1.

```
function y = Gauss_Chebyshev(N)
i = 1:N;
t = cos((2*i - 1)*pi/(2*N));
w = pi/N;
f = w*atan(t)./t;
sums = cumsum(f);
z = sums(1:i:length(f));
y = z(N)/2;
```

```
function test_Gauss_Chebyshev()
N = 1;
diff = 1;
while diff >= 1e-4
    y_comp = Gauss_Chebyshev(N);
    y_true = pi/2*log(1+sqrt(2));
    diff = abs(y_comp - y_true);
    d(N) = diff;
    N = N+1;
end
disp(N-1) % min N
disp(d(N-1))
```

```
>> test_Gauss_Chebyshev
     5

     4.118198329061684e-05
```

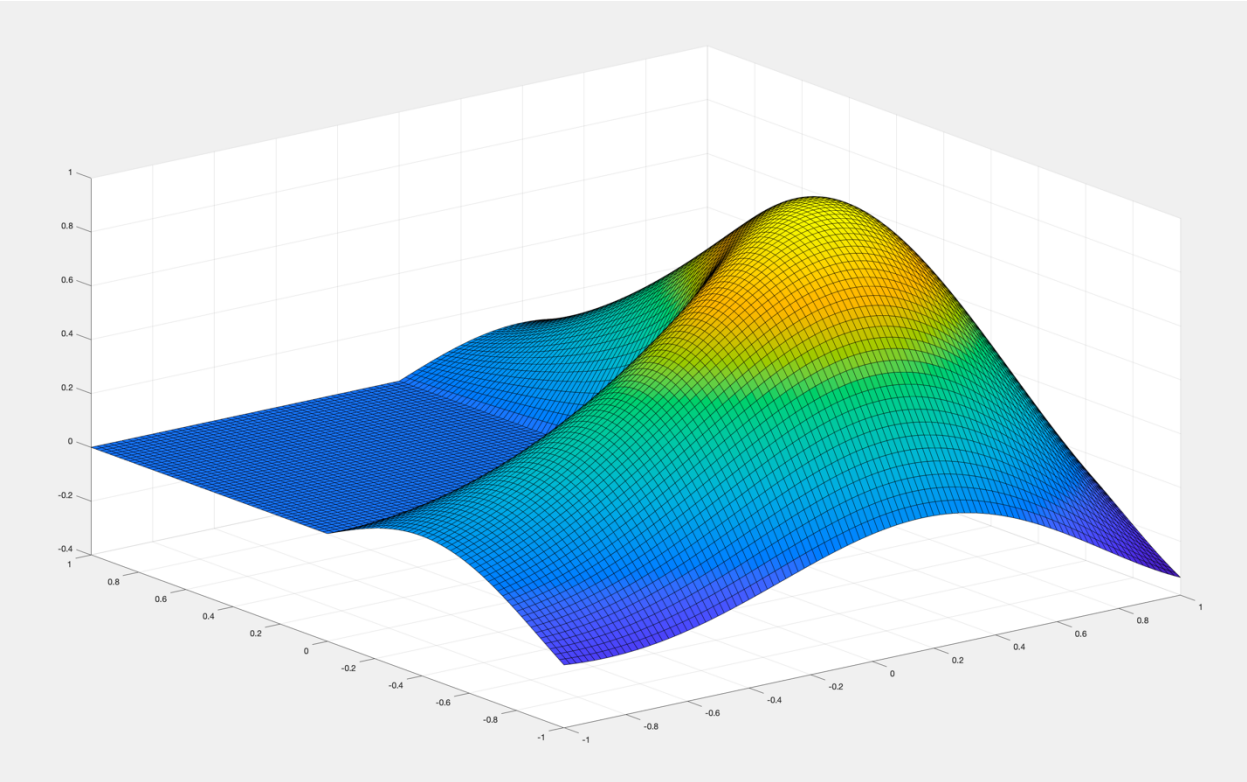
2.

