In this exercise, we want to show that the subgraph isomorphism problem is NP-complete. The subgraph isomorphism problem takes two undirected graphs G_1 and G_2 as input and asks whether G_1 is isomorphic to a subgraph of G_2.

Theorem 1. *The subgraph isomorphism problem is NP-complete.*

Proof. A problem is NP-complete if and only if it is in NP and it is NP-hard. First, we show that the subgraph isomorphism problem is in NP, i.e. we can verify a proposed solution in polynomial time. Given graphs G_1 and G_2 and a mapping $f : V_{G_1} \to V_{G_2}$, we must determine the following in polynomial time:

1. Whether f is an injection.
2. Whether for all edges $(u, v) \in E_{G_1}$, $(f(u), f(v)) \in E_{G_2}$.

The function f is an injection if and only if for all $u, v \in V_{G_1}$, $f(u) = f(v)$ implies $u = v$. A simple algorithm to verify this is as follows:

Algorithm 1: Verify Injection

```
Input: Graphs G_1 and G_2, mapping f : V_{G_1} → V_{G_2}
Output: Whether f is an injection
Initialize hash map H;
for u ∈ V_{G_1} do
    if H[f(u)] is defined then
        return false;
    end
    H[f(u)] = u;
end
return true;
```

The runtime of this algorithm is bounded solely by the number of vertices in G_1, i.e. $O(|V_{G_1}|)$, so it runs in polynomial time. To verify the second condition, we simply iterate over all edges in G_1 and check whether the corresponding edges exist in G_2. A simple algorithm to verify this is as follows:

Algorithm 2: Verify Edge Mapping

```
Input: Graphs G_1 and G_2, mapping f : V_{G_1} → V_{G_2}
Output: Whether f maps edges of G_1 to edges of G_2
Initialize hash set S from E_{G_2};
for (u, v) ∈ E_{G_1} do
    if (f(u), f(v)) ∉ S then
        return false;
    end
end
return true;
```

The runtime of this algorithm is bounded by the number of edges in G_1, i.e. $O(|E_{G_1}|)$, so it runs in polynomial time. Therefore, the subgraph isomorphism problem is in NP.

Next, to show that the subgraph isomorphism problem is NP-hard, we reduce the clique problem to the subgraph isomorphism problem. The clique problem takes an undirected graph G and an integer k as input and asks whether G contains a clique of size k, where a clique is a complete subgraph. Given an instance of the clique problem (G, k), we construct the following instance of the subgraph isomorphism problem (G_1, G_2):

Algorithm 3: Construct Subgraph Isomorphism Instance

```
Input: Graph G, integer k
Output: Graphs G_1, G_2
Initialize graph G_1;
for i = 1 to k do
    Add vertex v_i to G_1;
end
for i = 1 to k do
    for j = 1 to k do
        if i ≠ j then
            Add edge (v_i, v_j) to G_1;
        end
    end
end
Initialize graph G_2 as a copy of G;
return (G_1, G_2);
```

This construction algorithm runs in polynomial time due to the following factors:

1. Creating G_1 (a complete graph with k vertices) takes $O(k^2)$ time.
2. G_2 is just a copy of the original graph G, so copying it takes $O(|V_G|+|E_G|)$ time.

Now, we need to show that this reduction is correct, i.e., G has a clique of size k if and only if G_1 is isomorphic to a subgraph of G_2. First we show the forward direction:

(⇒) Assume that G contains a clique of size k. That means there exists some subgraph of G that is a complete graph on k vertices. By construction, G_1 is a complete graph on k vertices, so G_1 is isomorphic to this subgraph of G. Since G_2 is a copy of G, G_1 is isomorphic to a subgraph of G_2. This proves that this reduction preserves the answer in the forward direction.

And now the backward direction:

($\Leftarrow$) Now assume that G_1 is isomorphic to a subgraph of G_2. By construction, G_1 is a complete graph on k vertices. By our initial assumption, there then exists a clique of size k in G_2. Since G_2 is a copy of G, there exists a clique of size k in G, and thus the reduction preserves the answer in both directions.

At this point we have proven the following:

1. There exists a polynomial-time checking algorithm for the subgraph isomorphism problem.
2. The reduction from the clique problem to the subgraph isomorphism problem is correct.
3. The reduction runs in polynomial time.

Therefore, the subgraph isomorphism problem is in NP and is NP-hard, so it is NP-complete. $\square$