```
/*----------------------------------------------------------------
*
* This code is based on the original UCLA model, and is modified
* by CIRCS group of Northeastern University.
*
* Contact Information:
*
* Center for interdisciplinary research on complex systems
* Departments of Physics, Northeastern University
*
* Alain Karma           a.karma (at) northeastern.edu
* Mingwang Zhong        mingwang.zhong (at) gmail.com
*
* The code was used to reproduce simulations in
* Transient outward K+ current (Ito) underlies the right ventricular
* initiation of polymorphic ventricular tachycardia in a transgenic
* rabbit model of long QT type 1, Bum-Rak Choi, Weiyan Li, Dmitry
* Terentyev, Anatoli Kabkov, Mingwang Zhong, Colin M Rees, Radmila
* Terentyeva, Tae Yun Kim, Zhilin Qu, Xuwen Peng, Alain Karma,
* and Gideon Koren (2018).
*---------------------------------------------------------------- */

// Information of original UCLA model:
/*--------------------- UCLA Model ver 1.00 ----------------------
*
* Contact Information
*
* Departments of Medicine (Cardiology)
* David Geffen School of Medicine at UCLA
*
* Daisuke Sato           dasato (at) mednet.ucla.edu
* Yohannes Shiferaw      yshiferaw (at) csun.edu
* James N Weiss          JWeiss (at) mednet.ucla.edu
*
* The code was used to produce simulations in
* A. Mahajan, Y. Shiferaw, D. Sato, A. Baher, R. Olcese, L.-H. Xie,
* M.-J. Yang, P.-S. Chen, J. G. Restrepo, A. Karma, A. Garfinkel,
* Z. Qu, and J. N. Weiss, A rabbit ventricular action potential model
* replicating cardiac dynamics at rapid heart rates, Biophysical Journal,
* 94 (2008), pp. 392¿410.
*---------------------------------------------------------------- */

#include <queue>
#include <iostream>
#include <cmath>

using namespace std;

class CCell{
        private:
                double PaceX(double stim=0);
                static const int N=50;
                static const double Vc;
                static const double stim;
                static const double stimduration;
                static const double temp; // temperature (K)
                static const double xxr; //
                static const double xf; // Faraday's constant
                static const double frt;

                #ifndef ___USE_VAR_FOR_CONST
                        static const double xnao; //mM, external Na
                        static const double xki; // mM, internal K
                        static const double xko; // mM, external K
                        static const double cao; // mM, external Ca
                        static const double ek;

                        static const double gca; // Ical conductance
                        static const double gtos; // ito slow conductance
                        static const double gtof; // ito fast conductance
                        static const double gnaca; // exchanger strength
                        static const double gks;
                        static const double gkr;
```

```
                        static const double vup; // uptake strength
                        static const double gna; // sodium conductance (mS/micro F)
                        static const double gK1; // Ik1 conductance
                        static const double gnak;

                        static const double taur; // spark lifetime (ms)
                        static const double taus; // diffusional delay (ms)
                        static const double taua; // NSR-JSR diffusional delay (ms)
                        static const double av;
                        static const double cstar;
                #endif

        public:
                double comp_ina (void);
                double comp_ikr(void);
                double comp_iks(void);
                double comp_ik1(void);
                double comp_ito(void);
                double comp_inak(void);
                double comp_inaca(double csm);
                double comp_hh_ical(double ica,double csm); // ica is single channel
 flux
                double comp_ica(double csm); // get single channel flux
                double comp_svipca(void); // PMCA
                double comp_iuptake(void);
                double comp_ileak(void);
                double comp_inst_buffer(double c);
                double comp_Q(void);
                double comp_dir(double Qr, double JCa,double dcj);

                double Pace(double stim=0);
                double PaceVClamp(double clampv);
                void ClampAP(double t, double BCL, double APD=0); //BCL ms
                void Prepare(double BCL=300, int Iter=0);


                double setdt(double DT){dt=DT;return dt;}
                double getdt(void){return dt;}
                int getDim(void){return N;}
                double getVc(void){return Vc;}
                double getstim(void){return stim;}
                double getstimduration(void){return stimduration;}

                CCell(void);
                virtual ~CCell();
                CCell& operator=(const CCell& cell);

                double vold, dtt, dt; // dtt is dt/N
                double *y;
                double &hf, &hd, &hf_ca, &ica; // for ical
                double &xm, &xh, &xhl, &xj; // for INa
                double &xs1, &xs2; // for IKs
                double &xtos, &ytos, &xtof, &ytof; // for Ito
                double &IKrC1, &IKrC2, &IKrC3, &IKrO, &IKrI; // for IKr
                double &v, &ci, &cs, &cj, &cjp, &cp, &step; // other
                double &xir, &xnai, &tropi, &trops, &jrel, &fspark; // other
                double _inaca, _ical, _iks, _ikr, _itof, _itos, _ik1, _ina, _inak, _
iup, _svipca, _up, _ir; // output

                #ifdef ___USE_VAR_FOR_CONST
                        double gca; // ical conductance
                        double gtos; // ito slow conductance
                        double gtof; // ito fast conductance
                        double gnaca; // exchanger strength
                        double gks;
                        double gkr;
                        double vup;
                        double gna; // sodium conductance (mS/micro F)
                        double gK1; // Ik1 conductance
                        double gnak;

                        double xnao; //mM, external Na
                        double xki; //mM, internal K
```

```
                        double xko; //mM, external K
                        double cao; //mM, external Ca
                        double ek;

                        double taus; // diffusional delay (ms)
                        double taur; // spark lifetime (ms)
                        double taua; // NSR-JSR diffusional delay (ms)
                        double av;
                        double cstar;
                #endif

};

////////////////////////////// constant parameters ////////////////////////////
const double CCell::Vc=-80;
const double CCell::stim=80;
const double CCell::stimduration=2;
const double CCell::temp=308.0; // temperature (K)
const double CCell::xxr=8.314; //
const double CCell::xf=96.485; // Faraday's constant
const double CCell::frt=xf/(xxr*temp);

#ifndef ___USE_VAR_FOR_CONST
        const double CCell::xnao=136.0; //mM, external Na
        const double CCell::xki=140.0; // mM, internal K
        const double CCell::xko=5.40; //mM, external K
        const double CCell::cao=1.8; // mM, external Ca
        const double CCell::ek = (1.0/frt)*log(xko/xki);// K reversal potential

        const double CCell::gca=182; // ical conductance
        const double CCell::gtos=0.04; // ito slow conductance
        const double CCell::gtof=0.11; // ito fast conductance
        const double CCell::gnaca=0.84; // exchanger strength
        const double CCell::gkr=0.0125; // Ikr conductance
        const double CCell::gks=0.32;
        const double CCell::gK1=0.3; // Ik1 conductance
        const double CCell::gnak=1.5;
        const double CCell::vup=0.4; //0.3;// uptake strength
        const double CCell::taus=4.0; // diffusional delay (ms)
        const double CCell::gna=12.0; // sodium conductance (mS/micro F)
        const double CCell::taur=30.0; // spark lifetime (ms)
        const double CCell::taua=100.0; // NSR-JSR diffusional delay (ms)
        const double CCell::av=11.3;
        const double CCell::cstar=90.0;
#endif
```