Gramatyka grafowa do rekurencyjnej adaptacji siatek czworokątnych

Grupa 3 - produkcje P5 i P6

Andrzej Ratajczak i Filip Zybała

15.01.2022

Opis teoretyczny

Krótki opis, w jaki sposób produkcja została zaimplementowana i w jaki sposób przygotowane zostały testy.

Produkcje P5 i P6 zostały zaimplementowane w środowisku przygotowanym przez kolegów z drużyny nr 1

W jaki sposób sprawdzacie Pańswo, czy graf do którego stosujecie produkcje jest izomorficzny z grafem lewej strony produkcji?

Podczas sprawdzania czy produkcję można wykonać mamy szereg testów sprawdzających:

- izomorficzność krawędzi
- izomorficzność etykiet
- izomorficzność współrzędnych
- izomorficzność wierzchołków

Na jakiej podstawie decydujecie Państwo w którym miejscu w grafie będziemy stosować produkcje?

Model nie pozwala na inteligentne znajdowanie izomorficznych grafów, koledzy przygotowali framework w ten sposób, że trzeba wskazać konkretne miejsce gdzie produkcja ma zostać zastosowana. Niemniej jednak sprawdzamy czy taka produkcja ma być prawo zastosowana, jeżeli nie, operacja po prostu się nie wykona.

Jak wyszukujecie Państwo w dużym grafie podgraf izmomorficzny z grafem lewej strony produkcji?

Jak wyżej, należy podać wcześniej zainteresowany przez nas obszar do potencjalnej produkcji.

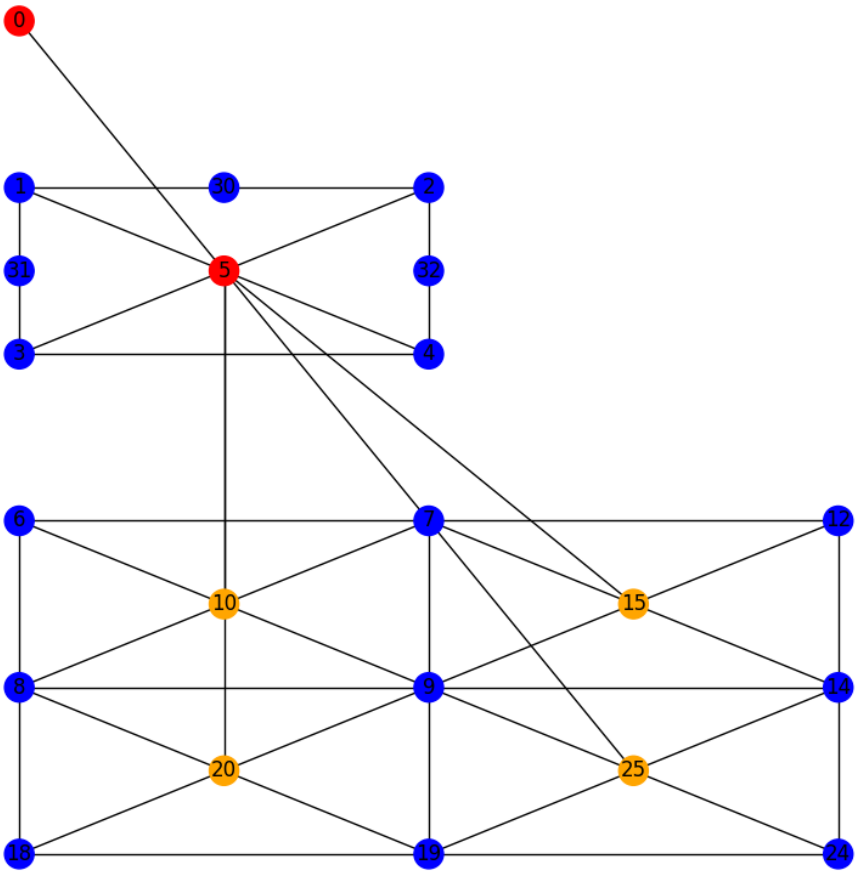
Jak sprawdzacie Państwo, czy w wyniku zastosowania produkcji do danego grafu powstanie "poprawny" graf?

Nie mamy do końca sprawdzenia w kodzie, jednak dodaliśmy ponad 30 testów jednostkowych (po 15 do każdej produkcji), które między innymi sprawdzają poprawność lewej i prawej strony produkcji grafu.

