Input the number of boundary elements 'nb', the number of domain nodes 'nd' and the material property a0

```
nb = 4; nd = 5; a0 = 1.;
```

Input the coordinates of the of ends of boundary elements (xe,ye) and coords of domain nodes (xd,yd)

```
xe = {1., 1., 0., 0.}; ye = {0., 1., 1., 0.};
xd = {0.25, 0.75, 0.5, 0.25, 0.75}; yd = {0.25, 0.25, 0.5, 0.75, 0.75};
```

Input the type of boundary conditions 'tbc' (1 for Dirichlet & 2 for Nuemann) and their values 'vbc'

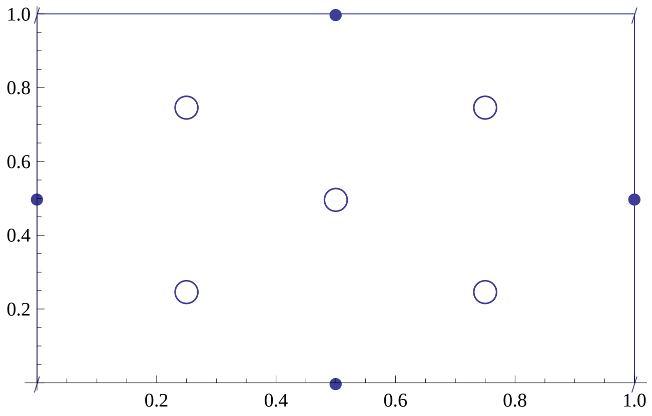
```
tbc = {2, 1, 2, 1}; vbc = {0., 1., 0., 0.};
```

Input the coordinates of the midpoints of boundary elements (xm,ym)

```
xm = ym = Table[0, {i, 1, nb}];
jb = If[j == 1, nb, j - 1];
Do[xm[[j]] = (xe[[j]] + xe[[jb]]) / 2;
  ym[[j]] = (ye[[j]] + ye[[jb]]) / 2, {j, 1, nb}];
```

Plot, the geometry of the problem showing the boundary elements & nodes and the domain nodes

```
dat1 = Table[{xe[[i]], ye[[i]]}, {i, 1, nb}];
dat2 = Table[{xm[[i]], ym[[i]]}, {i, 1, nb}];
dat3 = Table[{xd[[i]], yd[[i]]}, {i, 1, nd}];
p1 = ListPlot[dat1, PlotStyle → PointSize[0.02],
   PlotMarkers → "/", Joined → True];
p2 = ListPlot[dat2, PlotStyle → PointSize[0.02], PlotMarkers → "●"];
p3 = ListPlot[dat3, PlotStyle → PointSize[0.02], PlotMarkers → "○"];
Show[p1, p2, p3]
```



Apply the boundary conditions

```
um = Array[u, {nb}]; qm = Array[q, {nb}];
Do[If[tbc[[i]] == 1, um[[i]] = vbc[[i]], qm[[i]] = vbc[[i]]], {i, 1, nb}]
```

Print the boundary data

```
boundarydat = Table[{i, xe[[i]], ye[[i]],
    xm[[i]], ym[[i]], tbc[[i]], vbc[[i]]}, {i, 1, nb}];
TableForm[boundarydat, TableHeadings -> {None,
   {"Element No. ", "XE", "YE", "XM", "YM", "BC-Type", "BC-Value"}}]
```

```
Element No.   XE  YE  XM   YM   BC-Type  BC-Value
--------------------------------------------------
1             1.  0.  0.5  0.   2        0.
2             1.  1.  1.   0.5  1        1.
3             0.  1.  0.5  1.   2        0.
4             0.  0.  0.   0.5  1        0.
```

