Programación en Shell: Condicionales `if`

Moisés García Villanueva

Febrero de 2018

1 Uso de las condicionales (`if`) en el shell

Nuestra vida cotidiana está llena de condiciones. En el ámbito escolar el más claro es cuando el profesor dice al estudiante: *SI estudias, aprobaras el examen.* La respuesta a dicha condición tiene dos valores lógicos: VERDADERO o FALSO, identificadas cuando la respuesta a dicha condición lleva como resultado responder un SI, cuando efectivamente se estudio y se aprobó el examen; y un NO, cuando por no estudiar se reprueba el examen. Algunos otros ejemplos de condicionales en nuestra vida cotidiana son:

- SI traigo dinero, puedo utilizar el transporte público
- SI me levanto temprano, llegaré a tiempo a mis actividades
- SI respeto las señales viales, evitaré tener la culpa de los accidentes

En computación, las decisiones se toman al comparar cantidades (por ejemplo: SI 3 es menor que 5 o 10 es mayor que 1), valores de variables del tipo entero, real o booleanas. Los operadores de comparación en el lenguaje de programación shell para cuando los operandos de la condicional son números enteros se describen en la tabla 1.

Operador	Significado	`a`	`b`	Ejemplo	Resultado
-lt	menor que (less than)	`a=10;` `a=21;`	`b=20;` `b=32;`	`[ $a -lt $b ]` `[ $b -lt $a ]`	Verdadero Falso
-gt	mayor que (greater than)	`a=10;` `a=21;`	`b=20;` `b=32;`	`[ $a -gt $b ]` `[ $b -gt $a ]`	Falso Verdadero
-le	menor o igual (less equal)	`a=100;` `a=321;`	`b=100;` `b=332;`	`[ $a -le $b ]` `[ $b -le $a ]`	Verdadero Falso
-ge	mayor o igual (greater equal)	`a=1110;` `a=321;`	`b=1100;` `b=332;`	`[ $a -ge $b ]` `[ $b -ge $a ]`	Verdadero Verdadero
-ne	no iguales (not equal)	`a=1110;` `a=123;`	`b=1100;` `b=123;`	`[ $a -ne $b ]` `[ $b -ne $a ]`	Verdadero Falso

Tabla 1: Operadores para variables de valores numéricos en las condicionales del lenguaje Shell.

En el shell existen otras dos formas de realizar comparaciones para números enteros:

1. Utilizando paréntesis dobles `(( ... ))` (forma más recientemente implementada)
2. Utilizando corchetes dobles `[[ ... ]]`

Al utilizar los paréntesis dobles, es posible realizar las comparaciones mediante los operadores que comúnmente empleamos en las clases de matemáticas para comparar cantidades, ver la tabla 2 que nos muestra ejemplos de los operadores de comparación mediante la sintaxis de paréntesis dobles.

La comparación o aritmética con valores de números reales es posible realizarla mediante el comando `bc`, un ejemplo para comparar dos valores reales es:

```
x=1.2134; y= 2.1439;
echo "scale=4; $x>$y" | bc
```

El valor que regresa la ejecución del comando con `bc` cuando utiliza los operadores de comparación son:

- 0 cuando el resultado de la comparación de valores es Falso
- 1 cuando el resultado de la comparación de valores es Verdadero

Ejemplos de uso de comparaciones de números reales mediante el comando `bc` en las condicionales del shell, se muestran en la Tabla 3.

Operador	Significado	a	b	Ejemplo	Resultado
<	menor que	a=10; a=21;	b=20; b=32;	((a < b)) ((b < a))	Verdadero Falso
>	mayor que	a=10; a=21;	b=20; b=32;	((a > b)) ((b > a))	Falso Verdadero
<=	menor o igual	a=100; a=321;	b=100; b=332;	((a <= b)) ((b <= a))	Verdadero Falso
>=	mayor o igual	a=1110; a=321;	b=1100; b=332;	((a >= b)) ((b >= a))	Verdadero Verdadero
==	iguales	a=1110; a=123;	b=1100; b=123;	((a == b)) ((b == a))	Falso Verdadero

Tabla 2: Operadores para variables de valores numéricos en las condicionales del lenguaje Shell.