Przeprowadzone testy i ich wyniki (np. zrzutki z ekranu), wraz z grafami do których zostały zastosowane testy

Poniżej znajduje się odwołanie do każdego punktu za jakie miały być oceniane produkcje jak i ich testy. Produkcje mają dopisane jaki nr testu odpowiada za sprawdzenie danej własności.

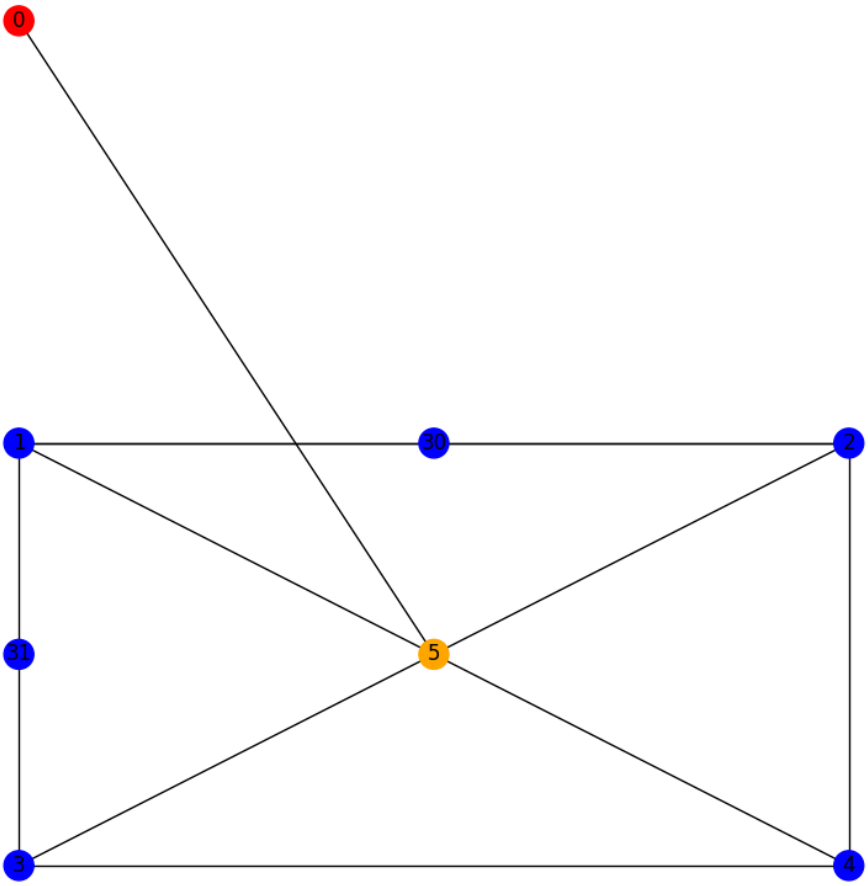
P5



test1

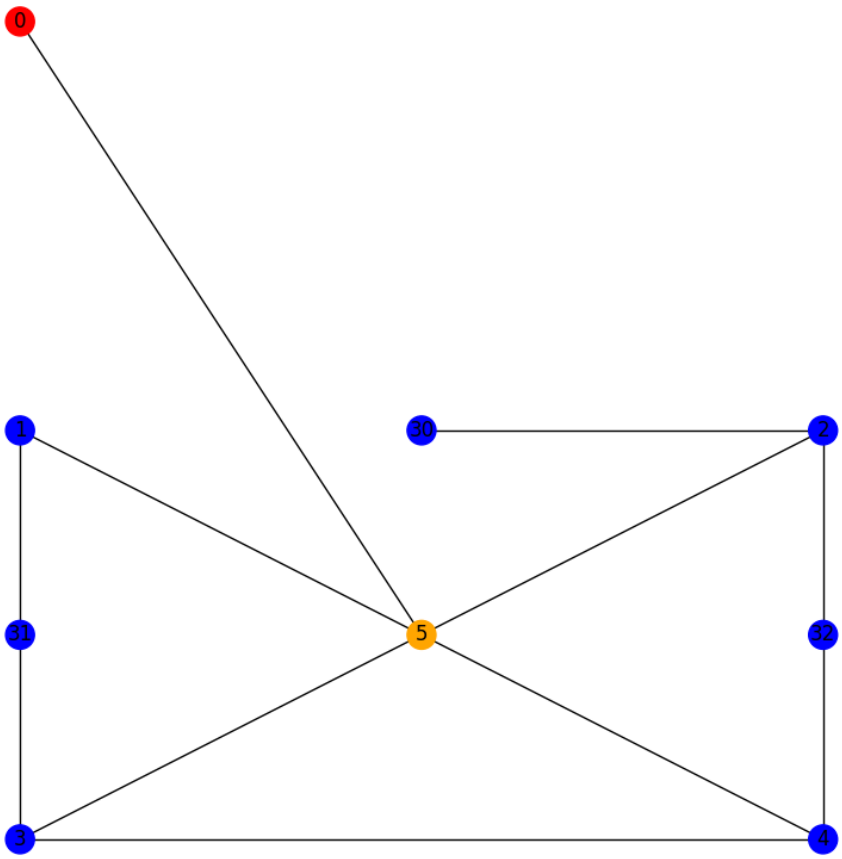
1. Czy produkcja sprawdza czy graf (podgraf grafu) do którego chcemy zastosować produkcję jest izomorficzny z grafem lewej strony produkcji (czy da się ją wykonać)?

