#CL. CLOJURE and RUBY and HASKELLCODES and Python.py

##AGAINST AN ANTI-CIA NETWORK WHICH CLAIMED TO ATTEMPT AGAINST CIAOPS IN NICARAGUA AND HONDURAS AND PENT5678. THIS ANTI-CIA NETWORK CLAIMED ITS MONIES CAME FROM SAUDI ARABIA AND FROM IRANIAN SHIITE MUSLIMS[pron.] .

###THESECODESAREPROPRIETARY%$^

<clojure::req:::lib::rb::py.Pythonlibraries([mathmodules]).Py.com>?

<require:graph.python.MODULES<>>> >?

<hs.nt.hs.ns.CL.cscriptings::Pythonlibraries:::? >><n>??Python.py.com}

<ph.PHP.ns.libraries.com>

```
def 67nihrgg a| b -> [Int] c <- [sort:: 5, 4, 3, 2] -> e [Int] ->?YES==[Int]:
Pair ->[Int] -> [Int] -> [Int] <- [Char] -> [Int]::Pair [h:: [Char] -> [BOOL] ]
xy -> lam//?y <- C

        let Name -> [to_String] <- [sort::54$%] -> E -> C <- [Char] -> [BOOL]:

                let Namesignature Namenum connection connect.[Char]->
[BOOL]:: until BOOL <- C = //?y <- C <- [e sort[Int  ] ]   xy -> lam//?x <- C

                seq [to_String] map vector{[sort]}.connection.connect.Char-
>?YES -> [Int] until seq map vector sort c

                map remapvec until vector = sort for connection.connect.save

Pair[h::[Char] -> [BOOL] = 1

                <php>
```

```
                $inter_Up = “%” +, “insert”, + “Enter the name of your
favorite code today.”;
                $Up_happy2 = “$$$>>>”,
“PHPLOAD::factor.radiobutton>?echoprintecho2’Up_hhhxxa’”;
                {
                        if $ inter_Up < “01”:
                        then echo “Up_hhhxxa”;
                                else
                         print “%”;
                }

                {
                        if $Up_happy2 == 3+5 < 5+3:
                        then print “Miamisoundmachine”;
                                else
                        print “PHPLOAD::fac2@radiobutton>?>
                                http://UPhhhxxa.secret.com.echo.pound>
                }

        {

        return inter_Up > “Allclear”;YES=BOOL
                </php>
```

```
        b| map seq cons vector || A && B == -1 print “I’m all clear!”
                b -> c| print “We have received your favorite code for
today!!!” C
                map C
                map b
                [map a] -> [Int]
                        connection.connect.save.flush
                rb.Ruby>
                ridged_variable3 = 3546578
                obli_var45_% = 2002345
                s = “$great_news3”
                        ridged_variable3.split
                        ridged_variable3.frequency
                                ridged_variable3 do { print |obli_var45_%.split| }
                print s
                print s

{



        <php>
```

```
            obli_var45_%.PHP->[echoprintecho2]-
>obli_var$%[radio.button]<n><html>.com
            echo obli_var45_%;
            print obli_var45_%;
            echo.remedy.radiobutton>?YES=BOOL
            print BOOL.%#

            </php>
}

            .rbrigidvar#3>||print rigid_variable3|print rigid_variable
            def Eleba_45%[n]:
                  mtnt47 = [“Eight”, “Four”, “Two”, “Two”, “Three”];
                  mtnt47.sort
                        mtnt47 = { [] }
                  print mtnt47
                  return mtnt47
            Eleba_45%[4567345]

            Int -> BOOLEAN -> p: Parser <- {1, 2, 3, 5,4 } : [hs, vt], [xs, xt]
-> [Int] ::: A <- C
                  BOOLEAN <- C
                        MyINtBOOLEAN == (4567890 <= 2+2) or ( 3 == 3);
```

```
                                    if Int <- C:Parser == [head], [tail] ->C
                                            then print myINtBOOLEAN;
                                                    end.


<hs.BOOLEAN.parser.p >?
end.rubyfunction<n>end
End
<PHP:LOAD.com.net>?
        print: techsupport<n>?y=BOOL
C<http:ftp::yourchoice234$%.mynetbuilt:2<hs.BOOLEAN.parser.pp>p.N.n.
hs.CL> ?
return 0;
print myIntBOOL
{


<php>
print “techSupport#$%!”;
echo [];
echo “your choice.”;



</php>
End
```