```
/*
Assignment #2: Numerical Simulation of 1D Couette Flow
Author: Shawn Hinnebusch
Date: 9/25/2020

To compile: gcc -o2 hw2.exe main.c -lm
Example input:
Re: 8.0
dpdx: -2.0
*/

#include <math.h>
#include <stdio.h>
#include <stdlib.h>

typedef double REAL;
typedef int    INT;

void CouetteFlow(REAL *uNum, REAL *uNumOld, REAL *ExactSoln, REAL Re, REAL dpdx,
 INT numOfSegments, const REAL Vp);

int main( )
{
    REAL       Re;
    REAL       dpdx;
    const INT  numOfSegments = 20;
    const REAL Vp            = 1.0; // Constant plate velocity

    // Prompt user to enter Re number and pressure gradient
    printf("Enter a Reynolds number: ");
    scanf("%lf", &Re);
    printf("Enter a pressure gradient: ");
    scanf("%lf", &dpdx);
    printf("\n");

    // Allocate memory and initialize velocity field to zero
    REAL *uNum      = calloc(numOfSegments + 1, sizeof(*uNum));
    REAL *uNumOld   = calloc(numOfSegments + 1, sizeof(*uNumOld));
    REAL *ExactSoln = calloc(numOfSegments + 1, sizeof(*ExactSoln));

    // Set Boundary Conditions
    uNumOld[ numOfSegments ] = Vp;
    uNum[ numOfSegments ]    = Vp;

    // Call function to calculate solutions
    CouetteFlow(uNum, uNumOld, ExactSoln, Re, dpdx, numOfSegments, Vp);

    // Output the results to the file
    FILE *output;
    output = fopen("Exact_numerical.dat", "w");

    if (!output) return 1;
    fprintf(output, "# Numerical Simulation of 1D Couette Flow. First column is exact. Second column is n
umerical.\n");
    for (int n = 0; n < numOfSegments + 1; n++) {
        fprintf(output, "%8.6lf\t%8.6lf\n", ExactSoln[ n ] / Vp, uNum[ n ] / Vp);
    }

    fclose(output);

    // Free variables in memory
    free(uNum);
    free(uNumOld);

    return 0;
}

void CouetteFlow(REAL *uNum, REAL *uNumOld, REAL *ExactSoln, REAL Re, REAL dpdx,
 INT numOfSegments, const REAL Vp)
```

```
{
    // Declare Variables
    const REAL pho = 1.0; // Density of fluid
    const REAL H   = 1.0; // Spacing between plates
    REAL       y;
    const REAL delta_y = H / numOfSegments;
    REAL       v; // Kinematic Viscosity
    v                = Vp * H / Re;
    REAL dt          = 0.5 * delta_y * delta_y / v;
    REAL timeToSS    = H * H / v;
    INT  nomOfPoints = numOfSegments + 1;
    INT  nTimeSteps  = timeToSS / dt + 1; // Added 1 to always round up

    printf("Re = %f\n", Re);
    printf("dpdx = %f\n", dpdx);
    printf("Number of time steps: %d\n\n", nTimeSteps);

    // Finite difference calcualtion
    for (int t = 0; t < nTimeSteps; t++) {
        for (int i = 1; i < numOfSegments; i++) {
            uNum[ i ] = dt
              * (-1 / pho * dpdx + v * (uNumOld[ i + 1 ] - 2 * uNumOld[ i ] + uN
umOld[ i - 1 ]) / (delta_y * delta_y))
              + uNumOld[ i ];
            uNumOld[ i ] = uNum[ i ];
        }
    }

    // Exact solution
    for (int i = 0; i < nomOfPoints; i++) {
        ExactSoln[ i ] = Vp * y / H + 1 / (2 * v) * dpdx * (y * y - H * y);
        if (i != nomOfPoints - 1) { y = y + H / numOfSegments; }
    }
}
```