Лабораторная № 4
«Триангуляция цилиндрической поверхности»

Рассмотрим поверхность S, заданную уравнениями

$$\begin{cases} x = & \rho(\theta)\cos\theta \\ y = & \rho(\theta)\sin\theta \\ z = & h, \end{cases}$$

где $\theta \in [0, 2\pi]$, $h \in [0, H]$, $H > 0$ и $\rho(\theta)$ – 2π-периодическая функция. Построим триангуляцию данной поверхности. Для этого зафиксируем натуральные числа n и m. Разобьем отрезок $[0, 2\pi]$ на n отрезков точками $\theta_i = i2\pi/n$, $i = 0, ..., n$, а отрезок $[0, H]$ точками $h_j = jH/m$, $j = 0, ...m$. На поверхности S этим значениям параметров θ и h соответствует набор точек $\{P_{ij} = (x_i, y_i, z_j)\}$, где

$$x_i = \rho(\theta_i)\cos\theta_i,\ y_i = \rho(\theta_i)\sin\theta_i,\ z_j = h_j.$$

Отметим, что в данном наборе точек можно исключить точки, соответствующие значению индекса $i = n$, так как они совпадут с точками для индекса $i = 0$. Для $i = 0, ..., n-2$ рассмотрим треугольники

$$\Delta P_{ij}P_{i+1,j}P_{i,j+1}, \quad \Delta P_{i,j+1}P_{i+1,j}P_{i+1,j+1}.$$

Последний столбец треугольников должен соединиться с первым столбом. Поэтому, для $i = n-1$ получим треугольники

$$\Delta P_{ij}P_{0,j}P_{i,j+1}, \quad \Delta P_{i,j+1}P_{0,j}P_{0,j+1}.$$

Итак, общее количество точек триангуляции равно $p = n(m+1)$, а общее количество треугольников равно $N = 2nm$. Сплошная нумерация точек осуществляется по формуле

$$Numb\_point(i, j) = (m+1)i + j,\ i = 0, ...n-1,\ j = 0, ..., m.$$

Так как осуществляется триангуляция поверхности, то для сохранения ее в файл формата `STL` нужно соответствующие изменения внести в функцию `savetostl()`. Новую функцию обозначим `surfacesavetostl`:

```
int surfacesavetostl(double X[p], double Y[p], double Z[p], int T[N][3],
int, int);
int surfacesavetostl(double X[p], double Y[p], double Z[p], int T[N][3],
int p, int N)
{
int k=0, ia, ib, ic;
double v, x1, y1, z1, x2, y2, z2,  nx, ny, nz;
```

```
fstream TRstl;
remove("triangulation.stl");
TRstl.open("triangulation.stl",ios::out|ios::app);

TRstl << "solid <Triangulation>\n";

for(k=0;k<N;k++)
{
ia=T[k][0];
ib=T[k][1];
ic=T[k][2];
x1=X[ib]-X[ia];
y1=Y[ib]-Y[ia];
z1=Z[ib]-Z[ia];
x2=X[ic]-X[ia];
y2=Y[ic]-Y[ia];
z2=Z[ic]-Z[ia];

nx=(y1*z2-y2*z1);
ny=(z1*x2-x1*z2);
nz=(x1*y2-x2*y1);
v=sqrt(nx*nx+ny*ny+nz*nz);
nx=nx/v;
ny=ny/v;
nz=nz/v;
TRstl << "facet normal "<< nx <<" "<< ny <<" "<< nz <<"\n";
TRstl <<"outer loop\n";
TRstl <<"vertex ";
TRstl << X[ia] <<" "<< Y[ia] <<" "<< Z[ia] <<"\n";
TRstl << "vertex ";
TRstl << X[ib] <<" "<< Y[ib] <<" "<< Z[ib] <<"\n";
TRstl << "vertex ";
TRstl << X[ic] <<" "<< Y[ic] <<" "<< Z[ic] <<"\n";
TRstl << "endloop\n";
TRstl << "endfacet\n";
}
TRstl <<"endsolid";
TRstl.close();

return 0;
}
```

Задание. Написать программу, которая вычисляет триангуляцию цилиндрической поверхности и сохраняет ее в массивы $X[p], Y[p], Z[p]$ и $T[N][3]$. Выполнить сохранение триангуляции в файл формата `STL` и с помощью ресурса `https://www.viewstl.com/classic/` визуально проверить правильность расчетов. В качестве примера можно рассмотреть круговой цилиндр $\rho(\theta) \equiv R > 0$, R – радиус цилиндра.