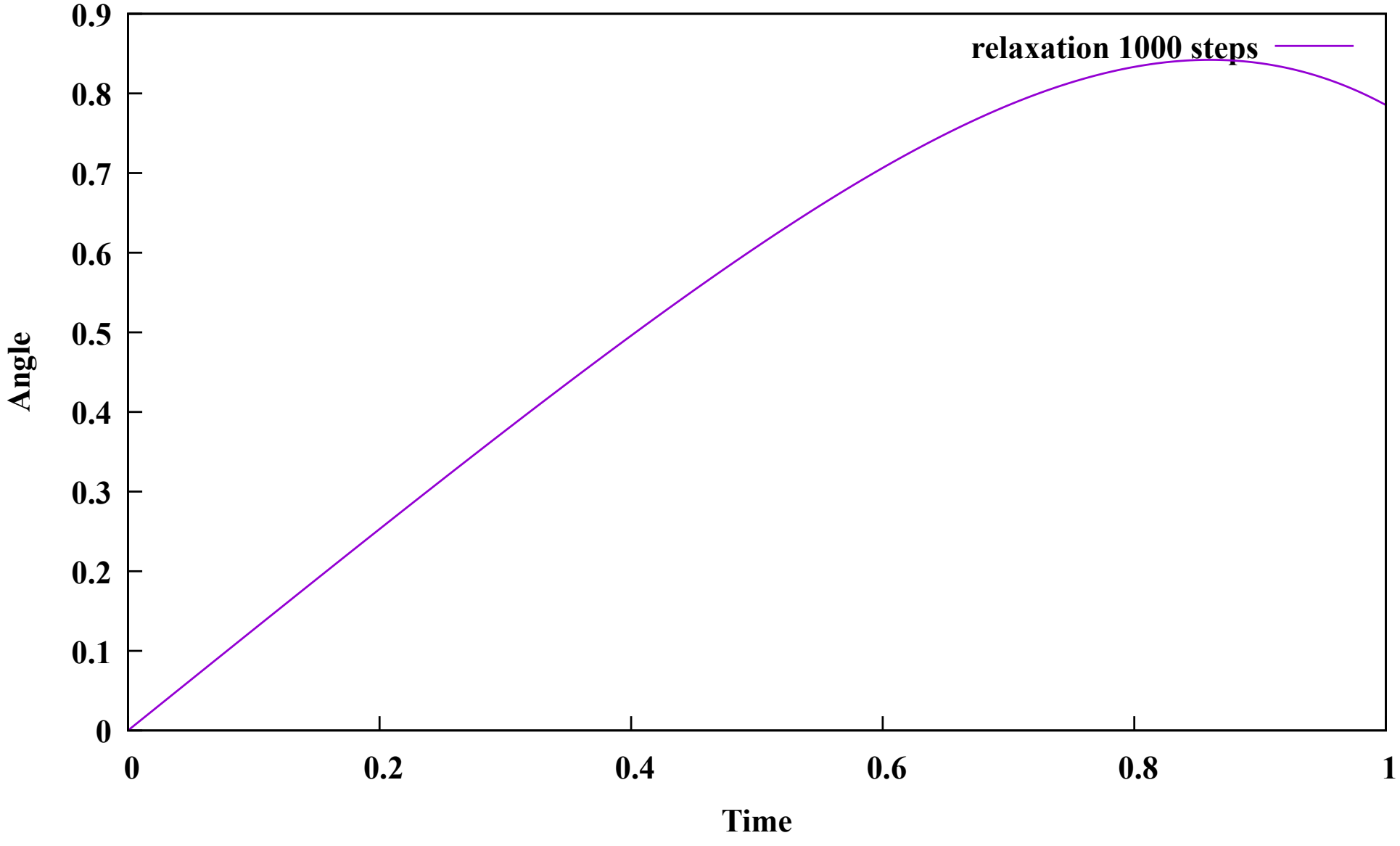
```cpp
#include <fstream>
#include <array>

//Constant expressions appearing in the problem
constexpr double PI = 3.14159265359;    //value of PI
constexpr size_t STEPS = 1000;  //number of steps

//Definition of data types in the problem
typedef double state_type;  //data type definition for dependant variables - array of x_0, x_1, ... x_n
typedef std::array<state_type, STEPS> solution;

//This is the differential Equation, with the higher order derivatives
state_type Pendulum(const state_type& x){
    return - PI * PI * sin(x);
}

//The relaxation step
void relaxation_step(state_type (*Diff_Equation)(const state_type& x), solution& x, const double& dt){
    solution temporary = x;
    for (size_t i = 1; i < STEPS - 1; i++) {
        x[i] = 0.5 * (temporary[i-1] + temporary[i+1] - dt * dt * Diff_Equation(temporary[i])); //Euler forward difference formula
    }
}

int main(){
    solution x_t{};   //variable to store the calculations

    double t_0 = 0.0;   //initial time
    double t_1 = 1.0;   //final time
    double dt = (t_1 - t_0) / (STEPS - 1); //step size
    state_type x_0 = 0.0, x_1 = PI / 4;   //boundary values for dependant variables

    //repeat the relaxation step
    for (size_t i = 0; i < STEPS; i++) {
        x_t[i] = x_0 + i * (x_1 - x_0) / (STEPS - 1); //initial linear guess
    }
    for (size_t i = 0; i < 10000; i++) {
        relaxation_step(Pendulum, x_t, dt);    //step forward
    }

    std::ofstream outfile;  //file handle to save the results in a file
    outfile.open("./output/relaxation.txt", std::ios::out | std::ios::trunc );
    for (size_t i = 0; i < STEPS; i++) {
```

```
44          outfile << t_0 + i * dt << "\t" << x_t[i] << std::endl;
45      }
46      outfile.close();
47  }
```


relaxation 1000 steps
Angle
Time
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
0.2
0.4
0.6
0.8
1