Travessia de Grafos

Aplicações

Prof. Edson Alves

2018

Faculdade UnB Gama

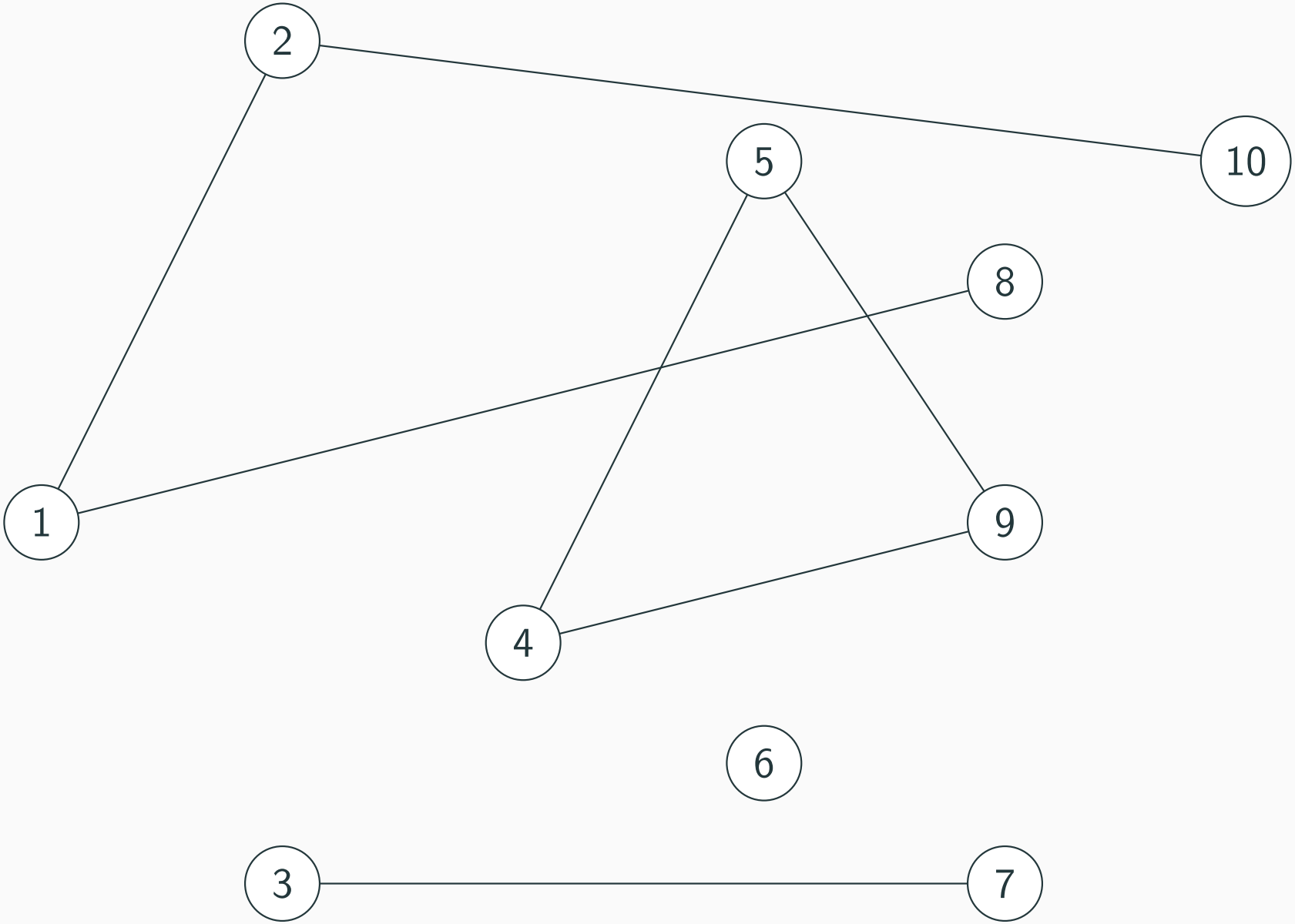
Sumário

Componentes Conectados

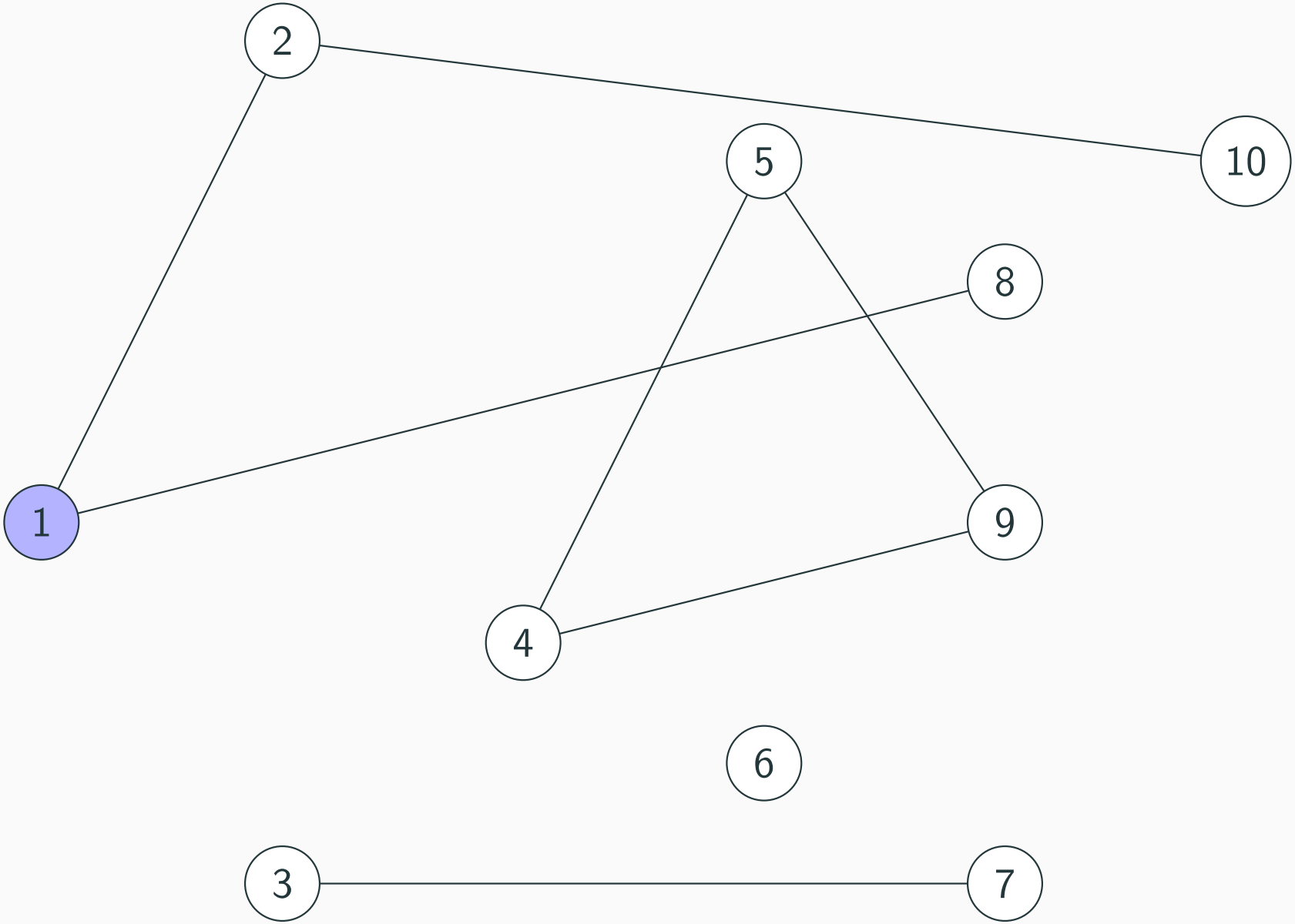
Conectividade de um grafo

- Um grafo G é dito conectado se, para qualquer par de vértices $u, v \in G$, com $u \neq v$, existe ao menos um caminho de u até v
- Uma maneira de se verificar se um grafo é conectado ou não é iniciar uma travessia em um vértice s qualquer