a) czy zmiana grafu do którego stosujemy produkcję poprzez usunięcie losowego wierzchołka nie psuje tego mechanizmu



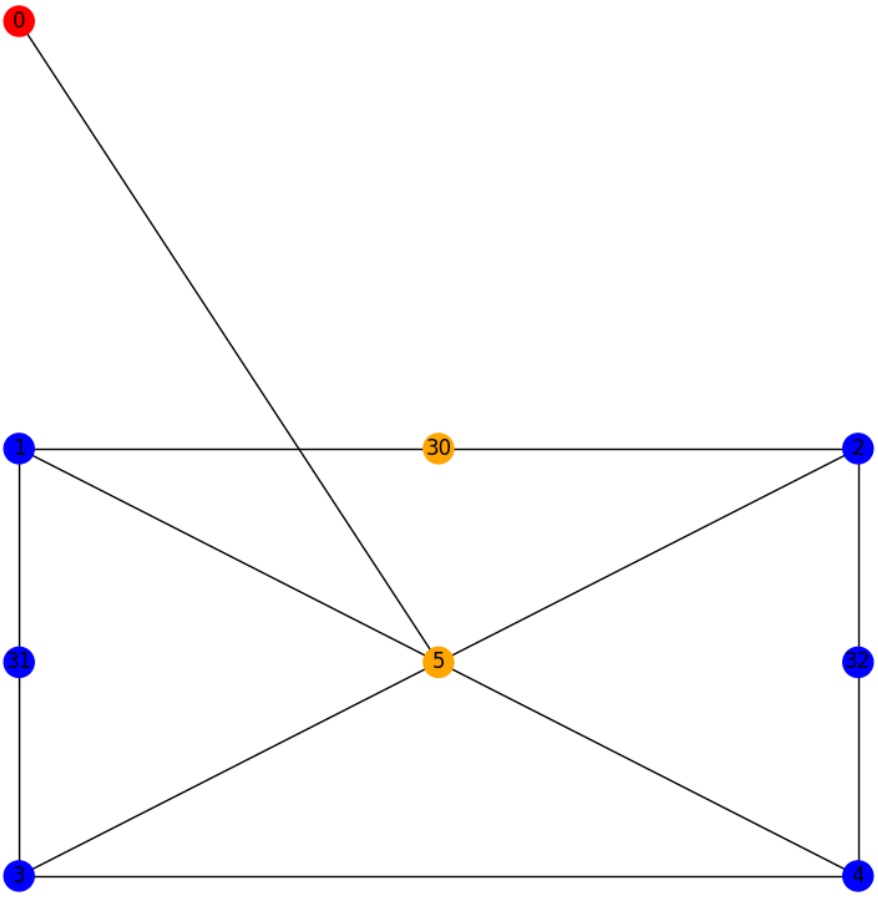
test2

b) czy zmiana grafu do którego stosujemy produkcję poprzez usunięcie losowej krawędzi nie psuje tego mechanizmu