Operador	x	y	Código de ejemplo	Resultado

<	3.14	8.21	x=3.14;y=8.21;
			if ((‘echo "scale=2;$x<$y" | bc‘ == 1))
			then echo "$x es menor que $y";
			else echo "Falso"; fi
				3.14 es menor que 8.21

>=	3.14	3.14	x=3.14;y=3.14;
			if ((‘echo "scale=2;$x>=$y" | bc‘ == 1))
			then echo "$x es mayor/igual que $y";
			else echo "Falso"; fi
				3.14 es mayor/igual que 3.14

Tabla 3: Códigos de ejemplo para implementar condicionales en el shell con números reales a través del comando `bc`.

1.1 Estructura de las condicionales `if`

Es importante conocer la forma correcta en como debe escribirse la estructura de las condicionales `if` en las instrucciones del `shell` y lograr aplicarlo a los diversos problemas que se plantean en las soluciones implementadas en sistemas de cómputo. Recordar que existen tres formas de implementar las comparaciones, así que expresaremos la estructura de las condicionales empleando las tres formas de comparación:

1. Paréntesis dobles

```
if (( EXPRESIÓN-DE-COMPARACIÓN ))
then
        # Instrucciones a ejecutarse cuando la comparación
        # dentro de los paréntesis dobles es VERDADERA
else
        # Instrucciones a ejecutarse en caso de que la
        # EXPRESIÓN-DE-COMPARACIÓN sea FALSA
fi
```

2. Corchetes sencillos

```
if [ EXPRESIÓN-DE-COMPARACIÓN ]
then
        # Instrucciones a ejecutarse cuando la comparación
        # dentro de los paréntesis dobles es VERDADERA
else
        # Instrucciones a ejecutarse en caso de que la
        # EXPRESIÓN-DE-COMPARACIÓN sea FALSA
fi
```

3. Corchetes dobles

```
if [[ EXPRESIÓN-DE-COMPARACIÓN ]]
then
        # Instrucciones a ejecutarse cuando la comparación
```

```
        # dentro de los paréntesis dobles es VERDADERA
    else
        # Instrucciones a ejecutarse en caso de que la
        # EXPRESIÓN-DE-COMPARACIÓN sea FALSA
    fi
```

El siguiente código de ejemplo nos imprime en la salida estándar de la terminal sólo los valores que son múltiplos de 11, para lograrlo se emplea una condicional que involucra la comparación del resultado que nos arroja el comando `‘expr $i % 11‘`, en donde `$i` es el valor que va tomando la variable i de la secuencia que se genera en el ciclo `for` desde 1 a 123; y el operador `%` significa el valor del residuo que se obtiene de dividir $\frac{i}{11}$, entonces cuando el residuo es cero y se compara con el operando del lado derecho del operador `==` en la condicional `if` dando como resultado VERDADERO, se ejecuta la instrucción que imprime un mensaje: `echo "El $i es múltiplo de 11"`.

```
for i in ‘seq 1 223‘
do
    if (( ‘expr $i % 11‘ == 0 ))
    then
        echo "El $i es múltiplo de 11"
    fi
done
```

Escribir en un archivo de texto plano, con nombre `ejem-if.sh`, el código del ejemplo anterior y ejecutar en una terminal de comandos de Linux de la siguiente forma:

```
bash ejem-if.sh
```

Ejercicios

1. Editar nuevamente el archivo `ejem-while.sh`, cambiar el valor 11 por el texto `$1`, con ello logrará proporcionar en la línea de comandos el número que usted desee utilizar como múltiplo de los valores de la secuencia. Después del cambio realizado, ejecutar el código de la siguiente forma:
   `bash ejem-if.sh 7`

2. Para pasar los valores de la secuencia desde la línea de comandos, debe cambiar los valores en el comando `seq` por `$2` y `$3`. Una de las posibles formas de ejecutar ahora el archivo `ejem-if.sh` puede ser de la siguiente forma:
   `bash ejem-if.sh 7 237 1439`

Nota: El paso de parámetros en un programa, permite sin modificar el código, probar para diferentes valores la ejecución y observar los diferentes resultados.