- Se a travessia visitar todos os N nós de G o grafo é conectado
- Caso um ou mais vértices não seja visitado, os nós visitados formam um componente conectado de G
- Para identificar todos os componentes conectados do grafo basta iniciar uma nova travessia em um dos vértices não visitados, enquanto houverem vértices não visitados

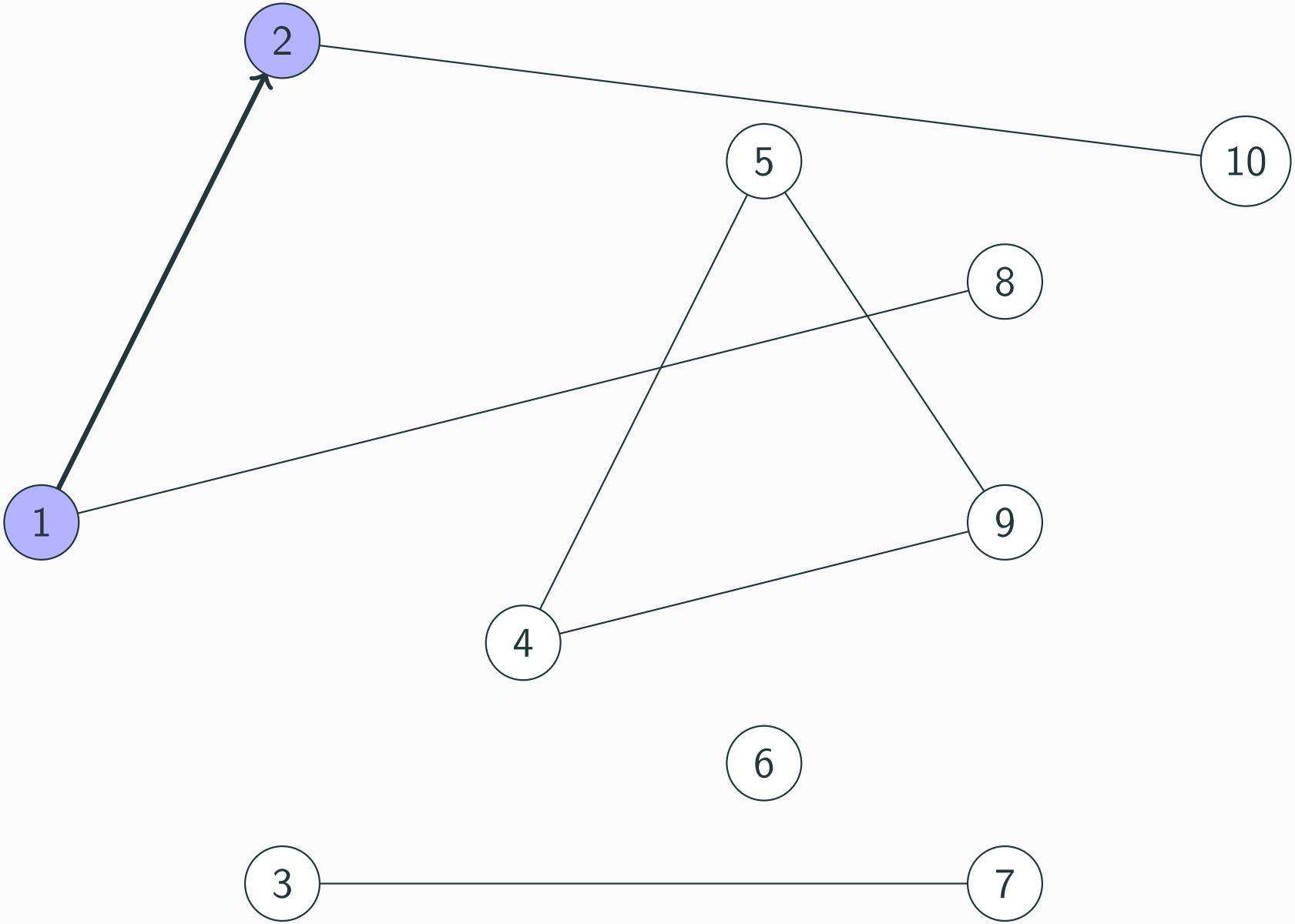
Visualização da identificação dos componentes conectados



