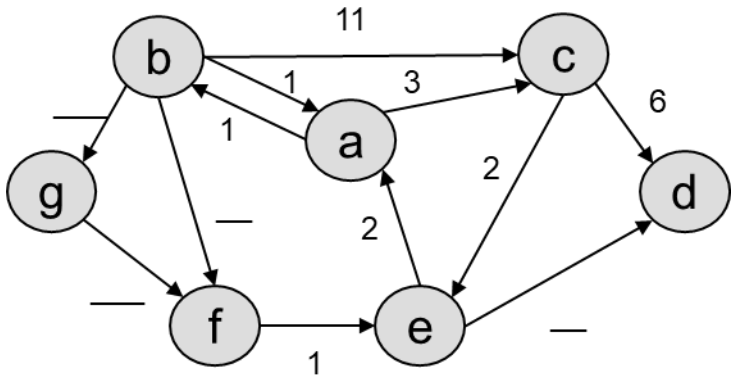
(8 pts.) Tracing and Graphs: Assume you are given the directed graph below, **G**, with **positive edge weights** (though some are missing). Analyze the code below and understand its operation generally. **Now assuming m is an integer, a call is made to** `myst1(G, "b", m)`. **Find appropriate values for the missing edge weights and the value of m that is passed that will produce the following output**. (Note: There may be more than 1 possible answer for certain edge weights or m. List any that works)



Output

```
b a c d
b a c e
b g f e a c
b g f e d
b f e
```

Value of initial m arg: _____

```
// Directed edge struct
struct DEdge {
  string tgt; // target node of edge
  size_t wt;  // weight
};
typedef map<string, vector<DEdge> > GraphT;

void myst1(GraphT& G, string u, int m)
{ vector<string> i;
  set<string> y;
  myst2(G, u, m, i, y);
}
void myst2(GraphT& G, string u, int m, vector<string> items, set<string> y)
{ y.insert(u);
  items.push_back(u);
  bool any = false;
  for( vector<DEdge>::iterator it = G[u].begin(); it != G[u].end(); ++it){
    string v = it->tgt;
    if( y.find( v ) == y.end() ) {
      if( m - (int)it->wt >= 0 ){
        myst2(G, v, m - it->wt, items, y);
        any = true;
  } } }
  if(!any){
    for(size_t k = 0; k < items.size(); k++)
      cout << items[k] << " ";
    cout << endl;
} }
```