章末問題解答

第　1　章

章末問題 (1)

次のように式を変形して用いればよい。

$$\sqrt{x+y}-\sqrt{x}=\frac{y}{\sqrt{x+y}+\sqrt{x}}$$

章末問題 (2)

絶対値が最小となる小数は，指数部 $e=1$ および仮数部 $b_i=0 \quad (i=9,\cdots,31)$ の場合であるからつぎの値となる。

$$\pm 2^{1-127}=\pm 2^{-126}=\pm 1.17549435\times 10^{-38}$$

絶対値が最大となる小数は，$e=254$ および $b_i=1 \quad (i=9,\cdots,31)$ の場合であるからつぎの値となる。

$$\pm 2^{254-127}\times\left(1+\sum_{i=9}^{31}\frac{1}{2^{i-8}}\right)=\pm 3.40282367\times 10^{38}$$

第　2　章

章末問題 (1)

行列の大きさを 3×3 としたときのプログラムをつぎに示す。

```
#include <stdio.h>
#define N 3
int main(void)
```

```
{
    int i, j, k;
    double a[N][N], b[N][N], c[N][N];
    for(i=0; i<N-1; i++){
        for(j=0; j<N-1; j++){
            printf("a[%d][%d] =", i, j);
            scanf("%lf", &x);
            a[i][j] = x;
        }
    }
    for(i=0; i<N-1; i++){
        for(j=0; j<N-1; j++){
            printf("b[%d][%d] =", i, j);
            scanf("%lf", &x);
            b[i][j] = x;
        }
    }

    for(i=0; i<=n-1; i++){
        for(j=0; j<=n-1; j++){
            c[i][j]=0.0;
            for(k=0; k<=n-1; k++){
                c[i][j]+=a[i][k]*b[k][j];
            }
        }
    }

    for(i=0; i<N-1; i++){
        for(j=0; j<N-1; j++){
            printf("c[%d][%d] = %f¥n", i, j, c[i][j]);
        }
    }
    return 0;
}
```

章末問題 (2)

```
#include <stdio.h>
#include <math.h>
#define  N  3

void crout(double a[N][N], double L[N][N], double U[N][N]);
```

```
int main(void)
{
    double a[N][N]={{6,5,4},{12,13,10},{18,21,17}},
           L[N][N], U[N][N], x;
    int i, j;

    crout(a, L, U);

    printf("¥n");
    for(i=0; i<=N-1; i++){
        for(j=0; j<=N-1; j++){
            printf("%f  ", L[i][j]);
        }
        printf("¥n");
    }
    printf("¥n");

    for(i=0; i<=N-1; i++){
        for(j=0; j<=N-1; j++){
            printf("%f  ", U[i][j]);
        }
        printf("¥n");
    }
    return 0;
}

void crout(double a[N][N], double L[N][N], double U[N][N])
{
    int i, j, k;

    L[0][0] = 1.0;
    U[0][0] = a[0][0];
    for(j=1; j<=N-1; j++)  U[0][j] = a[0][j];
    for(i=1; i<=N-1; i++)  U[i][0] = 0.0;
    for(i=0; i<=N-1; i++)  L[i][0] = a[i][0] / U[0][0];
    for(j=1; j<=N-1; j++)  L[0][j] = 0.0;

    for(i=1; i<=N-1; i++){
        L[i][i] = 1.0;
        U[i][i] = a[i][i];
```

```
        for(k=0; k<=i-1; k++){
            U[i][i] -= L[i][k] * U[k][i];
        }

        for(j=i+1; j<=N-1; j++){
            L[j][i] = a[j][i];
            for(k=0; k<=i-1; k++){
                L[j][i] -= L[j][k] * U[k][i];
            }
            L[j][i] /= U[i][i];
            U[j][i] = 0.0;
            L[i][j] = 0.0;
            U[i][j] = a[i][j];
            for(k=0; k<=i-1; k++){
                U[i][j] -= L[i][k] * U[k][j];
            }
        }
    }
}
```