```
function s = simpsons_rule_double(f,a,b,c,d,n,m)
%input is # points minus 1
hx = (b-a)/n;
hy = (d-c)/m;
s = 0;
for i = 0:n
    if (i == 0 || i == n)
        p = 1;
    elseif (rem(i,2) ~= 0)
        p = 4;
    else
        p = 2;
    end
    for j = 0:m
        if (j == 0 || j == m)
            q = 1;
        elseif (rem(j,2)~= 0)
            q = 4;
        else
            q = 2;
        end
        x = a + i*hx;
        y = c + j*hy;
        s = s + p*q*f(x,y);
    end
end
s = (hx*hy)/9 * s;
end
```

```
function test_simpsons_rule_double()
z1 = @(x,y) (1 - x.^2)*(1 - y.^2);
z2 = @(x,y) (1 - (cos(pi*x/2)).^2)*(1 - (cos(pi*y/2)).^2);
a = -1;
b = 1;
c = -1;
d = 1;
n = 100;
m = 100;
s_z1_true = 16/9;
s_z2_true = 1;
s1 = simpsons_rule_double(z1,a,b,c,d,n,m);
s2 = simpsons_rule_double(z2,a,b,c,d,n,m);
fprintf('s_z1_true is %f, s1 via simpsons is %f, error
is %f\n',s_z1_true,s1,abs(s_z1_true - s1))
fprintf('s_z2_true is %f, s2 via simpsons is %f, error
is %f\n',s_z2_true,s2,abs(s_z2_true - s2))
```

```
>> test_simpsons_rule_double
s_z1_true is 1.777778, s1 via simpsons is 1.777778, error is
0.000000
s_z2_true is 1.000000, s2 via simpsons is 1.000000, error is
0.000000
```

```
function z = LOGO()
N=101;
s = linspace(-1, 1, N);
[x, y] = meshgrid(s, s);
z = membrane(1, (N-1)/2);
surf(x, y, z)
```



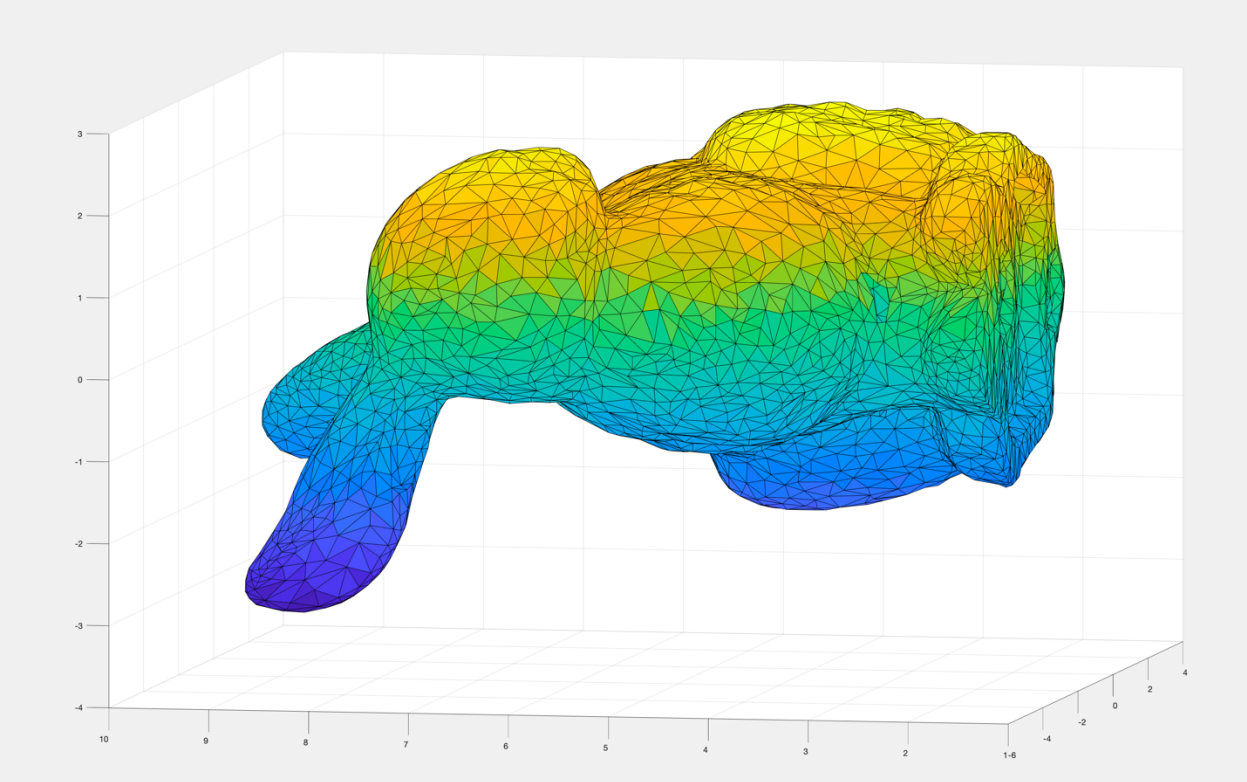
```
function s = IntegralLOGO()
N = 101;
z = membrane(1, (N-1)/2);
a = -1;
b = 1;
c = -1;
d = 1;
n = 101;
m = 101;


hx = (b-a)/(n-1);
hy = (d-c)/(m-1);
s = 0;
for i = 1:n
    if (i == 1 || i == n)
        p = 1;
    elseif (rem(i,2) ~= 0)
        p = 2;
    else
        p = 4;
    end
    for j = 1:m
        if (j == 1 || j == m)
            q = 1;
        elseif (rem(j,2)~= 0)
            q = 2;
        else
            q = 4;
        end
        s = s + p*q*z(j,i);
    end
end
format long
s = (hx*hy)/9 * s;



>> IntegralLOGO

ans =

   1.048683520569986
```

3.

