```
n = 8; (*number of nodos*)
k = 7; (*vertex degree*)
numperms = 400; (*number of random permutations*)

matrix = Normal[AdjacencyMatrix[RandomGraph[{n, k}]]]; (*random graph*)
graph= AdjacencyGraph[matrix, VertexStyle ->RGBColor[1, .78, .72], EdgeStyle ->Black] (*plot
the random graph*)

(*generate the isomorphisms*)
perms = Cycles[{#}] & /@ RandomChoice[Permutations[Table[i, {i, n}]], numperms];(*generators of
permutations*)
matpermuted = DeleteDuplicates[Permute[Table[Permute[matrix[[i]], #], {i, n}], #] & /@ perms]
;(*permuted matrices*)

complexity = Table[Null, Length[matpermuted] + 1]; (*list for K-complexity (1-D
representation)*)
complexitymat = Table[Null, Length[matpermuted] + 1]; (*list for K-complexity (adjacency
matrix)*)
entropy = Table[Null, Length[matpermuted] + 1]; (*list for entropy values*)

(*complexity of the random graph*)
mat = Flatten[matrix];
If[First[mat] == 0, output = "0" <> ToString[FromDigits[mat]], output =
ToString[FromDigits[mat]]];
complexity[[1]] = StringBDM[output] // N;
complexitymat[[1]] = BDM[rg, 4] // N;
entropy[[1]] = Entropy[output] // N;

(*measure the complexity of the isomorphisms*)
Do[
 rg = matpermuted[[l]];
 mat = Flatten[rg];
 If[First[mat] == 0, output = "0" <> ToString[FromDigits[mat]], output =
ToString[FromDigits[mat]]];
 complexity[[l + 1]] = StringBDM[output] // N;
 complexitymat[[l + 1]] = BDM[rg, 4] // N;
 entropy[[l + 1]] = Entropy[output] // N;
  If[ IsomorphicGraphQ[AdjacencyGraph[matrix], AdjacencyGraph[matpermuted[[l]]]] == False,
Abort[]] (*check isomorphism*)
 , {l, 1, Length[matpermuted]}]

(*plot the results*)
seqcomplexity = DeleteCases[complexity, Null];
matcomplexity = DeleteCases[complexitymat, Null];
ShannonEntropy = DeleteCases[entropy, Null];
ListLinePlot[{Rescale[seqcomplexity, {0, Max[seqcomplexity]}], Rescale[matcomplexity, {0,
Max[matcomplexity]}], Rescale[ShannonEntropy, {0, Max[ShannonEntropy]}]}, AxesLabel ->
Automatic, PlotRange -> {{0, Length[seqcomplexity]}, {0.4, 1.01}}, PlotLegends -> Placed[{"K-
Complexity (1-D representation)", "K-Complexity (adjacency matrix)", "Entropy"}, {.5, .2}],
Frame -> True, GridLines -> Automatic, FrameLabel -> {ToString[Length[seqcomplexity]] <> "
Isomorphic Graphs G'≅G", "C(G')"}]
```