第 3 章

章末問題 (1)

```
#include <stdio.h>
#include <math.h>
#define  N  3

void gauss(double a[N][N], double x[N], double b[N]);

int main(void)
{
    double a[N][N]={{6,5,4},{12,13,10},{18,21,17}}, x[N];
    double b[N]={8,16,27}, temp;
    int i, j;

    gauss(a, x, b);
    printf("¥n");

    for(i=0; i<=N-1; i++){
        printf("x[%d] = %f¥n", i, x[i]);
    }
```

```
    return 0;
}

/* ガウスの消去法計算ルーチン */
void gauss(double a[N][N], double x[N], double b[N])
{
    double copy, akk, aik;
    int i, j, k, max;

    /* ピボット選択 */
    for(k=0; k<=N-2; k++){
        max = k;

        for(i=k+1; i<=N-1; i++){
            if(fabs(a[i][k]) > fabs(a[max][k]))    max = i;
        }

        /* 例外処理 (対角成分が小さな値になったとき) */
        if(fabs(a[max][k]) < 1.0e-7)    exit(0);

        /* 行の入れ替え */
        if(max != k){
            for(j=0; j<=N-1; j++){
                copy = a[k][j];
                a[k][j] = a[max][j];
                a[max][j] = copy;
            }
            copy = b[k];
            b[k] = b[max];
            b[max] = copy;
        }

        /* 前進消去 */
        akk = a[k][k];

        for(j=k; j<=N-1; j++){
            a[k][j] /= akk;
        }
        b[k] /= akk;
```

```
        for(i=k+1; i<=N-1; i++){
            aik = a[i][k];

            for(j=k; j<=N-1; j++){
                a[i][j] -= aik * a[k][j];
            }
            b[i] -= aik * b[k];
        }
    }

    /* 後退代入 */
    x[k] = b[k] / a[k][k];

    for(k=N-2; k>=0; k--){
        for(j=0; j<=N-1; j++){
            x[k] -= a[k][j] * x[j];
        }
        x[k] += b[k];
    }
}
```

章末問題 (2)

```
#include <math.h>
#define  N  3   /* 変数の数 */
#define  M  400  /* 反復回数 */

void seidel(double a[N][N], double x[N], double b[N]);

int main(void)
{
    double a[N][N]={{6,5,4},{12,13,10},{18,21,17}}, x[N],
           b[N]={8,16,27}, temp;
    int i, j;

    for(i=0; i<N; i++)  x[i]=1.0;
    for(i=0; i<M; i++)  seidel(a, x, b);

    printf("¥n");
    for(i=0; i<N; i++){
        printf("x[%d] = %f¥n", i, x[i]);
    }
```

```
    return 0;
}

/* ガウスーザイデル法計算ルーチン */
void seidel(double a[N][N], double x[N], double b[N])
{
    double copy, akk, aik;
    int i, j;

    for(i=0; i<N; i++){
        x[i] = b[i];
        for(j=0; j<N; j++){
            if(i!=j)  x[i] -= a[i][j] * x[j];
        }
        x[i] /= a[i][i];
    }
}
```

第　4　章

章末問題 (1)

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

#define  N  5 /* 行列のサイズ */
#define PI 3.1415926535897

void jacobi(double a[N][N], double e[N], double v[N][N]);

int main(void)
{
    int i, j;
    double a[N][N]={{2.0,-1.0,0.0,0.0,0.0},
                    {-1.0,2.0,-1.0,0.0,0.0},
                    {0.0,-1.0,2.0,-1.0,0.0},
                    {0.0,0.0,-1.0,2.0,-1.0},
                    {0.0,0.0,0.0,-1.0,2.0}},
           e[N], v[N][N], x;

    jacobi(a, e, v);
```

```
    for(j=0; j<N; j++){
        printf("e[%d]=%15.8f¥n", j, e[j]);
        printf("固有ベクトル¥n");
        for(i=0; i<N; i++){
            printf("%15.8f¥n", v[i][j]);
        }
    }
    return 0;
}

/* ヤコビ法による固有値計算 */
void jacobi(double a[N][N], double e[N], double v[N][N])
{
    int i, j, kmax=100, repeat, p, q;
    double eps, c, s, theta, gmax;
    double apq, app, aqq, apqmax, apj, aqj, vip, viq;

    gmax=0.0;
    for(i=0; i<N; i++){
        s=0.0;
        for(j=0; j<N; j++){
            s += fabs(a[i][j]);
        }
        if(s>gmax) gmax=s;
    }
    eps=0.000001*gmax;

    for(i=0; i<N; i++){
        for(j=0; j<N; j++){
            v[i][j]=0.0;
        }
        v[i][i]=1.0;
    }

    for(repeat=1; repeat<kmax; repeat++){
        /* 収束判定 */
        apqmax = 0.0;
        for(p=0; p<N; p++){
            for(q=0; q<N; q++){
                if(p!=q){
                    apq=fabs(a[p][q]);
```

