1. The set S must be a subset of $A \times B$ (cartesian product of A and B)

$$S \subset (A \times B) \quad \text{where} \quad \subset = \text{subset}$$

2. All elements in set A must have a corresponding element in set B such that the pair (a,b) is an element of S

$$\forall a \in A, \exists b \in B, (a,b) \in S$$

3. All elements in set B must have a corresponding element in set A such that the pair (a,b) is an element of S

$$\forall a \in B, \exists b \in A, (a,b) \in S$$

4. There are no two pairs (a,b) and (a',b') in S such that b' is higher preference for a than b, and a is higher preference for b' than a'

$$\sim \exists (a,b)\ (a',b') \in S \text{ such that } a \rightarrow b' > b \text{ and } b' \rightarrow a > a' \quad \text{where } \rightarrow \text{ means prefer}$$

5. There are no two pairs (a,b) and (a',b') in S such that a' is higher preference for b than a, and b is higher preference for a' than b'

$$\sim \exists (a,b)\ (a',b') \in S \text{ such that } b \rightarrow a' > a \text{ and } a' \rightarrow b > b' \quad \text{where } \rightarrow \text{ means prefer}$$

proof by contradiction

- If both sets contain n elements, the Gale-Shapley algorithm always results in n pairs
  - Assume GS algorithm will not result in n pairs (~p)
  - condition of GS algorithm is that all elements in set A have a corresponding element in set B.
  - Assume set A and set B have n elements.
    According to condition, set A and set B will result in n pairs
  - Therefore ~p is a contradiction, then p must be valid therefore the statement is valid

- The resulting pairs are stable; as in, there are no unstable pairs when the algorithm finishes.
  - Assume that by completion there will be a unstable pair (~p)
  - condition of GS algorithm is that there are no two pairs (a, b) and (a', b') in S such that a' is higher preference for b than a, and b is higher preference for a' than b' (in this case b and a' would break their assigned match in favor of match (a', b))
  - Therefore this would result in a stable pairing
  - Therefore ~p is a contradiction, then p must be valid therefore the statement is valid.

Pseudocode

```
void checkstability (map<string,string>& match, map<string, vector<string>>& listA, listB)
    bool stable = true;
    for ( auto it = match.begin; it != match.end(); ++it) {
        string listNameA = it->first
        string listNameB = it->second
        vector<string> preflist = listA.at(listBName);
        for (auto it = preflist.begin(); it != preflist.end(); ++it) {
            if (*it == listAName)
                break;
            if ( prefers (prefList, *it, listAName) && prefers (listB.at(*it), listBName, match.at(*it))
            {
                stable = false;
            }
        }
    }
    if (stable)
        cout << "Result of verification function: true" << endl;
```