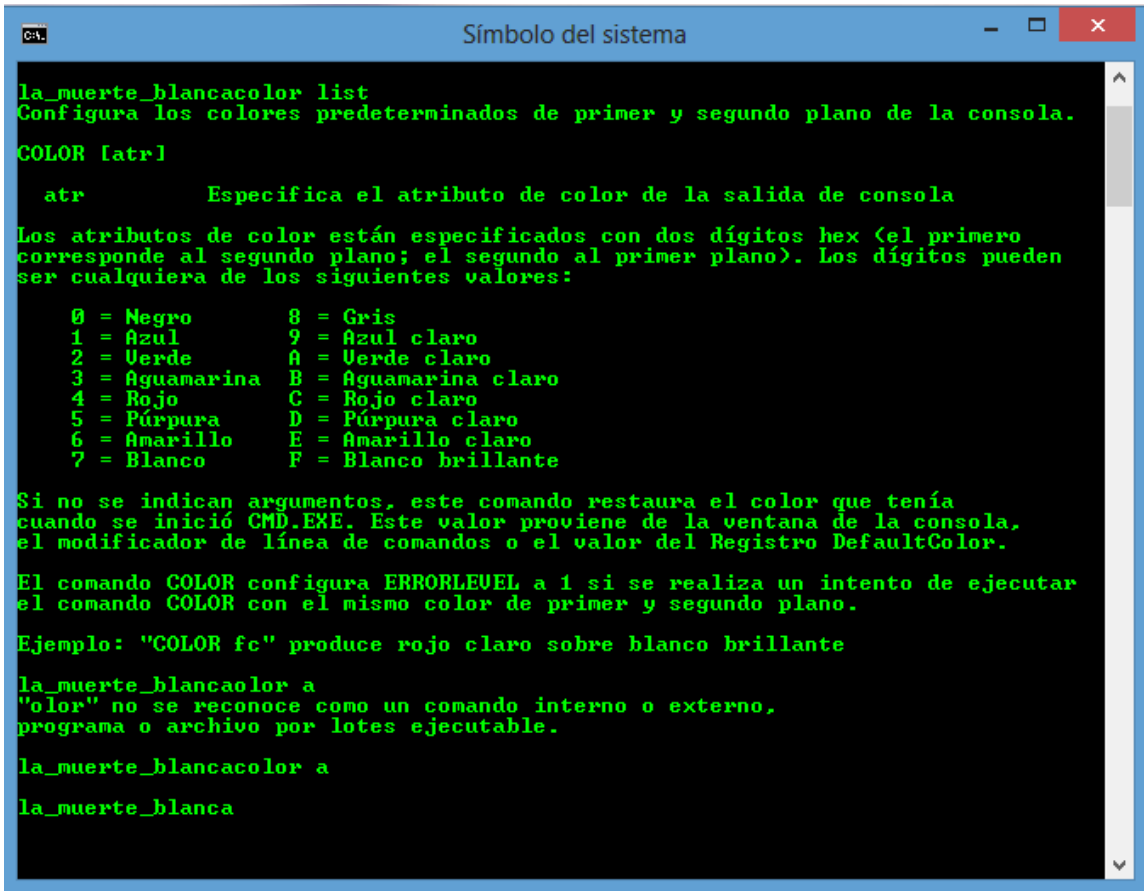
Vamos a probarlo, en nuestro caso escribiré prompt la_muerte_blanca_de_undercode, pero ustedes pueden poner el nombre que quieran.



Se recomienda escribir un espacio al final, para que lo que escriban no se una al nombre y los lleve a confusión.

Este comando tiene una utilidad estética porque permite cambiar el nombre del directorio, disimulando un dato que no quieran mostrar o porque quieran mostrar un nick, sin tener luego que editar la imagen.

Color: Como indica su nombre, este comando sirve para cambiar el color de las letras. En la consola de comandos, escribimos **color list** y aparecerá un listado de colores como puede verse en la imagen siguiente. Elegid el que más les guste. Mucha gente elige color "a" ya que les gusta la pantallita con las letras verdes como en las películas XD.



Title: Luego del “colorido” comando precedente, **title** nos que sirve para poner un título en la parte superior. Puede ser interesante usarlo, por ejemplo, para crear un programilla y ponerle de título un nombre.