1.2 Rectángulo de dos colores

En este ejercicio, se utilizaran dos estructuras de instrucciones repetitivas, dentro de las cuales se colocaran las condicionales para rellenar la mitad del rectángulo de un color y la otra mitad de otro color, véase la Figura 1 en donde se tiene una imagen de 500 píxeles de ancho por 100 píxeles de alto, coloreada la mitad izquierda de un color y la mitad derecha de otro color.

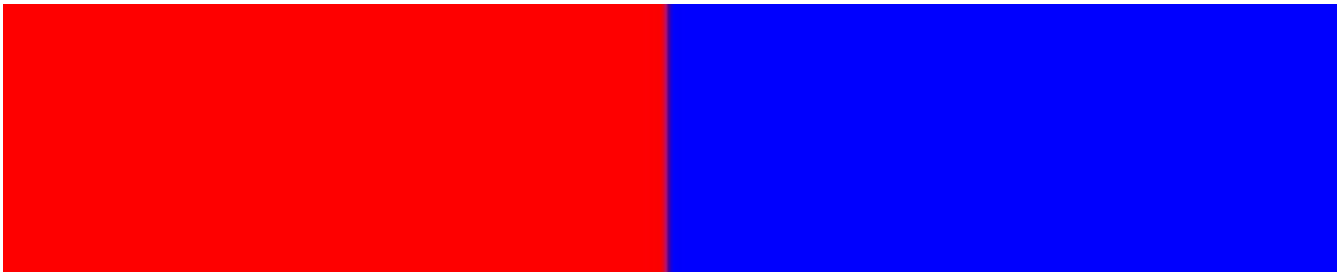

Figure 1: Rectángulo coloreado por mitad de colores diferentes. Imagen de 500 píxeles de ancho por 100 píxeles de alto.

La siguiente secuencia de actividades a realizar, permiten entender la forma en que se va construyendo la imagen y cómo es que las condicionales se aplican.

1. Primero escribiremos en un archivo (`rectangulo.sh`) el código para imprimir el encabezado del archivo `ppm` y dos ciclos `for` que permite imprimir todos los píxeles de la imagen en color rojo.

```
# Imprimir el encabezado de la imagen
echo -e "P3\n500 100\n255"

# ciclo para la cantidad de filas o renglones en la imagen
for i in ‘seq 1 100‘
    do
        #ciclo para la cantidad de columnas en la imagen
        for j in ‘seq 1 500‘
        do
            echo -n "255 0 0 "
        done
        echo ""
done
```

   Ejecutar en la terminal el archivo `rectangulo.sh` de la siguiente forma:
   `bash rectangulo.sh > r1color.ppm; eog r1color.ppm`

2. Para imprimir la mitad de columnas de un color y la otra mitad de otro color, se colocará una condicional en la que se indique que SI el valor de la variable del ciclo que cuenta la cantidad de columnas es menor a la mitad del ancho de la imagen (en este caso 250), se deben imprimir los valores que corresponden al color ROJO de un píxel. Se deben imprimir los valores del color AZUL para el caso en que no sea verdadera la condicional anterior, que es el caso cuando la variable `$j` del ciclo que imprime las columnas de la imagen es mayor al valor de la mitad de la imagen. Observe y analice el siguiente código que incluye la condicional que se explica en este párrafo, ahora escriba en un NUEVO archivo con el nombre `rectangulo-2.sh` el siguiente código.

```
echo -e "P3\n500 100\n255"

# ciclo para la cantidad de filas o renglones en la imagen
for i in ‘seq 1 100‘
    do
        #ciclo para la cantidad de columnas en la imagen
        for j in ‘seq 1 500‘
        do      # Condicional para dividir el color de las columnas en el rectángulo
                if (( j < 250 ))
                then
                    echo -n "255 0 0 "
                else
                    echo -n "0 0 255 "
                fi
        done
        echo ""
done
```

