
Subject: Re: Meaning of the assignment

Posted by [Suresh Negi](#) on Wed, 21 Sep 2016 09:06:39 GMT

[View Forum Message](#) <> [Reply to Message](#)

On Wednesday, September 21, 2016 at 12:39:12 PM UTC+5:30, Helder wrote:

> On Wednesday, September 21, 2016 at 8:12:54 AM UTC+2, Sanu wrote:

>> On Wednesday, September 21, 2016 at 11:26:48 AM UTC+5:30, Sanu wrote:

>>> exp = 'BYTE(BYTE(b1 LE 650 AND b2 GT 200)*1 + 0)'

>>>

>>> For this b< 650 and b2> 200 will be assigned as class 1.

>>> What is the meaning of 0 which is added?????

>>> Kindly help.

>>

>> Similar thing i found on thid

>>

>> exp = 'BYTE(BYTE(b1 GE 0.4)*21 OR BYTE((b2/b3) GE 4.0)*21)'

>> What is the meaning of *21

>

> Well, I also don't understand what you have. But you should maybe provide some more information: Normally a string is a string and will not be computed unless you use execute(). Is this what you are doing? Where is this exp being used?

>

> If the assignment were not a string, then you can figure out things as they were described in the post from yesterday.

> Take things apart, a bit like an onion... layer by layer.

> Let's consider this expression:

> BYTE(BYTE(b1 GE 0.4)*21 OR BYTE((b2/b3) GE 4.0)*21)

> You have at the most inner parts expressions like: b1 GE 0.4

> IDL evaluates if b1 is greater or equal to 0.4. If it is greater or equal, it will "substitute" in the expression this part with a 1. If not, it will put a 0. IDL returns from such an operation a byte value, so the conversion to byte using the function BYTE() is useless. The expression you gave can be simplified to:

> BYTE((b1 GE 0.4)*21 OR ((b2/b3) GE 4.0)*21)

> Now it gets a bit tricky.

> on the left side of the OR you have

> (b1 GE 0.4)*21

> on the right side

> ((b2/b3) GE 4.0)*21

>

> You have to evaluate these two separately. The first will give you:

> 1 * 21 ---> if b1 is greater or equal than 0.4

> 0 * 21 ---> if b1 is smaller than 0.4

> The second will give you

> 1 * 21 ---> if b2/b3 is greater or equal than 4.0

> 0 * 21 ---> if b2/b3 is smaller than 4.0

>

> The result of each operation is an integer. Either 0 or 21.

>

> The OR operation will return either 0 or 21 depending on the input:
> 0 OR 0 ---> will give 0
> 21 OR 0 ---> will give 21
> 0 OR 21 ---> will give 21
> 21 OR 21 ---> will give 21
>
> The outer shell of the onion is a byte function that converts the previous result to a byte value.
> So depending on the last operation, you will either get a 0 or a 21 as a byte instead of an integer.
>
> You should be able to figure out the rest alone. If not, I seriously recommend some basic programming reference book.
>
> Helder

Thank You Helder.

Do you have some e-book which I can refer for ENVI IDL programming.