Es muy fácil de utilizar, simplemente ponemos **title “aquí el título”**.

Nosotros, para mostrar su uso, podremos de **title** underc0de:

Si bien, los comandos son simples, los ejercicios combinan lo que venimos aprendiendo.

Ejercicio 7: Realizar una multiplicación de varios números pidiéndoselos al usuario; y si da superior o igual a 300 cambiar el nombre de directorio con prompt a mayor_de_trescientos. En caso de ser igual o inferior, no hacer nada.

Ejercicio 8: En este ejercicio tendrán que utilizar variables, condicionales y el comando color; de forma tal, que al ejecutar el programa te pida un número y si este número es menor que 7 el color cambie a azul, si es igual a 7 que el color cambie a rojo y si es mayor que 7 cambie a verde.

Ejercicio 9: Por último, tendrán que crear un programa que pida al usuario introducir una palabra la cual se pondrá como título a la consola de comandos.

ERRORES TÍPICOS

En este apartado, explicaremos alguno de los errores muy típicos que se suelen tener en Batch.

Palabras con tilde y algunos caracteres especiales:

Si ya han probado algunas cosas en Batch, habrán visto que desde el **cmd** se pueden escribir palabras con tildes, pero desde el block de notas no.

¿Qué podemos hacer?

Para escribir palabras con tildes y la ñ deberemos usar comandos **alt**:

á = alt 0160

é = alt 0130

í = alt 0161

ó = alt 0162

ú = alt 0163

Á = alt 0181

É = alt 0144

Í = alt 0214

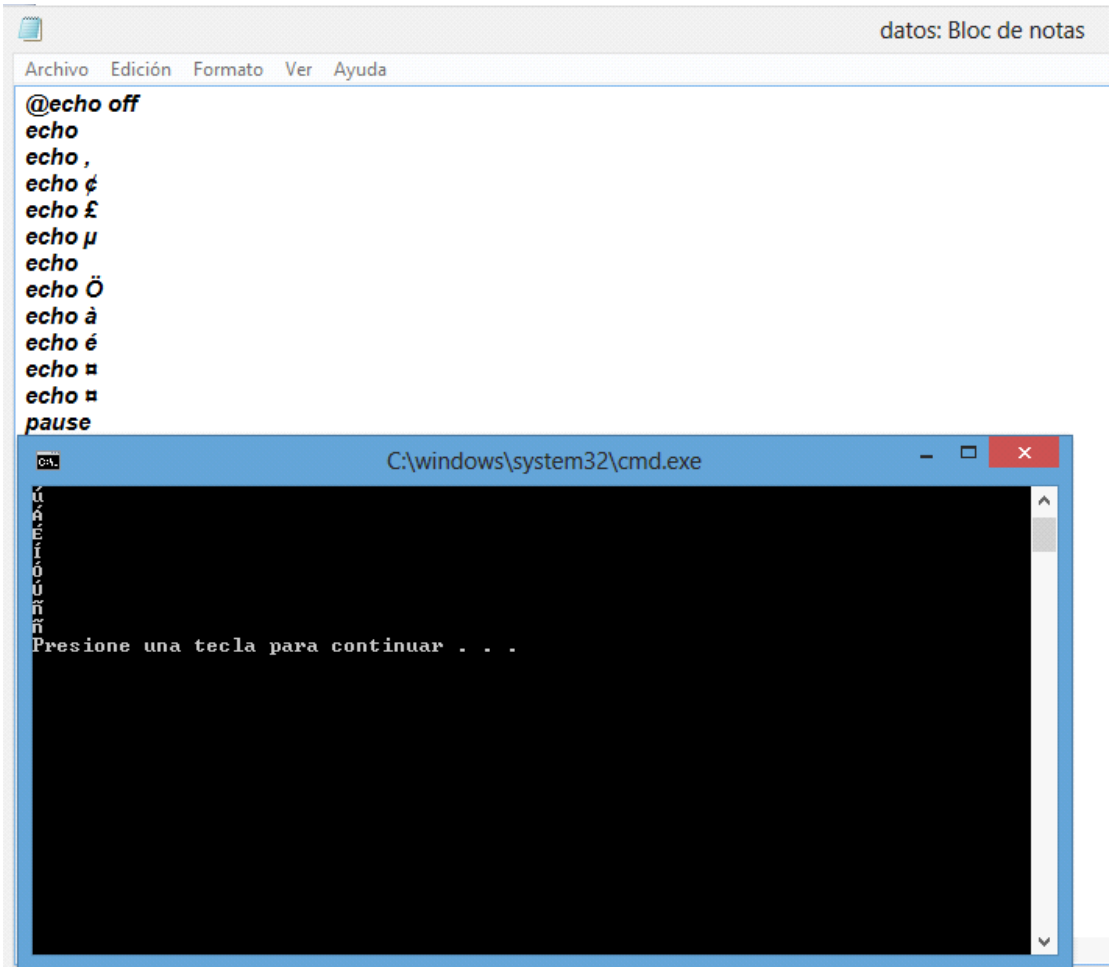
Ó = alt 0224

Ú = alt 0233

ñ = alt 0164

Ñ = alt 0165

Ahora, vamos a probar llamando a todos desde un archivo .bat:



Nota: La á y la Á son espacios, pero no son los espacios “normales” porque si prueban con ellos no funcionarán debido a que son unos espacios “especiales”; a efectos que se comprenda, podemos decir que están constituidos por caracteres invisibles.

No es necesario que se aprendan de memoria estos comandos con **alt,** pero se recomienda tener una lista con ellos ya que son muy útiles.

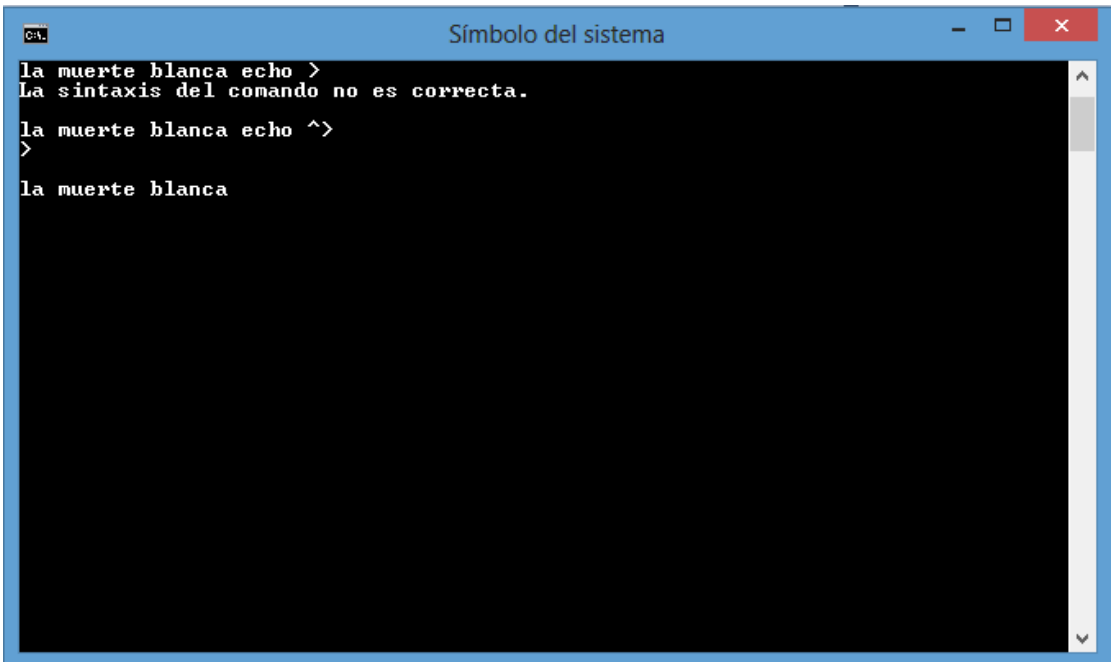
Por ejemplo, si quisiésemos escribir: “Tengo la solución”, tendríamos que poner:

echo Tengo la soluci¢n

Como pueden observar, utilizamos **alt 0162** para poner el signo que la consola de comandos identifica como ó.

También nos gustaría comentarles que si quieren realizar varios comandos en un **if** utilicen el comando **&**.