```
function area = tri(S)
% S = stlread("file name");
pts = S.Points; % n by 3 marix
tris = S.ConnectivityList;
trisurf(S)

l = size(tris);
lth = l(1);
area = 0;
for i = 1:lth
    t1 = tris(i,1);
    t2 = tris(i,2);
    t3 = tris(i,3);
    p = [pts(t1,1)-pts(t2,1); pts(t1,2)-pts(t2,2);pts(t1,3)-
pts(t2,3)];
    q = [pts(t1,1)-pts(t3,1); pts(t1,2)-pts(t3,2);pts(t1,3)-
pts(t3,3)];
    a(i) = 0.5*norm(cross(p,q));
    area = area + a(i);
end
```

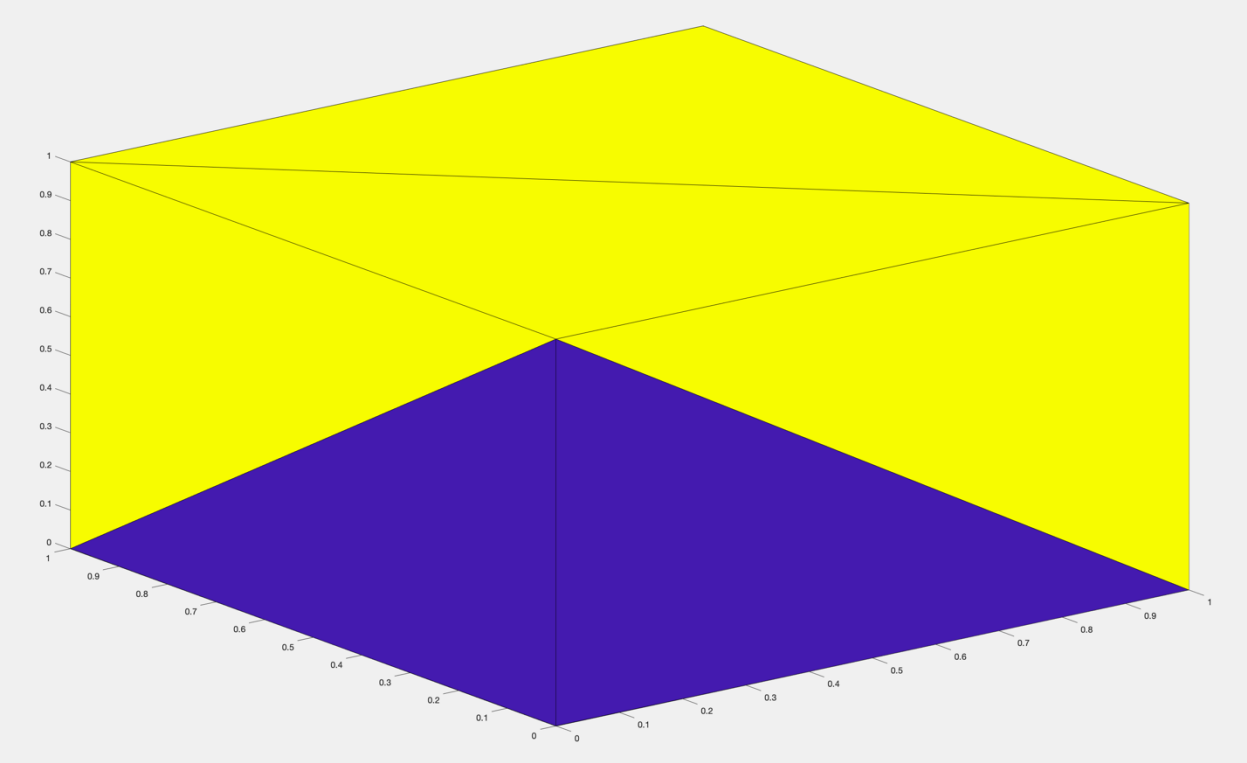
```
function area = P3_bunny()

S = stlread("bunny.stl");
area = tri(S);

end
```

```
>> P3_bunny

ans =

     1.427474195012045e+02
```



```
function area = P3_cube()

S = stlread("cube.stl");
area = tri(S);

end
```

```
>> P3_cube

ans =

     6
```



```
function area = P3_sphere()

S = stlread("sphere.stl");
area = tri(S);
error = abs(area - 4*pi);
disp(error)
end
```

```
>> P3_sphere
   0.059877974459187


ans =

  12.506492639899985
```