In [26]:
```
import networkx as nx
G = nx.Graph()
G1 = nx.Graph()
G2 = nx.Graph()
nodes = ['A', 'B', "C", "D", "E"]
G.add_nodes_from(nodes)
G1.add_nodes_from(nodes)
G2.add_nodes_from(nodes)
#Test A
G.add_edge("A","B",sign="+")
G.add_edge("A","C",sign="+")
G.add_edge("A","D",sign="+")
G.add_edge("A","E",sign="+")

G.add_edge("B","C",sign="+")
G.add_edge("B","D",sign="+")
G.add_edge("B","E",sign="+")


G.add_edge("C","D",sign="+")
G.add_edge("C","E",sign="+")


G.add_edge("D","E",sign="+")
#Test B
G1.add_edge("A","B",sign="+")
G1.add_edge("A","C",sign="+")
G1.add_edge("A","D",sign="+")
G1.add_edge("A","E",sign="-")

G1.add_edge("B","C",sign="+")
G1.add_edge("B","D",sign="+")
G1.add_edge("B","E",sign="-")


G1.add_edge("C","D",sign="+")
G1.add_edge("C","E",sign="+")


G1.add_edge("D","E",sign="+")
#Test C
G2.add_edge("A","B",sign="-")
G2.add_edge("A","C",sign="-")
G2.add_edge("A","D",sign="+")
G2.add_edge("A","E",sign="+")

G2.add_edge("B","C",sign="+")
G2.add_edge("B","D",sign="-")
G2.add_edge("B","E",sign="+")

G2.add_edge("C","D",sign="+")
G2.add_edge("C","E",sign="-")

G2.add_edge("D","E",sign="-")
```

```
In [27]: def checkba(gra):
             for nd in gra.nodes():
                 tri_point=[]
                 EG1=""
                 EG2=""
                 EG3=""
                 sign_list=[]
                 first_neighbors=[i for i in gra.neighbors(nd)]
                 for nd2 in first_neighbors:
                     second_neighbors=[j for j in gra.neighbors(nd2)]
                     for nd3 in second_neighbors:
                         if nd3 in first_neighbors:
                             tri_point.append(nd)
                             tri_point.append(nd2)
                             tri_point.append(nd3)
                             EG1=gra.edge[tri_point[0]][tri_point[1]]["sign"]
                             EG2=gra.edge[tri_point[1]][tri_point[2]]["sign"]
                             EG3=gra.edge[tri_point[0]][tri_point[2]]["sign"]
                             sign_list.append(EG1)
                             sign_list.append(EG2)
                             sign_list.append(EG3)
                             if EG1==EG2==EG3=="-":
                                 report="Not Balanced due to --- found {}{}{}".format(nd,n
         d2,nd3)
                                 return report
                                 break
                             if sign_list.count("-")==1:
                                     if sign_list.count("+")==2:
                                         report="Not Balanced due to ++- found {}{}{}".for
         mat(nd,nd2,nd3)
                                         return report
                                         break
                         else:continue
             return "Balanced Graph"
         checkba(G1)

Out[27]: 'Balanced Graph'

In [ ]:
```