```
            if(apq>apqmax) apqmax=apq;
        }
    }
}
if(apqmax<eps) break;

for(p=0; p<N-1; p++){
    for(q=p+1; q<N; q++){
        apq=a[p][q];
        app=a[p][p];
        aqq=a[q][q];
        if(fabs(apqmax)<eps) break;

        if(fabs(app-aqq)>=1.0e-15){
            theta = 0.5*atan(2.0*apq/(app-aqq));
        }else{
            theta = PI/4.0;
        }
        c = cos(theta);
        s = sin(theta);

        a[p][p] = app*c*c + 2.0*apq*c*s + aqq*s*s;
        a[q][q] = app*s*s - 2.0*apq*c*s + aqq*c*c;

        a[p][q] = 0.0;
        a[q][p] = 0.0;

        for(j=0; j<N; j++){
            if(j!=p && j!=q){
                apj = a[p][j];
                aqj = a[q][j];
                a[p][j] = apj*c + aqj*s;
                a[q][j] = -apj*s + aqj*c;
                a[j][p] = a[p][j];
                a[j][q] = a[q][j];
            }
        }
        for(i=0; i<N; i++){
            vip=v[i][p];
            viq=v[i][q];
            v[i][p] = vip*c + viq*s;
```

```
                    v[i][q] = -vip*s + viq*c;
                }
            }
        }
        eps=eps*1.05;
    }
    for(i=0; i<N; i++)  e[i] = a[i][i];
}
```

章末問題 (2)

```
#include <stdio.h>
#include <math.h>
#define  N  5 /* 行列のサイズ */
#define  M  100 /* 演算の繰り返し回数 */

void crout(double a[N][N], double L[N][N], double U[N][N]);
void product(double a[N][N], double L[N][N], double R[N][N]);

int main(void)
{
    int i, j, k;
    double a[N][N]={{2.0,-1.0,0.0,0.0,0.0},
                    {-1.0,2.0,-1.0,0.0,0.0},
                    {0.0,-1.0,2.0,-1.0,0.0},
                    {0.0,0.0,-1.0,2.0,-1.0},
                    {0.0,0.0,0.0,-1.0,2.0}},
           L[N][N], R[N][N], x;

    for(k=0; k<=M; k++){
        crout(a, L, R);
        product(a, L, R);
    }

    for(i=0; i<=N-1; i++){
        for(j=0; j<=N-1; j++){
            printf("%f  ", a[i][j]);
        }
        printf("¥n");
    }
        printf("¥n");
    return 0;
```

```
}

/* LU 分解計算ルーチン（crout 法） */
void crout(double a[N][N], double L[N][N], double U[N][N])
{
    int i, j, k;

    L[0][0] = 1.0;
    U[0][0] = a[0][0];
    for(j=1; j<=N-1; j++)  U[0][j] = a[0][j];
    for(i=1; i<=N-1; i++)  U[i][0] = 0.0;
    for(i=0; i<=N-1; i++)  L[i][0] = a[i][0] / U[0][0];
    for(j=1; j<=N-1; j++)  L[0][j] = 0.0;

    for(i=1; i<=N-1; i++){
        L[i][i] = 1.0;
        U[i][i] = a[i][i];
        for(k=0; k<=i-1; k++){
            U[i][i] -= L[i][k] * U[k][i];
        }

        for(j=i+1; j<=N-1; j++){
            L[j][i] = a[j][i];
            for(k=0; k<=i-1; k++){
                L[j][i] -= L[j][k] * U[k][i];
            }

            L[j][i] /= U[i][i];
            U[j][i] = 0.0;
            L[i][j] = 0.0;
            U[i][j] = a[i][j];

            for(k=0; k<=i-1; k++){
                U[i][j] -= L[i][k] * U[k][j];
            }
        }
    }
}