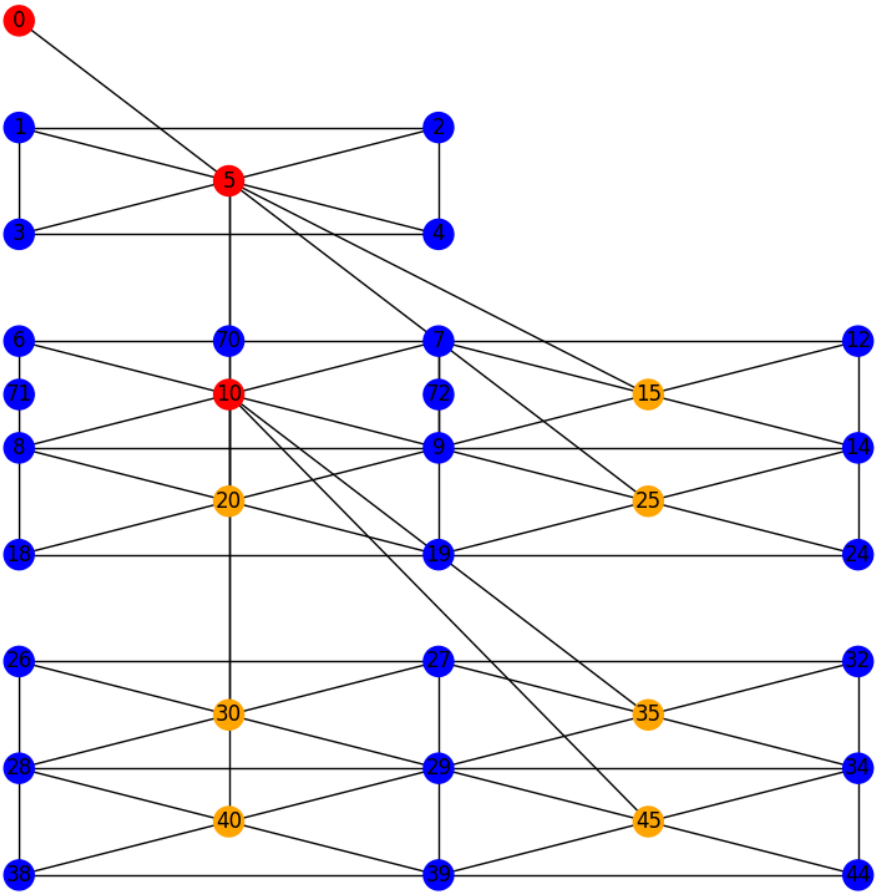
test3

c) czy zmiana grafu do którego stosujemy produkcję poprzez zmianę etykiety losowego wierzchołka nie psuje tego mechanizmu



test4

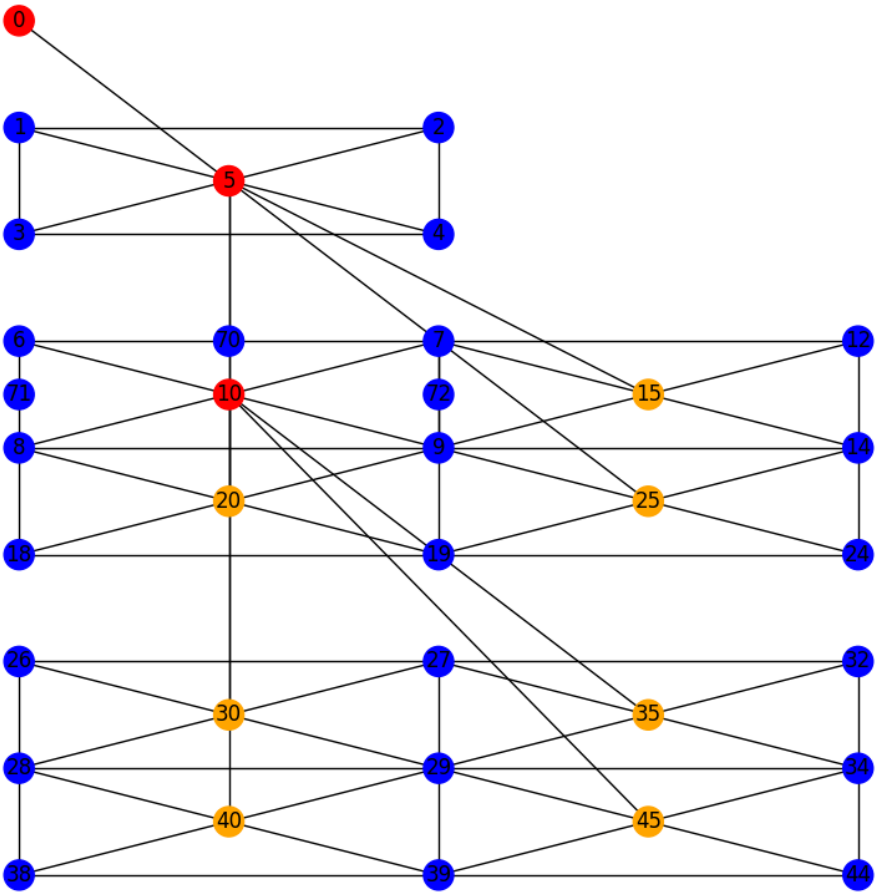
d) czy umieszczenie grafu izomorficznego z grafem lewej strony jako podgrafu większego grafu nie psuje tego mechanizmu