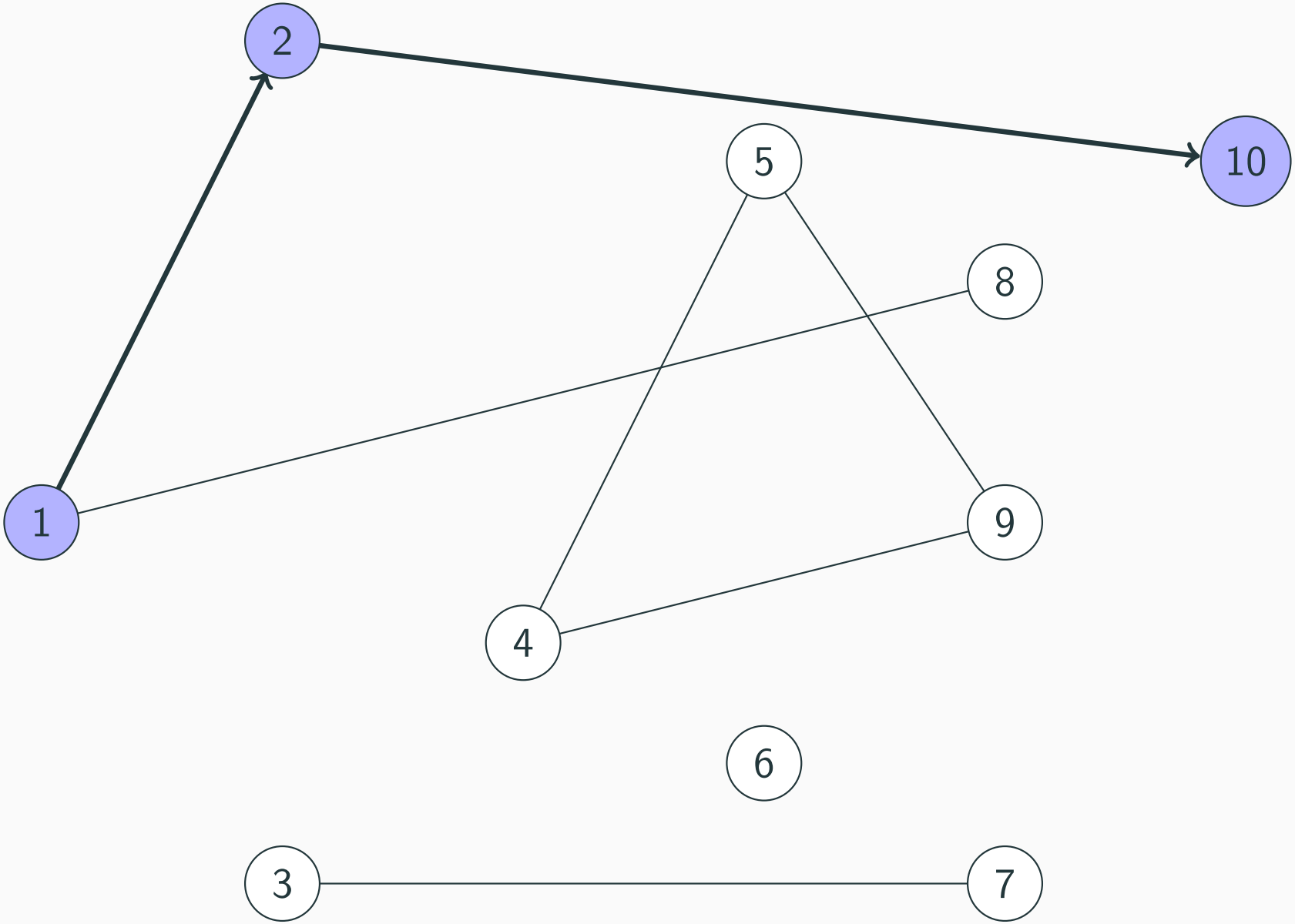
Visualização da identificação dos componentes conectados