/* 行列の積の計算 */
void product(double a[N][N], double L[N][N], double R[N][N])
```

```
{
    int i, j, k;
    for(i=0; i<=N-1; i++){
        for(j=0; j<=N-1; j++){
            a[i][j]=0.0;
            for(k=0; k<=N-1; k++){
                a[i][j]+=R[i][k]*L[k][j];
            }
        }
    }
}
```

第 5 章

章末問題 (1)

```
#include <stdio.h>
#define  N  6  /* データ数 */

int main(void)
{
    int i, j;
    double x[N]={0.0,1.0,2.0,3.0,4.0,5.0},
           y[N]={10.2,12.0,15.7,17.0,20.5,22.4},
           a0,a1,sumx=0.0,sumy=0.0,sumxx=0.0,sumxy=0.0;

    for(i=0; i<N; i++){
        sumx += x[i];
        sumy += y[i];
        sumxx += x[i]*x[i];
        sumxy += x[i]*y[i];
    }

    a0 = (sumy*sumxx - sumx*sumxy)/(N*sumxx - sumx*sumx);
    a1 = (N*sumxy -sumx*sumy)/(N*sumxx - sumx*sumx);

    printf("a0=%f,  a1=%f¥n", a0, a1);

    return 0;
}
```

章末問題 (2)

第 1 主成分 $\boldsymbol{z}_1$ の分散はつぎのように表すことができる。

$$V_{z_1} = \frac{1}{n-1}\sum_{i=1}^{n} z_{i1}^2$$
$$= \boldsymbol{a}^t \left(\frac{1}{n-1}\sum_{i=1}^{n} \boldsymbol{u}_i \boldsymbol{u}_i^t \right) \boldsymbol{a} = \boldsymbol{a}^t \boldsymbol{R} \boldsymbol{a}$$

ここで分散を求めるときにつぎの制約条件を利用する。

$$\sum_{i=1}^{p} \boldsymbol{a}_i^2 = \boldsymbol{a}^t \boldsymbol{a} = 1$$

ラグランジュの未定乗数法のための定式化を行う。

$$f(\boldsymbol{a}, \lambda) = \boldsymbol{a}^t \boldsymbol{R} \boldsymbol{a} - \lambda(\boldsymbol{a}^t \boldsymbol{a} - 1)$$

この式をベクトル $\boldsymbol{a}$ で微分して 0 とおくとつぎのようになる。

$$\frac{\partial f}{\partial \boldsymbol{a}} = 2\boldsymbol{R}\boldsymbol{a} - 2\lambda\boldsymbol{a} = 0$$

これより次式が得られる。

$$\boldsymbol{R}\boldsymbol{a} = \lambda\boldsymbol{a}$$

これより係数 a_i は上の固有値問題の最大固有値に対応する大きさ 1 の固有ベクトルを求めればよい。

第　6　章

章末問題 (1)

```
#include <stdio.h>

#define N 17
#define DIV 10

int main(void)
{
  int i, j;
  double x[N]={-4.0,-3.5,-3.0,-2.5,-2.0,-1.5,-1.0,-0.5,0.0,
               0.5,1.0,1.5,2.0,2.5,3.0,3.5,4.0},
         y[N]={0.0,0.1,0.4,0.9,1.6,2.5,3.6,4.9,6.4,
               11.3,14.9,17.4,19.0,19.9,20.3,20.4,20.4},
         xval, yval, h;

  h = 0.5 / (double)DIV;
```

```
  for(i=0; i<N-1; i++){
      for(j=0; j<DIV; j++){
          xval=x[i]+(double)j*h;
          yval=y[i]+(y[i+1]-y[i])*(xval-x[i])/(x[i+1]-x[i]);
          printf("x=%f, y=%f¥n", xval, yval);
      }
  }
  return 0;
}
```

章末問題 (2)

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

#define N 21

double lagrange(double xval,double x[N],double y[N],int k);

int main(int argc, char *argv[])
{
    int i, j, k=0;
    double x[N]={-5,-4.5,-4,-3.5,-3,-2.5,-2,-1.5,-1,-0.5,0,
                    0.5,1,1.5,2,2.5,3,3.5,4,4.5,5},
           y[N]={0.0066,0.01098,0.0179,0.0293,0.04742,
                    0.0758,0.1192,0.1824,0.2689,0.3775,0.5,
                    0.6224,0.7310,0.8175,0.8807,0.9241,
                    0.9525,0.9706,0.9820,0.9890,0.9933},
        xval, yval;

    for(i=0; i<=10*(N-1); i++){
        xval=x[k]+0.5*(double)i/10.0;
        yval = lagrange(xval, x, y, k);
        printf("%f¥t%f¥n", xval, yval);
    }
    return 0;
}

double lagrange(double xval,double x[N],double y[N],int k)
{
    int i,j;
    double s, nix;
```

```
    s = 0.0;
    for(i=k; i<=k+N-1; i++){
        nix = 1.0;
        for(j=k; j<=k+N-1; j++){
            if(i!=j){
                nix*=(xval-x[j])/(x[i]-x[j]);
            }
        }
        s+=y[i]*nix;
    }
    return s;
}
```