test5

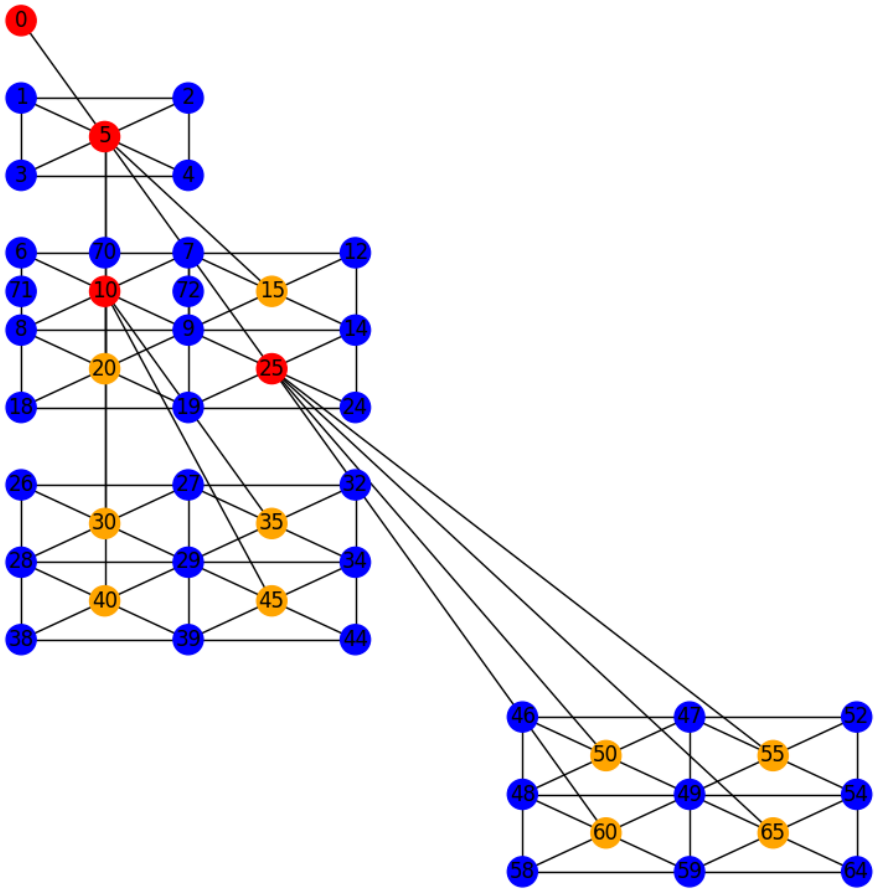
2. Czy produkcja dobrze się wykonała?

a) czy jeśli graf izomorficzny z grafem lewej strony jest umieszczony jako podgraf większego grafu, to czy produkcja nie „uszkadza" większego grafu



test5

b) czy jeśli graf izomorficzny z grafem lewej strony jest umieszczony w jako podgraf większego grafu, to czy produkcja dobrze transformuje osadzenie