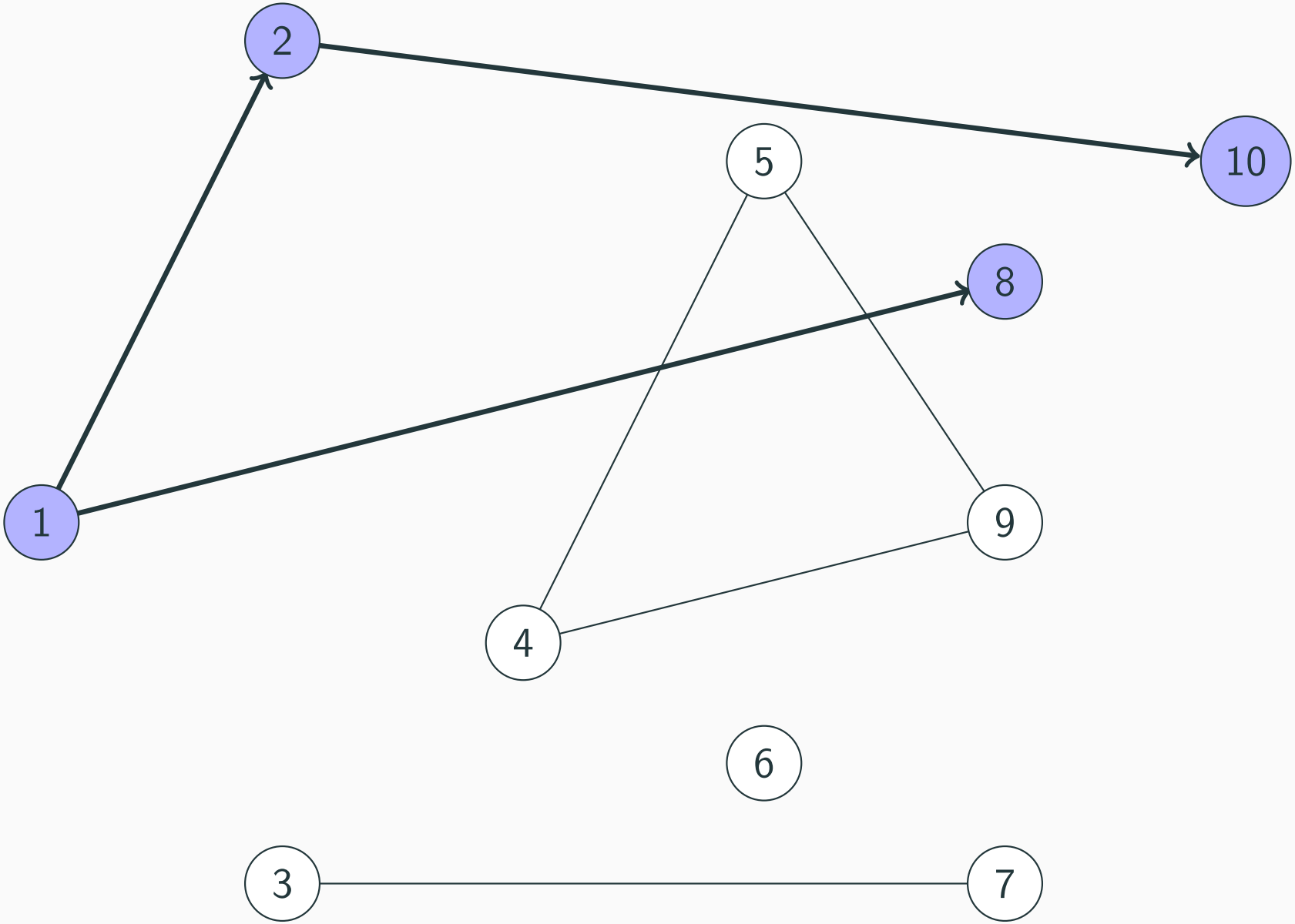
Visualização da identificação dos componentes conectados



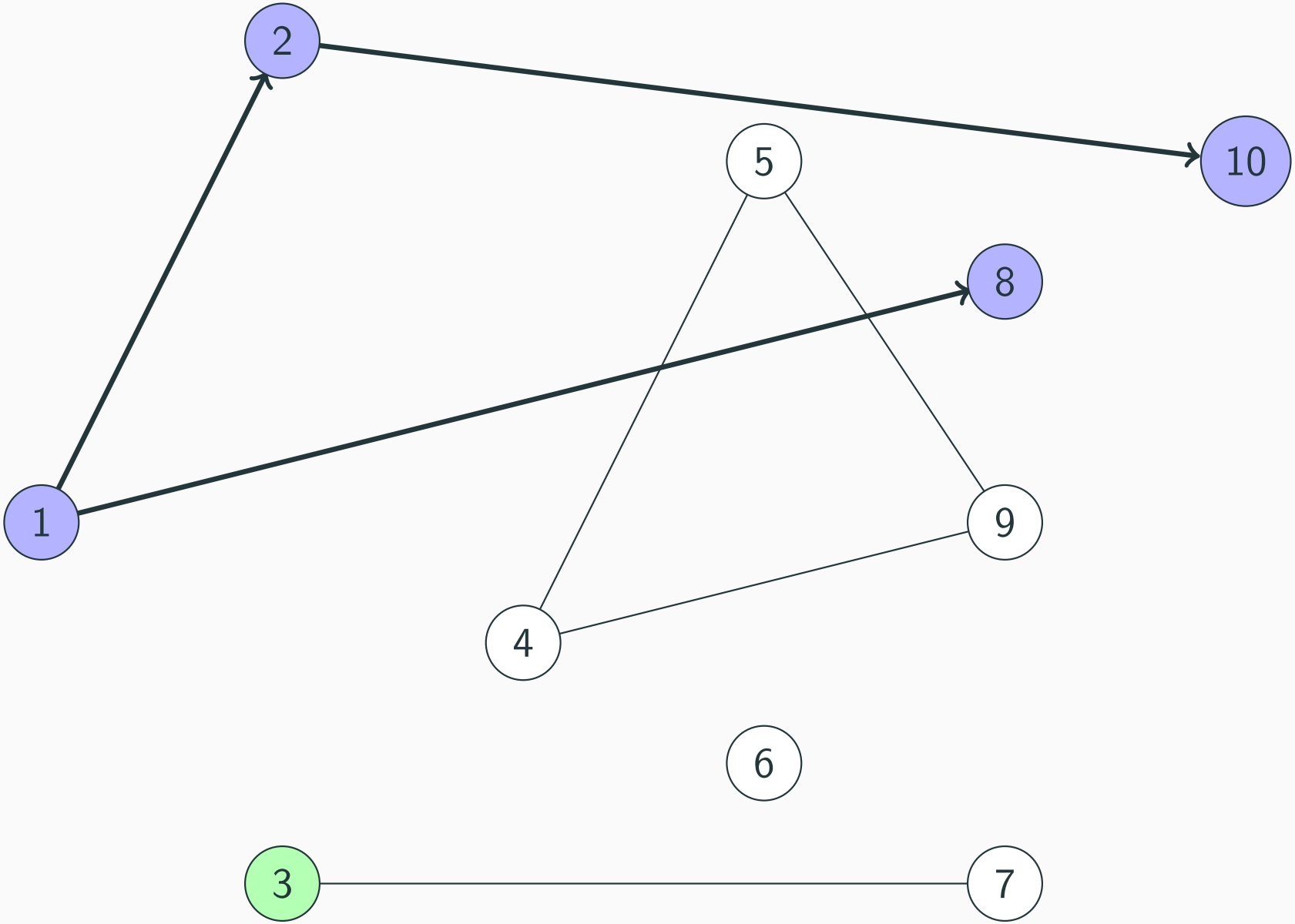
Visualização da identificação dos componentes conectados