   Para observar el resultado del script, ejecutar en la terminal de comandos de la siguiente forma:
   `bash triangulo-2.sh > r2color.ppm; eog r2color.ppm`
   se observa en pantalla la imagen de la Figura 1.

1.3 Imagen dividida en 3 secciones

Vamos a ampliar el uso de las condicionales en el caso práctico de dividir una imágen en 3 secciones, independientemente de las dimensiones de la imagen. La figura 2 nos muestra ejemplos de imágenes que fueron divididas en 3 secciones iguales a partir del valor de ancho de la imagen que se proporciona.

1. Incluir el siguiente código en el archivo `rect-3c.sh` y ejecutarlo en la terminal de comandos para observar el resultado de la imagen generada:
   `bash rect-3c.sh 50 25 > r3colores.ppm; eog r3colores.ppm`

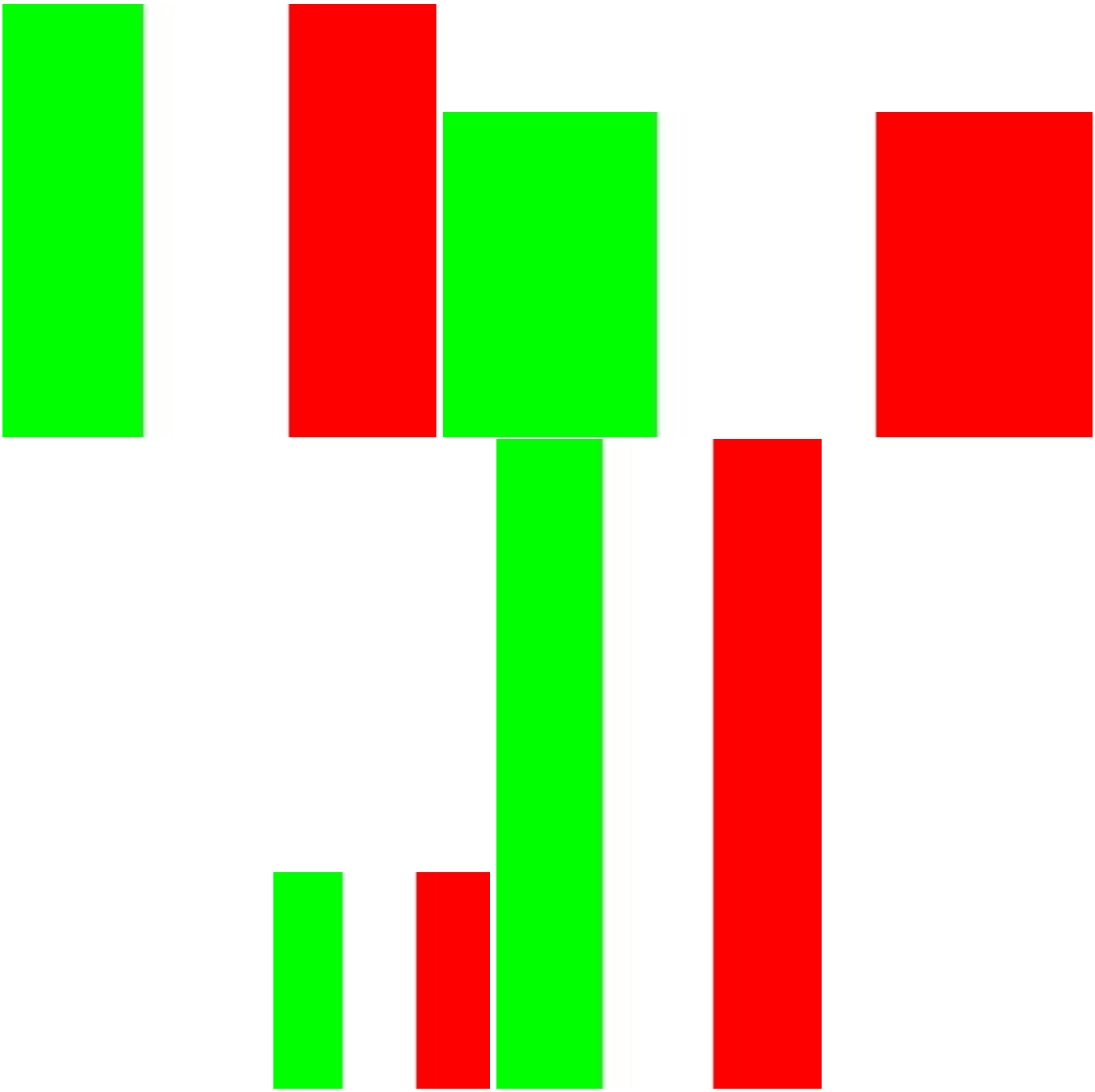

Figure 2: Se muestran 4 imágenes de diferentes dimensiones, cada una de ellas dividida en 3 secciones de color verde, blanco y rojo.

```
 El usuario deberá pasar los parámetros de ancho y alto de la imagen al
# momento de ejecutar el archivo con el código

ancho=$1
alto=$2
echo -e "P3\n$ancho $alto\n255"

seccion1=$((ancho/3))
seccion2=‘expr 2 \* $seccion1‘

# ciclo para la cantidad de filas o renglones en la imagen
for i in ‘seq 1 $alto‘
```

```
    do