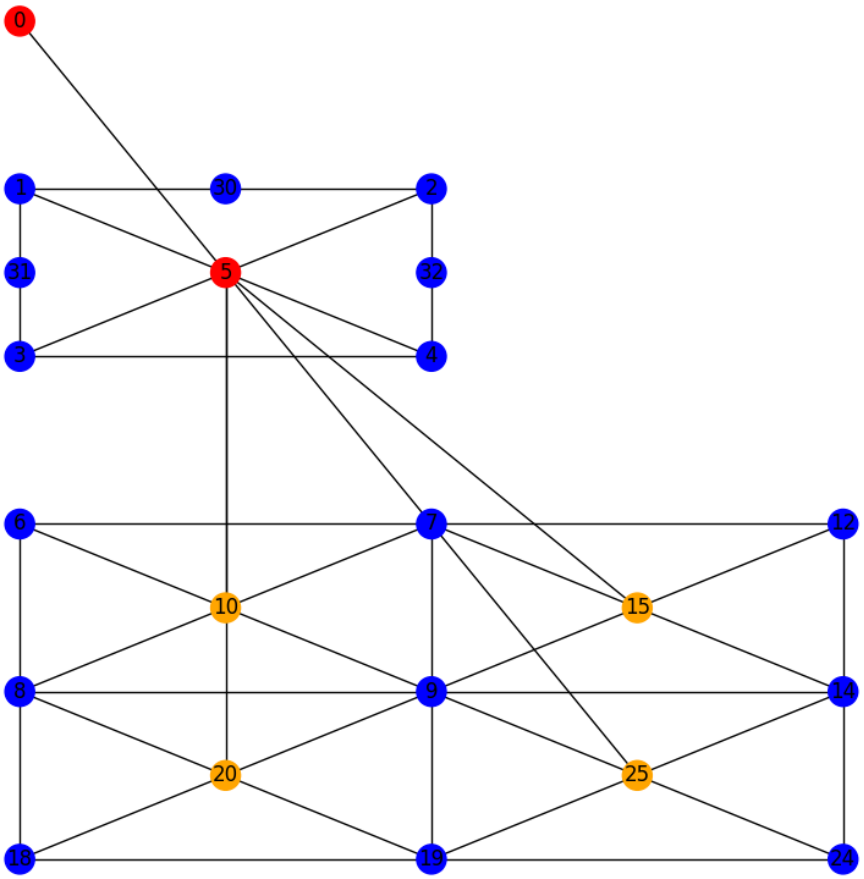
test6

c) czy graf izomorficzny z grafem prawej strony jest poprawny (czy ma wszystkie wierzchołki, krawędzie i poprawne etykiety)

Z powodu braku połączeń pomiędzy małymi kwadratami połaczenie musi wystarczyć jako istnienie wierzchołków na pozycji x, y i połączeń wewnątrz mniejszych kwadratów

```
for x in [0, 1, 2]:
    for y in [-3, -4, -5]:
        vertex = find_vertice_with_coordinates_and_remove_duplicates(x, y, lower_squares)
        assert vertex.label == VertexLabel.E
for x in [0.5, 1.5]:
    for y in [-3.5, -4.5]:
        vertex = find_vertice_with_coordinates_and_remove_duplicates(x, y, lower_squares)
        assert vertex.label == VertexLabel.I
```

```
assert lower_squares[0].edges == [(9, 8), (8, 6), (6, 7), (7, 9), (9, 10), (8, 10), (6, 10),
assert lower_squares[1].edges == [(9, 7), (7, 12), (12, 14), (14, 9), (9, 15), (7, 15), (12,
assert lower_squares[2].edges == [(18, 8), (8, 9), (9, 19), (19, 18), (18, 20), (8, 20), (19
assert lower_squares[3].edges == [(9, 14), (14, 24), (24, 19), (19, 9), (9, 25), (14, 25),
```