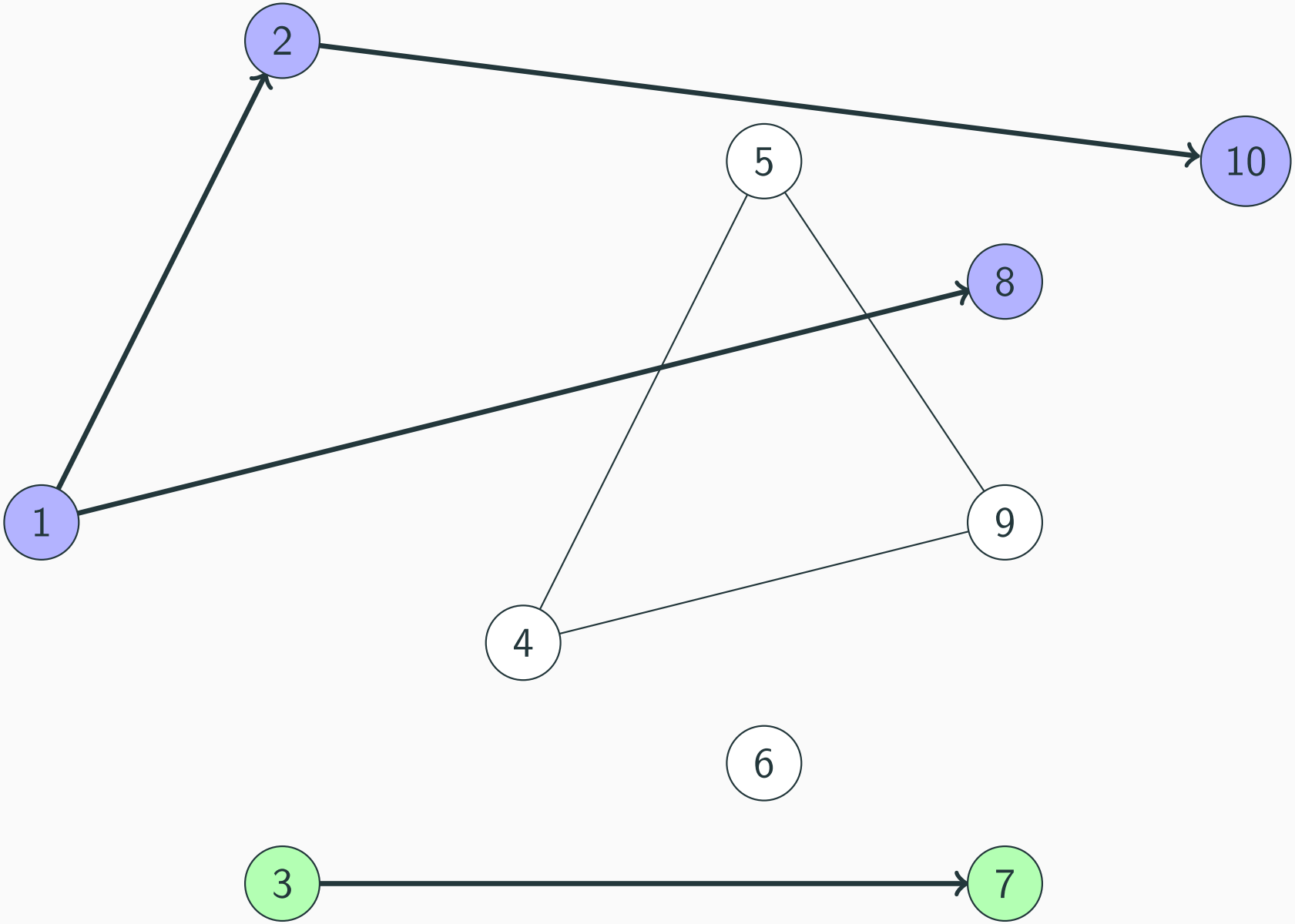
Visualização da identificação dos componentes conectados