第　7　章

章末問題 (1)

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

#define PI 3.14159265358979323846
#define N 64 //FFT のデータ点数

void dft(double x[N], double Xr[N], double Xi[N]);

int main(void)
{
//DFT 計算に必要な変数
    double x[N], Xr[N], Xi[N];
    int i;

    for(i=0; i<N; i++)  x[i] = 0.0;
    for(i=N/4; i<3*N/4; i++)  x[i] = 1.0;

    dft(x, Xr, Xi);

    for(i=0; i<N; i++){
        printf("%f,%f,%f,%f¥n", Xr[i], Xi[i],
                sqrt(Xr[i]*Xr[i]+Xi[i]*Xi[i]), x[i]);
    }
    return 0;
}
```

```
void dft(double x[N], double Xr[N], double Xi[N])
{
    int k, p;

    for (k = 0; k < N; k++){
        Xr[k] = 0.0;
        Xi[k] = 0.0;

        for (p=0; p<N; p++){
                Xr[k]+=x[p]*cos(2.0*PI*p*k/(double)N);
                Xi[k]-=x[p]*sin(2.0*PI*p*k/(double)N);
        }
        Xr[k]=Xr[k]/sqrt((double)N);
        Xi[k]=Xi[k]/sqrt((double)N);
    }
}
```

第　8　章

章末問題 (1)

```
#include  <stdio.h>

double f(double t, double x);

int main(void)
{
    double  t, x , dx, h;
    int i;

    h=0.1;  /* 計算の刻み幅 */
    t=0.0;  x=1.0;  /* 微分方程式を解くための初期値 */

    for(i = 0; i < 21; i++){
        printf("%10.6f, %10.6f¥n", t, x);
        dx = h * f(t, x);  /* 増分の計算 */

        /* 値の更新 */
        x += dx;
        t += h;
    }
```

```
    return 0;
}

double f(double t, double x)
{
    double dif;
    dif =  - x;
    return(dif);
}
```

章末問題 (2)

```
#include  <stdio.h>

double f1(double t, double x1, double x2);
double f2(double t, double x1, double x2);

int main(void)
{
    double  t, x1, x2, dx1, dx2;
    double  k11, k12, k13, k14, k21, k22, k23, k24, h;
    int i ;

    h = 0.01;  /* 刻み幅の設定 */
    t = 0.0;  x1 = 0.0;  x2 = 0.0;  /* 初期値の設定 */

    for(i = 0; i < 20; i++) {
        k11=h*f1(t, x1, x2);
        k21=h*f2(t, x1, x2);
        k12=h*f1(t+h/2.0, x1+k11/2.0, x2+k21/2.0);
        k22=h*f2(t+h/2.0, x1+k11/2.0, x2+k21/2.0);
        k13=h*f1(t+h/2.0, x1+k12/2.0, x2+k22/2.0);
        k23=h*f2(t+h/2.0, x1+k12/2.0, x2+k22/2.0);
        k14=h*f1(t+h, x1+k13, x2+k23);
        k24=h*f2(t+h, x1+k13, x2+k23);
        /* 各変数の増分の計算 */
        dx1=(k11+2.0*k12+2.0*k13+k14)/6.0;
        dx2=(k21+2.0*k22+2.0*k23+k24)/6.0;

        printf("%10.6f  %10.6f  %10.6f¥n", t, x1, x2);