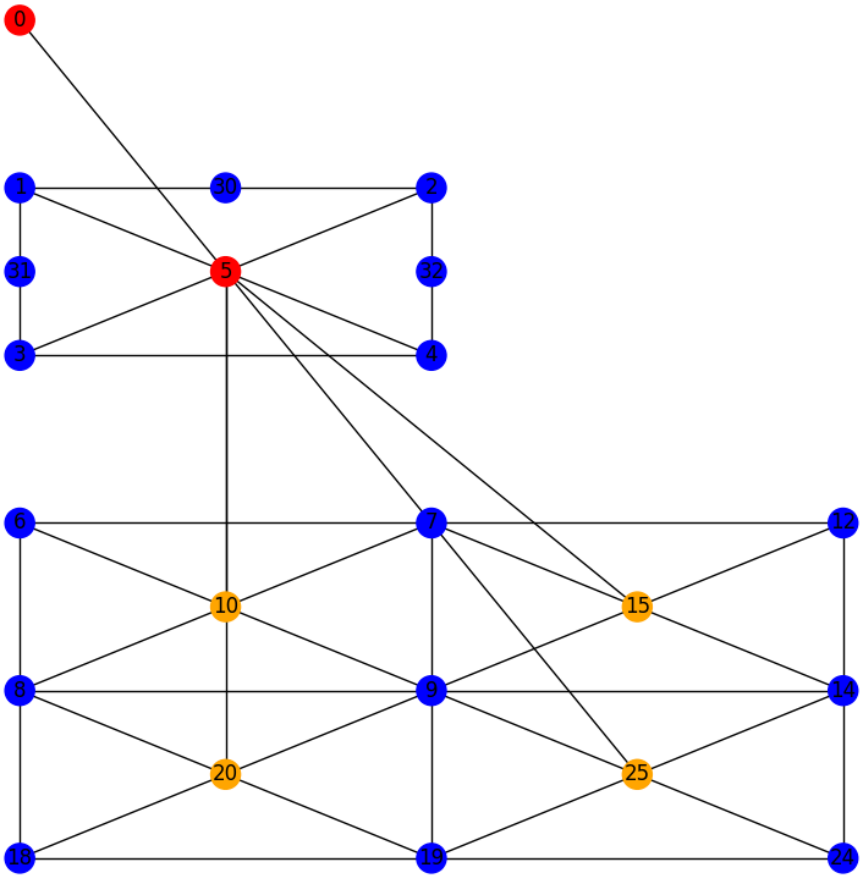
test7

d) czy współrzędne nowych wierzchołków w tym grafie są poprawne

Wierzchołki da się wyliczyć za pomocą wzoru:

- kolumna: od `0` do `2 ^ poziom`
- rząd: od `-sum for i in range(0, poziom): 2^poziom` do `-(sum for i in range(0, poziom) 2^poziom) - (2 ^ poziom)` (intuicja: suma wszystkich coraz większych kwadratów)

```
for s in lower_squares:
    assert s.layer_number == lower_layer_number
for x in [0, 1, 2]:
    for y in [-3, -4, -5]:
        vertex = find_vertice_with_coordinates_and_remove_duplicates(x, y, lower_squares)
        assert vertex.label == VertexLabel.E
for x in [0.5, 1.5]:
    for y in [-3.5, -4.5]:
        vertex = find_vertice_with_coordinates_and_remove_duplicates(x, y, lower_squares)
        assert vertex.label == VertexLabel.I
```