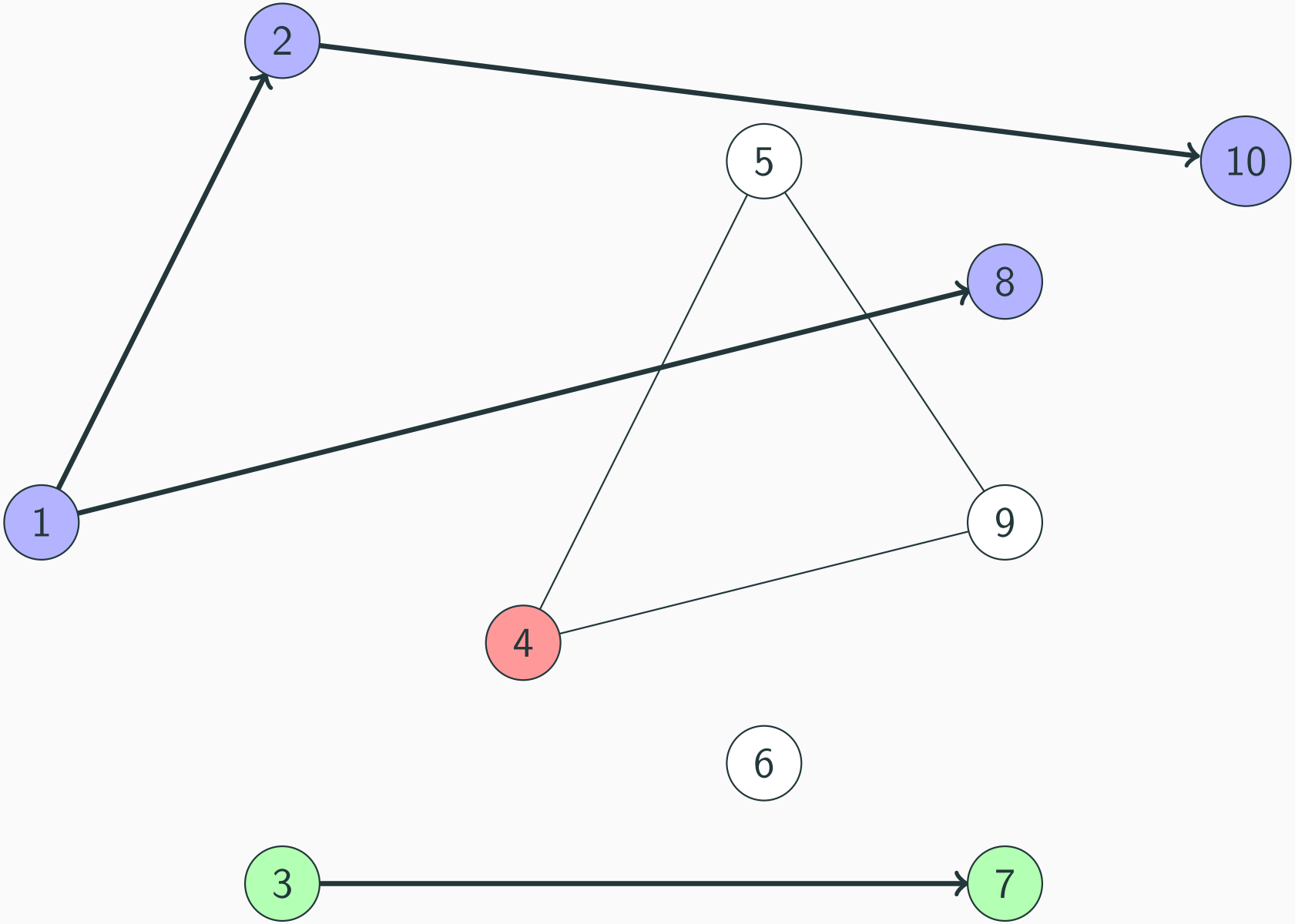
Visualização da identificação dos componentes conectados



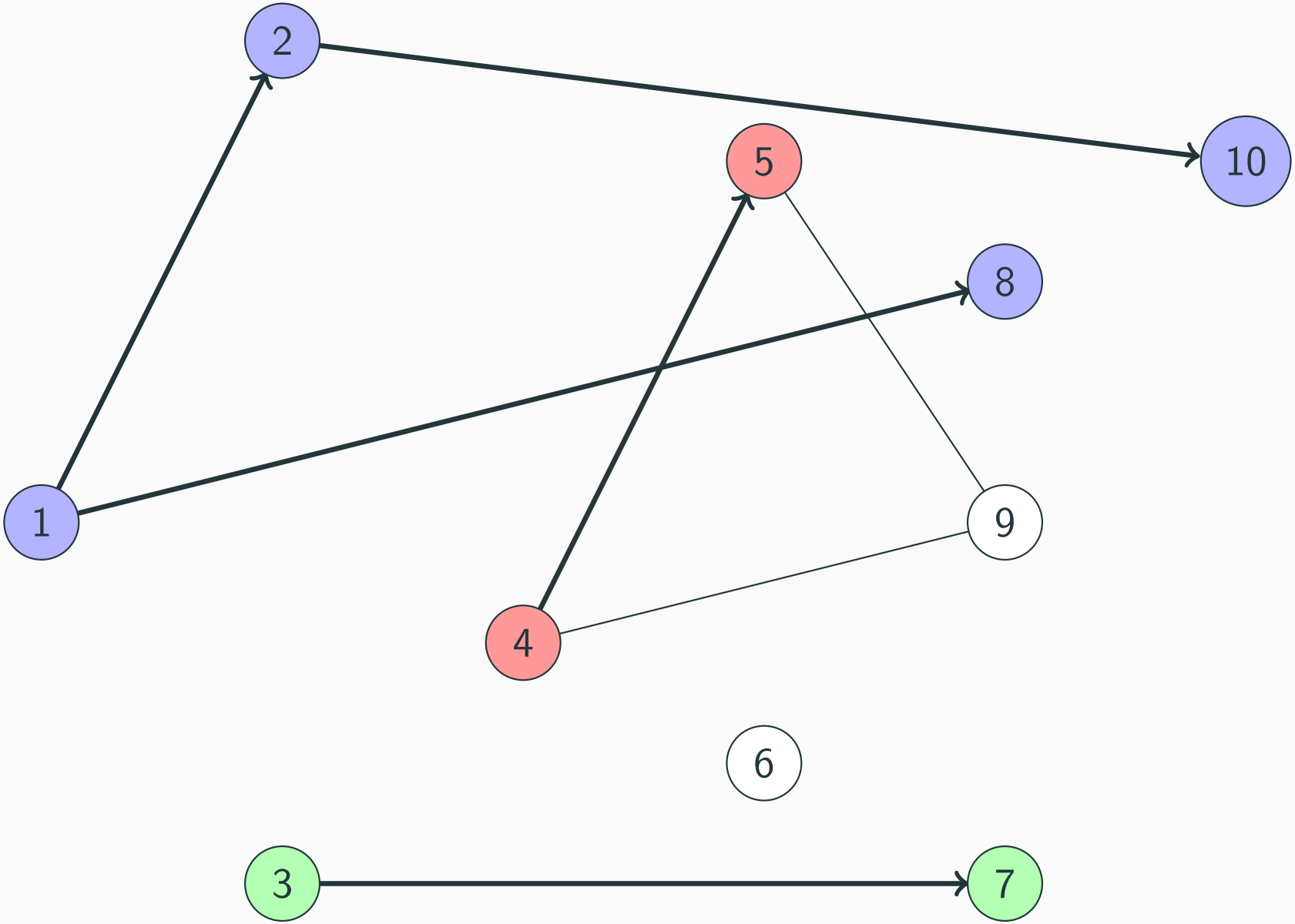
Visualização da identificação dos componentes conectados