        /* 値の更新 */
```

```
        x1+=dx1;
        x2+=dx2;
        t+=h;
    }
    return 0;
}

double f1(double t, double x1, double x2)
{
    double dif;
    dif =  x2;
    return(dif);
}

double f2(double t, double x1, double x2)
{
    double dif;
    dif = x2 + 2*x1 + 6;
    return(dif);
}
```

第 9 章

章末問題 (1)

```
#include <stdio.h>
#include <math.h>
#define EPS 0.01

int main(void)
{
  double x1=0.5,x2=1.0,dx1,dx2,f1,f2,j11,j12,j21,j22,norm;
  int i, j;

  for(i=0; i<20; i++){
      f1 = 2.0*x1*x1*x1-x2*x2*x2-9.0;
      f2 = x1*x1*x1-2.0*x2*x2*x2-4.0;

      j11 = 6.0*x1*x1;
      j12 = -3.0*x2*x2;
      j21 = 3.0*x1*x1;
      j22 = -6.0*x2*x2;
```

```
      dx1 = (j22*f1-j12*f2)/(j11*j22-j12*j21);
      dx2 = (-j21*f1+j11*f2)/(j11*j22-j12*j21);

      norm = sqrt(dx1*dx1+dx2*dx2);
      if(norm<EPS)  break;

      x1 -= dx1;
      x2 -= dx2;
      printf("x1=%f,  x2=%f¥n",x1, x2);
  }
  return 0;
}
```

第　10　章

章末問題 (1)

```
#include <stdio.h>

#define  EPS  0.01
#define  ALPHA  0.1
#define  M  100  /* 反復回数 */

int main(int argc, char *argv[])
{
  double x1=0.0, x2=0.0, dx1, dx2, norm;
  int i, j;

  for(i=0;i<M;i++){
      dx1 = 2.0*(x1-2.0) + 2.0*(x1-2.0*x2);
      dx2 = -4.0*(x1-2.0*x2);
      norm = sqrt(dx1*dx1 + dx2*dx2);
      if(norm < EPS)  break;
      x1 -= ALPHA * dx1;
      x2 -= ALPHA * dx2;
      printf("x1=%f,  x2=%f¥n", x1, x2);
  }
  return 0;
}
```

第 11 章

章末問題 (1)

```
/* 台形公式による数値積分 */
#include <stdio.h>
#include <math.h>
#define  N  20

double daikei(double y[N], double dx);
double func(double x);

int main(void)
{
    int i;
    double y[N],h,a=1.0,b=2.0,x,integral;
    h = (b-a)/(double)N;
    for(i=0; i<=N; i++){
        x=a+(double)i*h;
        y[i]=func(x);
    }
    integral=daikei(y, h);
    printf("value = %f¥n", integral);
    return 0;
}

/* 台形公式による計算 */
double daikei(double y[N], double h)
{
    int i;
    double s, value;
    s = (y[0]+y[N])/2.0;
    for(i=1; i<N; i++)  s+=y[i];
    value = s*h;
    return value;
}

double func(double x)
{
    double y;
    y = 1.0/x;
```

```
    return y;
}
```

章末問題 (2)

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

#define  N  20

double simpson(double y[N], double dx);
double func(double x);

int main(void)
{
    int i;
    double y[N],h,a=1.0,b=2.0,x,integral;
    h=(b-a)/(double)N;

    for(i=0; i<=N; i++){
        x = a+(double)i*h;
        y[i] = func(x);
    }
    integral = simpson(y, h);
    printf("value = %f¥n", integral);
    return 0;
}

/* シンプソンの公式による計算 */
double simpson(double y[N], double h)
{
    int i;
    double s, s1, s2;
    s1 = y[1];
    s2 = 0.0;

    for(i=2; i<=N-2; i+=2){
        s1 += y[i+1];
        s2 += y[i];
    }
    s = (y[0]+y[N]+4.0*s1+2.0*s2)*h/3.0;
    return s;
```

```
}

double func(double x)
{
    double y;
    y=1.0/x;
    return y;
}
```