Por otra parte, para hacer que caracteres especiales como **>** salgan en la consola (que por cierto, no sale porque sirve para enviar datos a otro archivo), hay que colocar el símbolo **"^"**para que impida que la consola lo considere como un comando y crea que es una cadena. Aquí un ejemplo:



Ejercicio 10: Tendrán que preguntar al usuario: "¿Qué desea hacer escribir algo o una multiplicación con números? Elige escribir o multiplicar:"

En caso de que el usuario responda escribir, permitirle escribir algo y después mostrarlo en la consola con echo, y volver al principio.

En el caso de que el usuario elija multiplicar, permitirle al usuario escribir dos números y mostrar el resultado de su multiplicación con echo, y volver al principio.

En el caso de que el usuario escriba otra cosa distinta de la que se le pide, que la consola le de error y vuelva a elegir.

Para este ejercicio tendrán que usar muchas cosas de las aprendidas en el primer taller, como las etiquetas.

RESOLUCIÓN DE LOS EJERCICIOS

Para finalizar dejaremos una de las posibles soluciones a los ejercicios. También dar aviso, que pueden no tener un código igual, pero si el programa funciona el ejercicio estará correcto.

Ejercicio 1:

```
@echo off

set /p nombre=Introduzca su nombre:

echo Feliz cumpleaños %nombre%

pause
```

Ejercicio 2:

```
@echo off

set /p numero=elige el primer numero:

set /p otronumero=elige el segundo numero:

set/a suma=%numero%+%otronumero%

set/a multiplicacion=%numero%*%otronumero%

set/a resto=%numero%%%otronumero%

echo %suma%

echo %multiplicacion%

echo %resto%

pause
```

Ejercicio 3:

```
@echo off

set numero=1

:bucle

echo %numero%

set /a numero=%numero%+1

goto bucle
```

Ejercicio 4:

Este ejercicio puede resolverse con todos los números posibles, por ello mismo yo solo usaré el 1. Pondré como lo resolvería con el block de notas, pero también se puede realizar desde la consola de comandos:

```
@echo off

set /a operacion=1*1/1/1/1+1+1+1+1+1+1

echo %operacion%

pause
```

Ejercicio 5:

```
@echo off

set /p numero1=Elige el primer numero:

set /p numero2=Elige el segundo numero:

set /p numero3=Elige el tercer numero:

set/a operacion=%numero1% * %numero2% / %numero3%

echo %operacion%

pause
```

Ejercicio 6:

```
@echo off

set /p variable1=Elige un numero:

set /p variable2=Elige otro numero:

if %variable1%==%variable2% (echo son iguales) else (echo son diferentes)

pause
```

Ejercicio 7:

En este ejercicio no escribiremos @echo off para poder ver el directorio ya que lo haremos desde un .bat, pero si lo hacen desde la consola de comandos no es necesario ya que se verá igual.

```
set /p numero1=Elige un numero:

set /p numero2=Elige otro numero:

set /p numero3=Elige un ultimo numero:

set/a multiplicacion=%numero1%*%Numero2%*%numero3%
```

```
if %multiplicacion% GEQ 300 prompt mayor_de_trescientos

pause
```

Ejercicio 8:

```
@echo off

set /p numero1=Elige un numero:

if %numero1% LSS 7 color 1

if %numero1% ==7 color 4

if %numero1% GTR 7 color a

pause
```

Ejercicio 9:

```
@echo off

set/p titulo=Escribe una palabra:

title %titulo%

pause
```

Ejercicio 10:

```
@echo off


:hacer

set /p hacer="¿Que desea hacer escribir algo o una multiplicaci¢n con n£meros?Elige escribir o multiplicar:

if %hacer%==escribir goto escribir

if %hacer%==multiplicar goto multiplicacion

echo Error,vuelva a elegir & goto hacer


:escribir

set /p texto=¿Que deseas esribir?Escribe algo:

echo %texto%

pause
```

```
goto hacer


:multiplicacion
set /p numero1=Escribe el primer n£mero:
set /p numero2=Escribe el segundo n£mero:
set/a multiplicacion=%numero1%*%numero2%
echo %Multiplicacion%
pause
goto hacer
```

¡Hasta la próxima!