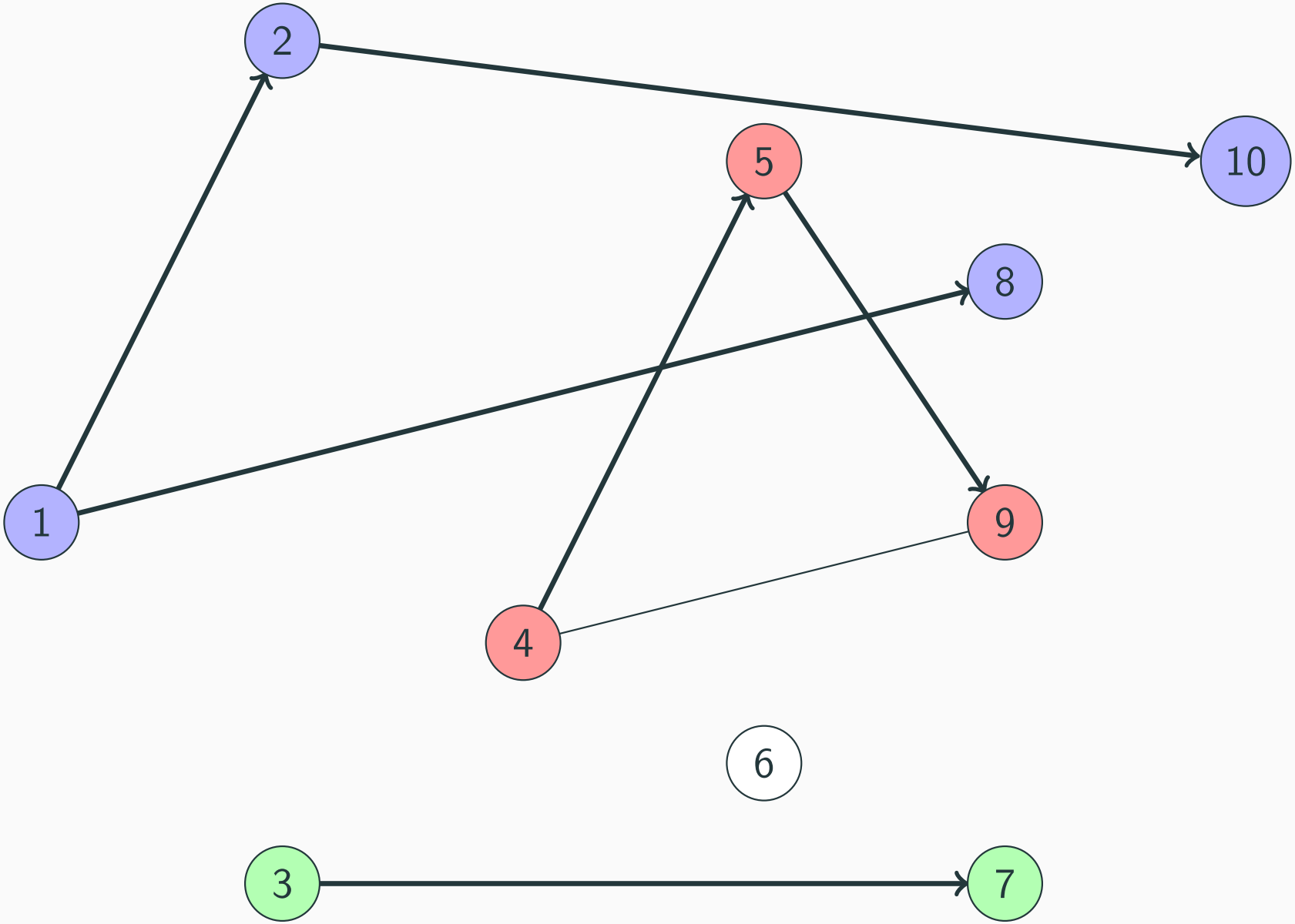
Visualização da identificação dos componentes conectados