Compute elements of Gb, Gd,Hb & Gd

```
Hb = Gb = Table[0, {i, 1, nb}, {j, 1, nb}];

Hd = Gd = Hd1 = Hd2 = Gd1 = Gd2 = Table[0, {i, 1, nd}, {j, 1, nb}];

r = Sqrt[(x - xi)^2 + (y - yi)^2];

      Log[1 / r]
us = -----------;
       2 π a0
qs = a0 * (D[us, x] * n1 + D[us, y] * n2);

usxi = D[us, xi];

usyi = D[us, yi];

qsxi = D[qs, xi];

qsyi = D[qs, yi];

<< NumericalDifferentialEquationAnalysis`;

np = 10; p = w = Table[Null, {np}];

Do[p[[i]] = GaussianQuadratureWeights[np, -1, 1][[i, 1]], {i, 1, np}]

Do[w[[i]] = GaussianQuadratureWeights[np, -1, 1][[i, 2]], {i, 1, np}]

Int[f_, z_] := Sum[(f /. z → p[[i]]) w[[i]], {i, 1, np}]
```

```
Do[xb = 1 / 2 * (xe[[jb]] * (1 - z) + xe[[j]] * (1 + z));

  yb = 1 / 2 * (ye[[jb]] * (1 - z) + ye[[j]] * (1 + z));

  L = √((xe[[j]] - xe[[jb]])^2 + (ye[[j]] - ye[[jb]])^2) ;

  ds = L / 2;

  nx = (ye[[j]] - ye[[jb]]) / L;

  ny = (xe[[jb]] - xe[[j]]) / L;

  Do[Gb[[i, j]] =
    Int[us * ds /. {x → xb, y → yb, xi → xm[[i]], yi → ym[[i]]}, z];

   Hb[[i, j]] = Int[qs * ds /. {x → xb, y → yb, n1 → nx,
       n2 → ny, xi → xm[[i]], yi → ym[[i]]}, z], {i, 1, nb}];

  Do[Gd[[i, j]] =
    Int[us * ds /. {x → xb, y → yb, xi → xd[[i]], yi → yd[[i]]}, z];

   Hd[[i, j]] = Int[qs * ds /.
      {x → xb, y → yb, n1 → nx, n2 → ny, xi → xd[[i]], yi → yd[[i]]}, z];

   Gd1[[i, j]] =
    Int[usxi * ds /. {x → xb, y → yb, xi → xd[[i]], yi → yd[[i]]}, z];

   Hd1[[i, j]] = Int[qsxi * ds /.
      {x → xb, y → yb, n1 → nx, n2 → ny, xi → xd[[i]], yi → yd[[i]]}, z];

   Gd2[[i, j]] =
    Int[usyi * ds /. {x → xb, y → yb, xi → xd[[i]], yi → yd[[i]]}, z];

   Hd2[[i, j]] = Int[qsyi * ds /. {x → xb, y → yb, n1 → nx, n2 → ny,
       xi → xd[[i]], yi → yd[[i]]}, z];, {i, 1, nd}], {j, 1, nb}];

Do[Gb[[i, i]] = L / (2 * π * a0) * (Log[2 / L] + 1.), {i, 1, nb}]

Do[Hb[[i, i]] = -Sum[Hb[[i, k]], {k, 1, nb}] + Hb[[i, i]], {i, 1, nb}]
```

```
sol = Solve[Hb.um == Gb.qm];

um = um /. sol[[1]];

qm = qm /. sol[[1]];
```

Print the boundary solution

```
BoundarySol = Table[{i, xm[[i]], ym[[i]],
    um[[i]] /. sol[[1]], qm[[i]] /. sol[[1]]}, {i, 1, nb}];
TableForm[BoundarySol, TableHeadings ->
  {None, {"Node No. ", "XM", "YM", "Potential (u)", "Flux (qn)"}}]
```

```
Node No.   XM   YM   Potential (u)  Flux (qn)
1          0.5  0.   0.5            0.
2          1.   0.5  1.             1.17461
3          0.5  1.   0.5            0.
4          0.   0.5  0.             -1.17461
```

Compute & print the domain solution

```
ud = (-Hd.um + Gd.qm);

q1 = (-Hd1.um + Gd1.qm);

q2 = (-Hd2.um + Gd2.qm);

DomainSol =
  Table[{i, xd[[i]], yd[[i]], ud[[i]], q1[[i]], q2[[i]]}, {i, 1, nd}];
TableForm[DomainSol, TableHeadings -> {None, {"Node No. ",
    "XD", "YD", "Potential(u)", "Flux-x(q1)", "Flux-y (q2)"}}]
```

```
Node No.   XD    YD    Potential(u)  Flux-x(q1)  Flux-y (q2)
1          0.25  0.25  0.287249      0.926941    -0.18065
2          0.75  0.25  0.712779      0.927305    0.180286
3          0.5   0.5   0.5           0.905615    2.05965×10^-17
4          0.25  0.75  0.287249      0.926941    0.18065
5          0.75  0.75  0.712779      0.927305    -0.180286
```