```
// the width of the channel is 1
// we have two geometrical parameters : the length of the channel lcanal and the length of the
reactive film lfilm
// limites du canal
// frontieres decrites dans le sens trigo direct
real lamont,lfilm,laval,lcanal,lmini;
cout << " Enter the length of the reactive film :"; cin >> lfilm;
lamont=5; laval=5;
lmini=lamont+laval+lfilm;
cout << " Enter the length of the channel (larger than"<< lmini << "):"; cin >> lcanal;
laval=lcanal-lamont-lfilm;
int namont,nfilm,naval,ncanal;
namont=floor(lamont);
nfilm=floor(lfilm);
naval=floor(laval);
ncanal=floor(lcanal);
// definition of the borders of the computation domain
border a(t=0,lamont) {x=t;y=0;}; // channel bottom upstream of film
border b(t=0,lfilm) {x=lamont+t;y=0;}; // film
border c(t=0,laval) {x=lamont+lfilm+t;y=0;}; // channel bottom downstream of film
border d(t=0,1) {x=lcanal;y=t;}; // outlet
border e(t=0,lcanal) {x=lcanal-t;y=1;}; // channel top
border f(t=0,1) {x=0;y=1-t;}; // inlet
int n=10;
cout << " Enter the mesh resolution (nb of meshes per channel width >1) :"; cin >> n;
// mesh construction and display
mesh th = buildmesh(a(namont*n)+b(nfilm*n)+c(naval*n)+d(n)+e(ncanal*n)+f(n));
plot (th,wait=1,ps="maillage.ps") ;
// definition of finite element spaces
// we use P2 elements

fespace Xh(th,P2); // velocity and concentration space
// u1,u2 components of fluid velocity
Xh u1,u2;
Xh r,rr,rold; // concentration
Xh rx,ry ; //  concentration gradients

// definition of the velocity
// parabolic velocity profile
// u1(y=0) = 0 ; u1(y=1) = 0 : average velocity <u1> = 1
func uin=6*y*(1-y) ;
u1=uin;
u2=0;
// Peclet number
real pe,invpe ;
cout << " Enter the Peclet number :"; cin >> pe;
invpe=1/pe;
// time step
real dt=0.1 ;
// initial concentration = 1
r=1; rold=r;
int i=0;
//  convection diffusion equation in variational formulation
// boundary conditions : concentration = 1 at the inlet of the channel (border f),
// concentration=0 on the film (border b)
problem Conc(r,rr)
```

```
    = int2d(th)(r*rr/dt + invpe*(dx(r) * dx(rr) + dy(r) * dy(rr)))
    - int2d(th)(convect([u1,u2],-dt,rold)*rr/dt)
     + on(f,r=1)
     + on(b,r=0);
// number of iterations
int ninit=500;
// iterative resolution on a uniform mesh
for (i=0;i<ninit;i++){
        Conc;
        rold=r;
        plot(cmm="iteration no "+i,r,nbiso=50,fill=1);
}
// remeshing based on the concentration field
th=adaptmesh(th,r);
// another iteration on the new mesh
Conc ;
plot (cmm="refined mesh",th,wait=1);
plot(cmm="concentration on refined mesh",r,nbiso=50,fill=1,wait=1);
// computation of concentration gradients
rx=dx(r);ry=dy(r);
// display of concentration gradients
plot(cmm="dc/dx",rx,nbiso=50,fill=1,wait=1);
plot(cmm="dc/dy",ry,nbiso=50,fill=1,wait=1);
// vectors used to plot vertical concentration profiles r(y) and concentration gradient dr/dy along the
film (y=0)
real[int] r1(100),r2(100),r3(100),ry0(100),pos(100);
for(i=0;i<100;i++){
        y=i*0.01;
        pos[i]=y;
        x=lamont;
        r1[i]=r;
        x=lamont+0.5*lfilm;
        r2[i]=r;
        x=lamont+lfilm;
        r3[i]=r;
}
plot(cmm="vertical concentration profiles at beginning, middle and end of the film",[r1,pos],[r2,pos],
[r3,pos],wait=1);
for(i=0;i<100;i++){
        y=0 ;
        x=lamont+0.01*lfilm*i;
        ry0[i]=ry;
}
plot(cmm="vertical concentration gradient along the film",[pos,ry0]);
// computation of the mass flux by integration of the gradient along the film
real flux ;
flux=int1d(th,b) (ry)/lfilm;
// with the definitions of the geometry and <u>=1, this flux is directly the value of the Sherwood
number
cout << "dimensionless average flux (Sherwood number) :" << flux << "\n" ;
```