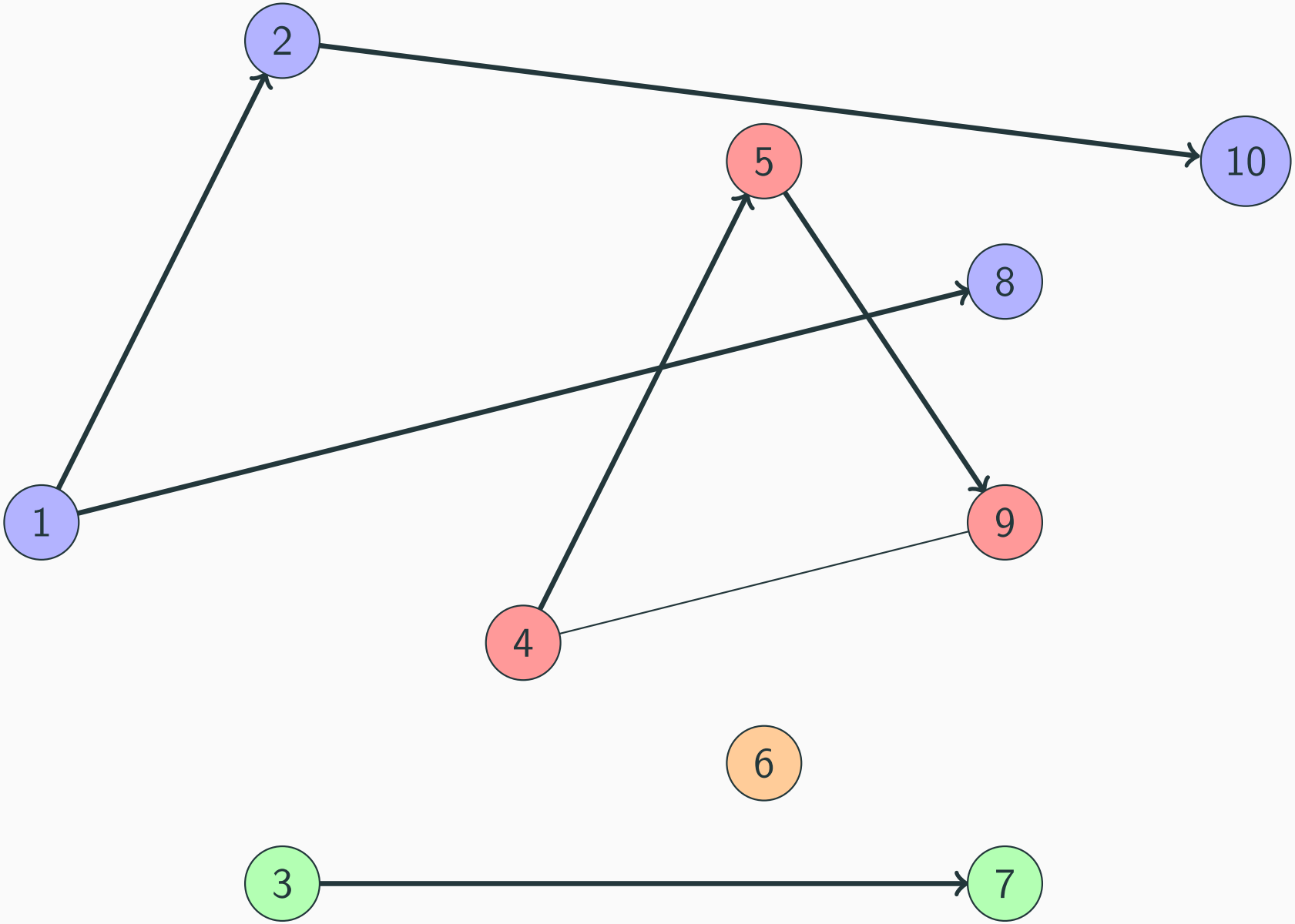
Visualização da identificação dos componentes conectados



Visualização da identificação dos componentes conectados



Visualização da identificação dos componentes conectados



Implementação da identificação dos componentes em C++

```cpp
#include <iostream>
#include <vector>
#include <bitset>

using namespace std;
using ii = pair<int, int>;

const int MAX { 100010 };
bitset<MAX> visited;
vector<int> adj[MAX];

void dfs(int u)
{
    if (visited[u]) return;
    visited[u] = true;

    cout << " " << u;

    for (const auto& v : adj[u])
        dfs(v);
}
```

Implementação da identificação dos componentes em C++

```
22
23 int connected_components(int N)
24 {
25     visited.reset();
26
27     int ans = 0;
28
29     for (int u = 1; u <= N; ++u)
30     {
31         if (not visited[u])
32         {
33             cout << "Component " << ++ans << ":";
34             dfs(u);
35             cout << endl;
36         }
37     }
38
39     return ans;
40 }
41
```

Implementação da identificação dos componentes em C++

```
42 int main()
43 {
44     ii edges[] { ii(1, 2), ii(1, 8), ii(2, 10), ii(3, 7), ii(4, 5),
45         ii(4, 9), ii(5, 9) };
46 
47     for (const auto& [u, v] : edges)
48     {
49         adj[u].push_back(v);
50         adj[v].push_back(u);
51     }
52 
53     connected_components(10);
54 
55     return 0;
56 }
```

Referências

1. **HALIM**, Felix; **HALIM**, Steve. *Competitive Programming 3*, 2010.
2. **LAAKSONEN**, Antti. *Competitive Programmer's Handbook*, 2018.
3. **SKIENA**, Steven S.; **REVILLA**, Miguel A. *Programming Challenges*, 2003.
4. **FILIPEK**, Bartlomiej. *C++17 in Detail*, 2018[1].

[1] https://leanpub.com/cpp17indetail