課題（常微分方程式の数値計算：高精度化）

常微分方程式系の数値計算を高精度化する方法として、一般に以下の３通りが考えられる。

１．丸め誤差を減少させるために、高精度の数値表現と数値演算を用いる。

２．打ち切り誤差を減少させるために、高次精度の差分近似式を用いる。

注：p 次精度の公式とは、微分方程式に対して $O(h^p)$ の打ち切り誤差を持つものである。

このとき 1 階常微分方程式系では近似解の厳密解に対する打ち切り誤差は $O(h^{p+1})$ である。

３．刻み幅 h の値を小さくする。

h を小さくしていけば、打ち切り誤差が減少して精度は高くなるが、計算時間は長くなる。
ただし h を小さくとればとるほど打ち切り誤差は減少するが、丸め誤差は微小な大きさではあるが増加するので、誤差を最小にするような h の値が存在する。（「2．丸め誤差と打ち切り誤差」を参照のこと）　通常は、丸め誤差の影響が無視できないほど小さい h の値はあまり用いられない。

本課題では、マス・バネ・ダンパ系の運動の数値シミュレーションを下記１，２，３に記したように高精化せよ。

１．丸め誤差の減少のために、double 型の数値表現と数値演算を用いる。

２．打ち切り誤差を減少させるために、2 次精度ルンゲ・クッタ法を 4 次精度ルンゲ・クッタ法にする。4 次精度ルンゲ・クッタ法においては、$t = t_n$ における解 $x_n = x(t_n),\quad v_n = v(t_n)$ から

$t_{n+1} = t_n + h$ における近似解 $x_{n+1} = x(t_n + h),\quad v_{n+1} = v(t_n + h)$ を以下のように求める。

$$
\begin{cases}
k_{x1} = hf_x(t_n, x_n, v_n) \\
k_{v1} = hf_v(t_n, x_n, v_n) \\
k_{x2} = hf_x(t_n + h/2, x_n + k_{x1}/2, v_n + k_{v1}/2) \\
k_{v2} = hf_v(t_n + h/2, x_n + k_{x1}/2, v_n + k_{v1}/2) \\
k_{x3} = hf_x(t_n + h/2, x_n + k_{x2}/2, v_n + k_{v2}/2) \\
k_{v3} = hf_v(t_n + h/2, x_n + k_{x2}/2, v_n + k_{v2}/2) \\
k_{x4} = hf_x(t_n + h, x_n + k_{x3}, v_n + k_{v3}) \\
k_{v4} = hf_v(t_n + h, x_n + k_{x3}, v_n + k_{v3})
\end{cases}
$$

$$x_{n+1} = x_n + \frac{1}{6}(k_{x1} + 2k_{x2} + 2k_{x3} + k_{x4})$$

$$v_{n+1} = v_n + \frac{1}{6}(k_{v1} + 2k_{v2} + 2k_{v3} + k_{v4})$$

３．時間刻み幅を、解が与えられた許容誤差 tol の範囲内に収まるまで、小さくする。
そのためのアルゴリズムは、次の全体アルゴリズムの中の項目 3 と 4 に記されたものである。

全体アルゴリズム：

1. **パラメータ値の入力**
   物理的な問題を規定するパラメータ値：
   バネ定数 K**、減衰定数** B**、質量** M
   変位と速度の初期値 $x0, v0$
   数値計算に必要なパラメータ値：
   時間刻み幅の初期値 $h0$**、許容誤差** tol
2. **初期値の設定：** $x(0) = x0, \quad v(0) = v0, \quad t(0) = 0.0, \quad h = h0$
3. **時間刻み幅** h **を調整して精度を確保しながら、時刻** t **における** $x(t), v(t)$ **から、時刻** $t+h$ **における解** $x(t+h), \quad v(t+h)$ **を 4 次精度ルンゲ・クッタ法により計算する。**

   3.1 **時間刻み幅** h **を用いて、時刻** t **における** $x(t), v(t)$ **から、時刻** $t+h$ **における解** $x(t+h), \quad v(t+h)$ **を計算する。**

   3.2 **時間刻み幅** $h/2$ **を用いて、時刻** t **における** $x(t), v(t)$ **から時刻** $t+h/2$ **における解** $x(t+h/2), \quad v(t+h/2)$ **を計算し、更にここから時間を** $h/2$ **だけ進ませて、時刻** $t+h$ **における解** $x(t+h), \quad v(t+h)$ **を計算する。この解を** $x_{h/2}(t+h), \quad v_{h/2}(t+h)$ **と表記する。**

   3.3 **時間刻み幅** h **を用いて計算した解** $x(t+h), \quad v(t+h)$ **と時間刻み幅** $h/2$ **を用いて計算した解** $x_{h/2}(t+h), \quad v_{h/2}(t+h)$ **を比較して、両方の差の絶対値の和が許容誤差** tol **以下**

   $$\left|x(t+h) - x_{h/2}(t+h)\right| + \left|v(t+h) - v_{h/2}(t+h)\right| < tol$$

