In[1]:=
```
(*Mathematica code from MATLAB.
   Below is line-by-line conersion of the
  "Example of MATLAB code for computing Gauss's Law" as given on p. 2055 of Lab
  Manual. The MATLAB lines  are numbered but not evaluated,being inside comments -
 (* ...*). The similar Mathematica lines of code which have identical numbering,
are evaluated immediately after. Specific comments are added after
 the Mathematica command. The entire notebook can be activated
 using  the 'Evaluate Notebook' from the the 'Evaluate' panel,
or using one input at a time with 'Shift-enter'. A more general
  Mathematica aproach to Gauss problem is used in the Lab202Gauss.nb  *)

(*1. MATLAB:   >>clear all*)
(*Mathematica: *)Clear[x, y, z];
```

In[3]:=
```
(*2. MATLAB: >>diary ('Lab202')% to save your work with file Lab202
3. MATLAB: >>diary on *)
(*Mathematica: lines 2,3 are not required in Mathematica since all,
including graphics,  is contained in a single notebook, which is saved upon exit *)
```

In[4]:=
```
(*4. MATLAB >>eps0=8.85*10^-12;k=1/(4*pi*esp0);
*)(*4. Mathematica: *) eps0 = 8.85 * 10^-12;
k = 1 / (4 * Pi * eps0);(*same, Pi with capital*)
```

In[5]:=
```
 (*5. MATLAB: >>q1=9*10^-9;*)
(*5. Mathematica:*)q1 = 9 * 10^-9; (*same*)
```

In[6]:=
```
(*6. MATLAB: >>syms x y z;*)
(*Mathematica: line 6. is not required*)

(*7. MATLAB: >>
  E= @(x,y,z) k*q1/((x-0.1)^2+(y-0.2)^2+(z-0.3)^2)^(3/2)*[x-0.1,y-0.2,z-0.3];*)
(*7. Mathematica:*)Clear[e];
e[x_, y_, z_] :=
 k * q1 / ((x - 0.1) ^2 + (y - 0.2) ^2 + (z - 0.3) ^2) ^ (3 / 2) * {x - 0.1, y - 0.2, z - 0.3};
(*Note: lower case for electric field "e" since "E" is taken;
arguments of e are added as [x_,y_,z_] (instead of @(x,y,z)),
and there are curly brackets around vector components*)
```

In[7]:=

In[8]:=
```
(*8. MATLAB: >>I=[1,0,0];J=[0,1,0];K=[0,0,1];*)
(*8. Mathematica:*)
I1 = {1, 0, 0};
J1 = {0, 1, 0};
K1 = {0, 0, 1}; (*changed to I1, J1 and K1 since "I" and "K" are taken;
changed to curly brackets around vectors*)
```

In[9]:=
```
(*9-14. MATLAB:
    Etop=dot(K,E(x,y,1));
Ebottom=dot(-K,E(x,y,-1));
Eleft=dot(-J,E(x,-1,z));
Eright=dot(J,E(x,1,z));
Efront=dot(I,E(1,y,z));
Eback=dot(-I,E(-1,y,z));*)
(*9-14. Mathematica:*)
Etop = K1.e[x, y, 1];
Ebottom = -K1.e[x, y, -1];
Eleft = -J1.e[x, -1, z];
Eright = J1.e[x, 1, z];
Efront = I1.e[1, y, z];
Eback = -I1.e[-1, y, z];
```

In[15]:=
```
(*15.-20. MATLAB.
    Phitop=int(int(Etop,x,-1,1),y,-1,1);
Phibottom=int(int(Edown,x,-1,1),y,-1,1);
Phileft=int(int(Eleft,x,-1,1),z,-1,1);
Phiright=int(int(Eright,x,-1,1),z,-1,1);
Phifront=int(int(Efront,y,-1,1),z,-1,1);
Phiback=int(int(Eback,y,-1,1),z,-1,1);
Note: in MATLAB the integral is still symbolic;
to evaluate it numericaly you should either use the 'vpa' command as below,
or just write "double(...)" around the symbolic expression*)
```

In[17]:=
```
(*15-20. Mathematica:*)
Phitop = NIntegrate[Etop, {x, -1, 1}, {y, -1, 1}];
Phibottom = NIntegrate[Ebottom, {x, -1, 1}, {y, -1, 1}];
Phileft = NIntegrate[Eleft, {x, -1, 1}, {z, -1, 1}];
Phiright = NIntegrate[Eright, {x, -1, 1}, {z, -1, 1}];
Phifront = NIntegrate[Efront, {y, -1, 1}, {z, -1, 1}];
Phiback = NIntegrate[Eback, {y, -1, 1}, {z, -1, 1}];
```

```
(*Note: only one NIntegrate  for a double integral;
each domain is in curly brackets. You could use Integrate
 for symbolic computations with an explicit (exact) answer,
but those are possible only in relatively simple cases,
e.g. for y,z=0 – see  Lab202Gauss.nb *)
```

In[24]:=
```
(*21. MATLAB.
 >>vpa(Phitop,10)
ans=233.9349563
*)
```

In[25]:=
```
(*21. Mathematica:*)
Phitop (*already has been evaluated*)
```

Out[25]=
```
233.935
```

```
(* Mathematica keeps more digits in memory than it shares on the screen;
you will see 233.93495673594776` if you copy and paste the output.  *)
```

In[27]:=
```
(*22–26. MATLAB
   >>vpa(Phibottom,10)
ans=121.2432412
   >>vpa(Phileft,10)
ans=131.9684772
   >>vpa(Phiright,10)
ans=204.4187722
    >>vpa(Phifront,10)
ans=180.4038896
   >>vpa(Phiback,10)
ans=144.9798159
   >>All_Phi=Phitop+Phibottom+Phileft+Phiright+Phifront+Phiback;
>>vpa(All_Phi,10)
ans=1016.949152
 >>vpa(esp0*All_Phi,5)
ans=.90001e-8
*)
```

In[28]:=
```
(*22–26. Mathematica: *)
```

In[29]:=
```
Phibottom
```

Out[29]=
```
121.243
```

In[30]:=
```
Phiright
```

Out[30]=
```
204.419
```

In[31]:=
```
Phileft
```
Out[31]=
```
131.968
```

In[32]:=
```
Phifront
```
Out[32]=
```
180.404
```

In[33]:=
```
Phiback
```
Out[33]=
```
144.98
```

In[34]:=
```
AllPhi = Phitop + Phibottom + Phileft + Phiright +
  Phifront + Phiback (*removed underscore in the name AllPhi*)
```
Out[34]=
```
1016.95
```

In[35]:=
```
eps0 * AllPhi
```
Out[35]=
```
9. × 10^-9
```

In[36]:=
```
(*which is the charge, confirming Gauss*)
```

In[37]:=