第 12 章

章末問題 (1)

```
#include <stdio.h>

#define  JMAX  50
#define  N  10 /* 領域の分割数 */
#define  Nt 10 /* 時間の分割数 */

int main(void)
{
    int i, j, n;
    double u0[N+1], u1[N+1], u2[N+1], ht, hx, x;

    hx = 1.0/(double)N; /* 領域の刻み幅 */
    ht = 1.0/(double)Nt; /* 時間の刻み幅 */

    /* 初期値 (t=t0) の設定 */
    for(i=0, x=0.0; x<0.5; x+=hx){
        u1[i]=2.0*x;
        i++;
    }
    for(x=0.5; x<1.00001; x+=hx){
        u1[i]=2.0*(1.0-x);
        i++;
    }

    u0[0] = 0.0;
    u0[N] = 0.0;
    for(i=1; i<N; i++){
        u0[i]=u1[i]
             +ht*ht*(u1[i+1]-2.0*u1[i]+u1[i-1])/(2.0*hx*hx);
```

```
    }

    u2[0] = 0.0;
    u2[N] = 0.0;
    for(j=0; j<JMAX; j++){
        for(i=1; i<N; i++){
            u2[i]=2.0*u1[i]
            -u0[i]+ht*ht*(u1[i+1]-2.0*u1[i]+u1[i-1])/(hx*hx);
        }
        for(i=0; i<=N; i++){
            printf("%f  ", u2[i]);
        }
        printf("¥n¥n");

        for(i=0; i<=N; i++){
            u0[i] = u1[i];
            u1[i] = u2[i];
        }
    }
    return 0;
}
```

第　13　章

章末問題 (1)

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

#define N   30000    /* 試行回数 */
#define PI  3.1415926535    /* 円周率 */

int main(void)
{
    int     i, n=0;
    double  a, L, y, y1, y2, r, x, judge, P, result;

    a = 1.0;  /* 平行線間隔 */
    L = 0.8;  /* 針の長さ */
```

```
    /* 乱数の初期化 */
    srand(time(NULL));

    i = 1;
    while(i <= N){
        r = (double)rand()/(double)RAND_MAX;
        y = a + 2.0 * a * r;  /* 針の中心位置 */
        r = (double)rand()/(double)RAND_MAX;
        x = PI * r;
        y1 = y - L * sin(x);  /* 針の端の位置（No.1) */
        y2 = y + L * sin(x);  /* 針の端の位置（No.2)*/
        judge = (y1 - 2.0 * a) * (y2 - 2.0 * a);
        if(judge < 0) n++;
        i++;
    }
    P = (double)n / (double)N;
    result = 2.0 * L / (a * P);  /* 円周率の計算 */
    printf("PAI = %f¥n", result);
    return 0;
}
```

章末問題 (2)

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <math.h>
#define N   30000    /* 試行回数 */
double f(double x);
int main(void)
{
    int     i;
    double  a, b, r, sum=0.0, I;
    a = 0.0;
    b = 1.0;

    /* 乱数の初期化 */
    srand(time(NULL));
    i = 1;
    while(i <= N){
        r = (double)rand()/(double)RAND_MAX;
        sum += f(r);
```

```
        i++;
    }

    I = (b - a) * sum / (double)N;
    printf("value = %f¥n", I);
    return 0;
}

double f(double x)
{
    double y;
    y = sqrt(1.0 - x * x);
    return y;
}
```

章末問題 (3)

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

#define N   100         /* 試行回数 */
#define n   10           /* 1試行あたりの歩数 */
#define SHIFT   15

int main(void)
{
    int i, j, x, nx[SHIFT]={0}, par;
    double  r, sum=0.0, ave, sqrsum=0.0, var;

    /* 乱数の初期化 */
    srand(time(NULL));

    for(j=1; j<=N; j++){
        i = 1;
        x = 0;
        while(i <= n){
            r = (double)rand()/(double)RAND_MAX;
            if(r < 0.5){
                x--;
            }else if(r > 0.5){
                x++;
```

```
            }
            i++;
        }

        sum+=(double)x;
        sqrsum+=(double)(x*x);

        if(x>=-(int)(SHIFT/2) && x<=(int)(SHIFT/2)){
            par = x + (int)(SHIFT / 2);
            nx[par]++;
        }
    }

    ave = sum/(double)N;
    var = sqrsum/(double)N-ave*ave;

    for(i=0; i<SHIFT; i++){
        printf("Nx[%d] = %f¥n",
               i-(int)(SHIFT / 2), (double)nx[i]/(double)N);
    }

    printf("¥naverage = %f¥n", ave);
    printf("variance = %f¥n", var);
    return 0;
}
```