   であるならば、時間刻み幅 h **で計算し続ける。もし** tol **よりも大きければ、** $h/2$ **を新たな時間刻み幅** h **に設定して、**3.1 **に戻る。**

   3.4 **上記** 3.1 ~ 3.3 **を許容誤差の条件が満たされるまで繰り返す。**
4. **以上** 3.1 ~ 3.4 **のやり方では、**h **は小さくなる一方なので、**h **が小さくなりすぎる恐れがある。これを防ぐために周期的に** h **を大きくする。例えば、時間を** 100 **回進めるごとに、** h **の値を** $h \times 16$ **にする。**
5. **所定の時間（**20 **秒）に達するまで、**3 ~ 4 **を繰り返す。**
6. **数値計算結果をファイルに出力する。**

SI **単位系を用いてパラメータ値を設定し、**20 **秒間の変位と速度を数値計算し、与えたパラメータ値と数値計算結果を次のようにファイルに記入する。**

C 言語文法上の注意：

・double 型変数の入出力上の書式指定

scanf　　float 型変数の読み込み　：%f
　　double 型変数の読み込み：%lf
printf　　float 型変数の浮動小数点数出力　：%f
　　double 型変数の浮動小数点数出力：%f と %lf のいずれでもよい
　　float 型変数と double 型変数の指数形式での出力：%e と %E のいずれでもよい

・絶対値を計算する関数 fabs

インクルードファイル　#include <math.h> を指定
プロトタイプは　　double fabs(double);
　　……　double 型の変数を引数にとり、double 型の値を返す。

ファイル出力

第 1 行：与えたパラメータ値
第 2 行：記入する数値計算結果の種類の記述
第 3 行以降：数値計算結果（時間番号、時間、変位、速度）

ファイル出力の例

```
K=50.000000 B=10.000000 M=10.000000 x0=10.000000 v0=0.000000 h0=1.000000 tol= 1.000e-06
    n        Time        Position        Velocity
    0  0.0000000e+00  1.0000000e+01  0.0000000e+00
    1  3.1250000e-02  9.9758482e+00 -1.5370866e+00
    2  6.2500000e-02  9.9045020e+00 -3.0194957e+00
    3  9.3750000e-02  9.7877563e+00 -4.4417863e+00
    :        :              :              :
  358  1.9750000e+01  2.0886955e-04  9.5290679e-04
  359  1.9875000e+01  3.1160827e-04  6.8579317e-04
  360  2.0000000e+01  3.7954286e-04  4.0002697e-04
```

前回の課題のファイル出力との相違

入力パラメータ *h* が *h*0 になったこと、*tol* という入力パラメータが増えたことのみである。

前回配布のグラフ表示プログラムの修正

数値計算では double 型の数値を用いても、ファイルを通じての数値計算結果データの受け渡しであるから、原則的には前回配布のグラフ表示プログラムをそのまま使うことができる。

ただし上記の出力例でもわかるとおり、1 行目の記述が 80 カラムを越えているので、ファイルから結果を読み込む関数 f_input において、以下の個所を以下のように修正する必要がある。

型宣言
　　char buf[80];　　===>　char buf[100];
関数 fgets により、ファイル fp から 1 行読み込む
　　fgets(buf, 80, fp); ===>　fgets(buf, 100, fp);
必要に応じて 100 を更に大きい数字にすればよい。

また前回の課題で気付いた人も多いと思うが、*t,x,v* 用の配列のサイズを大きくしておくこと：

#define NSIZE 101　===>　#define NSIZE 5001

提出期限：　１月２４日（木）

提出物：

数値計算プログラム	**ファイル名は　p○○××e91.c とすること**
グラフ表示プログラム	**ファイル名は　p○○××e92.c とすること**
数値計算結果の出力ファイル	**ファイル名は　p○○××e9.txt　とすること**

○○は学籍番号の上２桁の数字
××は学籍番号の下２桁の数字

- **＊　プログラムの第１行目には、コメントで（ /*　*/ でくくって)、学籍番号と名前を記述すること**
- **＊　数値計算結果の出力ファイル p○○××e9.txt は、各自が作成した計算プログラム p○○××e91.c によって数値計算を行った結果の出力ファイル**
- **＊　グラフ表示プログラム p○○××e92.c は､計算結果の出力ファイル p○○××e9.txt を読んで *x*,*v* の時間歴をグラフ表示できるように修正したプログラム（前頁「ファイル出力」参照のこと）**
  前回課題のバージョンアップ機能は含んでいても含まなくてもどちらでもよい。

提出形態：　E-mail に上記提出ファイルを添付すること

- **あて先**　cp2-69@cc.tuat.ac.jp
- **題**　cp2ex9
- **本文**　**グラフ表示プログラムで特記すべきことがあれば、記述**
  感想、学籍番号、名前を記述