        #ciclo para la cantidad de columnas en la imagen
        for j in `seq 1 $ancho`
        do  # Condicional para imprimir la sección de verde en las columnas de la imagen
            if (( j < seccion1 ))
            then
                echo -n "0 255 0 "
            elif (( j > seccion2 ))  # Condicional para imprimir la sección de rojo
            then
                echo -n "255 0 0 "
            else        # En caso de ser FALSAS las dos condicionales anteriores
                echo -n "255 255 255 "
            fi
        done
        echo ""
done
```

1.4 Ejercicios

1. Agregar una condicional más en el archivo en donde se dividió en 3 secciones la imagen, de tal forma que resulte una imagen dividida en 4 secciones.

2. Implemente la imagen de la Figura 3 mediante un código en shell que incluya instrucciones repetitivas y condicionales.

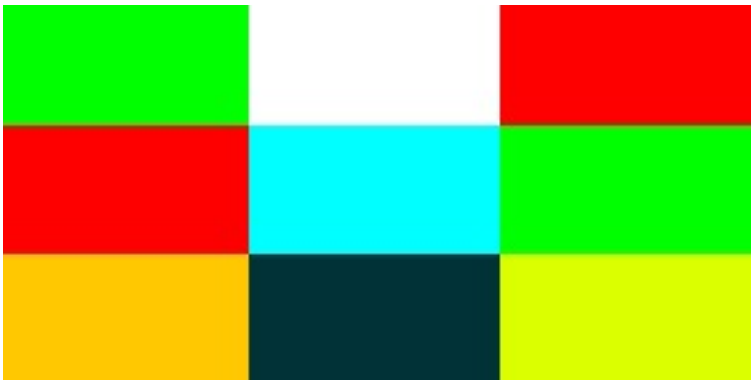

Figure 3: Imagen con 9 divisiones.

3. Implementar el código en el lenguaje del shell que proporcione la solución de una imagen que contenga un tablero de ajedrez, utilizando principalmente condicionales y que la dimensión de la imagen la proporcione el usuario. El programa deberá calcular la dimensión de los cuadrados del tablero de ajedrez de acuerdo a la dimensión de la imagen.