Московский ордена Ленина, ордена Октябрьской Революции и ордена Трудового Красного Знамени Государственный Технический Университет им. Н. Э. Баумана

Факультет ***«Робототехники и комплексной автоматизации»***

Кафедра ***«Системы автоматизированного проектирования»***

ОТЧЕТ

По лабораторной работе

По курсу «Модели и методы проектных решений»

Вариант 20

Выполнил студент группы РК6-61

Яблоков В. Е.

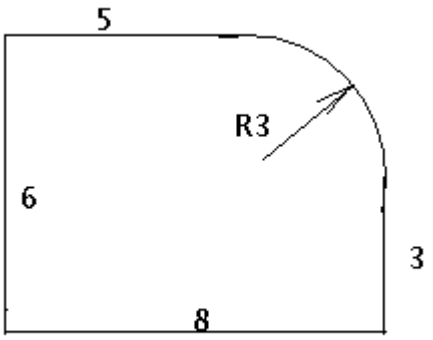
Проверил: к.т.н., доцент кафедры РК-6

Трудоношин В. А.

Москва, 2014

Задание

С помощью неявной разностной схемы решить нестационарное уравнение теплопроводности для пластины, изображенной на рис., там же указаны размеры сторон.



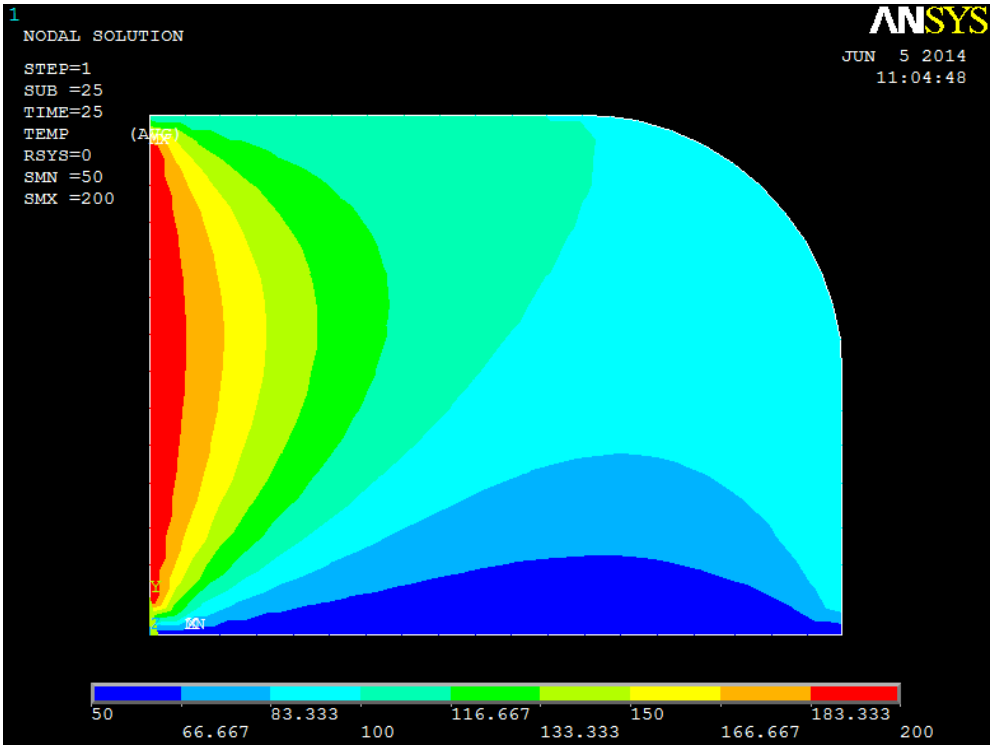
Начальное значение температуры пластины - 0 градусов.

Граничные условия следующие: слева 200 градусов, внизу 50, справа, на округлой границе и вверху 100.

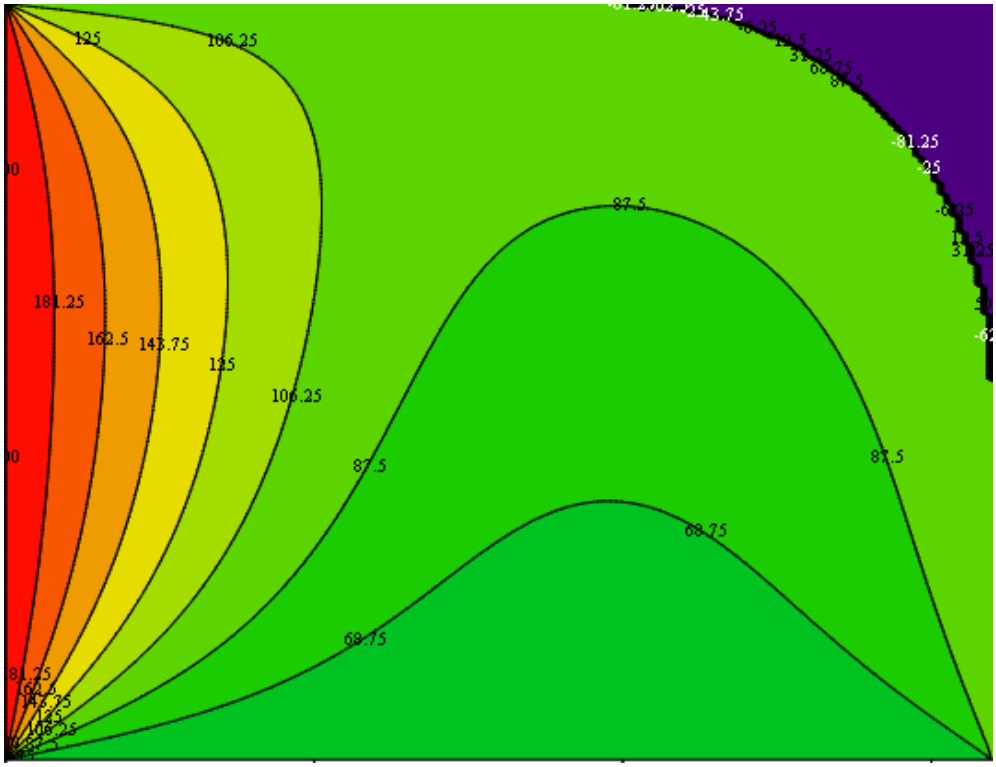
При выводе результатов показать динамику изменения температуры (например, с помощью цветовой гаммы). Отчет должен содержать: текст программы, рисунок объекта с распределением температуры в момент времени 25 сек, сравнение результатов расчета с результатами, полученными с помощью пакета ANSYS .

