```python
import numpy as np
from UpwindSolver import upwind_solver
from AnalyticSolution import analytic_solver

# gives the maximum error between the numerical solution and the analytical solution on nx grid points
def max_error(num_sol, a_sol, nx):
    max = 0
    for i in range(nx):
        test = np.abs(num_sol[i] - a_sol[i])
        if test > max:
            max = test
    return max

# subroutine to determine the length of the grid based on nu
def N(nu):
    return np.power(2, nu)

# computes the empirical order of convergence for nu
def EOC(nu):
    num1 = upwind_solver(N(nu-1))[-2][-1]
    ana1 = analytic_solver(N(nu-1), 0.2)
    err1 = max_error(num1, ana1, N(nu-1))
    num2 = upwind_solver(N(nu))[-2][-1]
    ana2 = analytic_solver(N(nu), 0.2)
    err2 = max_error(num2, ana2, N(nu))
    eoc = np.log2(err2/err1) / np.log2(N(nu - 1)/N(nu))
    return eoc

# gives an array of the empirical error of convergence from nu = 2 to nu_max
def cv_and_conquer(nu_max):
    cv = []
    for nu in range(2, nu_max):
        cv.append(EOC(nu))
    return cv
```