Полученные результаты

Результат работы программного комплекса ANSYS:



Результат выполнения лабораторной работы:



Исходный код программы:

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

double** A;
double** An;
const double botL = 8.0;
const double leftL = 6.0;
const double rightL = 3.0;
const double topL = 5.0;
const double radius = 3.0;
int Nx, Ny, Nt;
double hx, hy, ht;
double a=0.2;
double leftTemp = 200;
double rightTemp = 100;
double botTemp = 50;
double topTemp = 100;
double edgeTemp = 100;
int bndJ, bndK;
int stepNo;
double k1=1, k2=1, k12=1;

int getEdgeK(int j)
{
    if(j == Nx)
        return bndK;
    if(j > bndJ)
    {
        int edgeK = floor((sqrt(radius*radius - ((double)j*hx -
(double)bndJ*hx)*((double)j*hx - (double)bndJ*hx)) +
(double)bndK*hy) / hy);
        return edgeK;
    }
    else
        return Ny;
}

double getNext(int j, int k, double** Tn, double** T)
{
    if (j == 0 || k == 0) return T[j][k];
    else {
        return a*ht*(
                    (T[j+1][k] - 2*T[j][k] + T[j-1][k]) /
(hx*hx) +
                (T[j][k+1] - 2*T[j][k] + T[j][k-1]) / (hy*hy)
                ) + T[j][k];
    }
}
```

```
void makeStep(double** newMx, double** oldMx)
{
    for (int j = 0; j < Nx; j++){
        for (int k = 0; k < getEdgeK(j+1); k++){
            newMx[j][k] = getNext(j, k, newMx, oldMx);
        }
    }
}

void printMx(double** mat)
{
    FILE* s = fopen("mx.txt", "w");

    for (int j = 0; j < Nx+1; j++){
        for (int k = 0; k < Ny+1; k++)
            fprintf(s, "%f ", mat[j][k]);
        fputc('\n', s);
    }
    putchar('\n');
    fclose(s);
}

void clearMx(double** A)
{
    for (int j = 0; j < Nx+1; j++)
        for (int k = 0; k < Ny+1; k++)
            A[j][k] = 0;
}

void copyMx(double** dst, double** src)
{
    for (int j = 0; j < Nx+1; j++)
        for (int k = 0; k < Ny+1; k++)
            dst[j][k] = src[j][k];
}

int main(int argc, char *argv[])
{
    using namespace std;

    Nx = 160; Ny = 240;
    ht = 0.001;
    Nt = 25000;

    hx = botL / Nx;
    hy = leftL / Ny;
    bndJ = (int)Nx - (botL - topL)/hx;
    bndK = (int)Ny - (leftL - rightL)/hy;

    A = new double*[Nx+1];
    for (int i = 0; i < Nx+1; i++)
```

```
        A[i] = new double[Ny+1];

    for (int j = 0; j < Nx+1; j++){
        for (int k = 0; k < Ny+1; k++){
            A[j][k] = 0.0;
        }
    }

    An = new double*[Nx+1];
    for (int i = 0; i < Nx+1; i++)
        An[i] = new double[Ny+1];

    for (int j = 0; j < Nx+1; j++){
        for (int k = 0; k < Ny+1; k++){
            An[j][k] = -100.0;
        }
    }

    for (int j =0; j < Nx+1; j++){
        A[j][0] = botTemp;
        An[j][0] = botTemp;
    }
    for (int k = 0; k < Ny+1; k++){
        A[0][k] = leftTemp;
        An[0][k] = leftTemp;
    }

    for (int j = 0; j < Nx; j++) {
        for (int k = getEdgeK(j); k >= getEdgeK(j+1); k--) {
            A[j][k] = topTemp;
            An[j][k] = topTemp;
        }
    }

    for (int k = 0; k < bndK+1; k++){
        A[Nx][k] = rightTemp;
        An[Nx][k] = rightTemp;
    }

    stepNo = 0;
    for (stepNo=0; stepNo < Nt; stepNo++){
        makeStep(An, A);
        copyMx(A, An);
    }
    printMx(A);
    printf("Ready.\n");

    return 0;
}
```