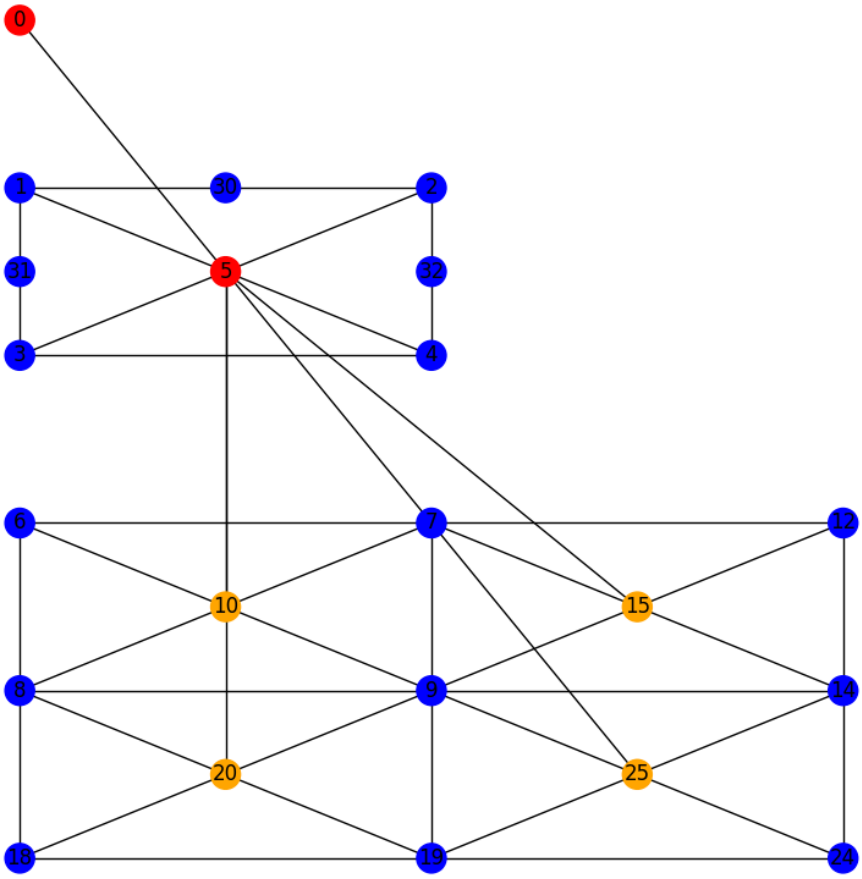


test7

e) czy nowy graf umieszczony jest na poprawnym poziomie

```
upper_layer_number = graph_fragment.layer_number
lower_layer_number = upper_layer_number + 1
```

```
lower_squares = [verticies_graph_fragment.get(x) for x in [10, 15, 20, 25]]
for s in lower_squares:
    assert s.layer_number == lower_layer_number
```



test7

3. Czy graf dobrze się rysuje?

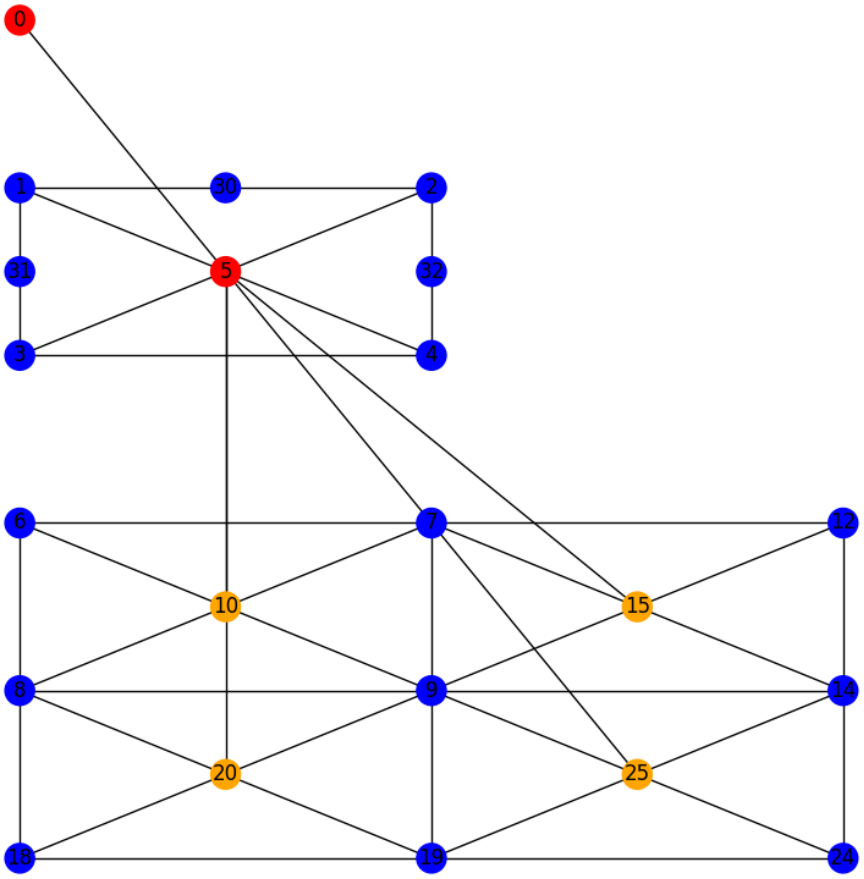
a) czy są wszystkie wierzchołki i krawędzie

Kopia 2. c)

```
for x in [0, 1, 2]:
    for y in [-3, -4, -5]:
        vertex = find_vertice_with_coordinates_and_remove_duplicates(x, y, lower_squares)
        assert vertex.label == VertexLabel.E
for x in [0.5, 1.5]:
```

```
    for y in [-3.5, -4.5]:
        vertex = find_vertice_with_coordinates_and_remove_duplicates(x, y, lower_squares)
        assert vertex.label == VertexLabel.I

assert lower_squares[0].edges == [(9, 8), (8, 6), (6, 7), (7, 9), (9, 10), (8, 10), (6, 10),
assert lower_squares[1].edges == [(9, 7), (7, 12), (12, 14), (14, 9), (9, 15), (7, 15), (12,
assert lower_squares[2].edges == [(18, 8), (8, 9), (9, 19), (19, 18), (18, 20), (8, 20), (19
assert lower_squares[3].edges == [(9, 14), (14, 24), (24, 19), (19, 9), (9, 25), (14, 25),
```



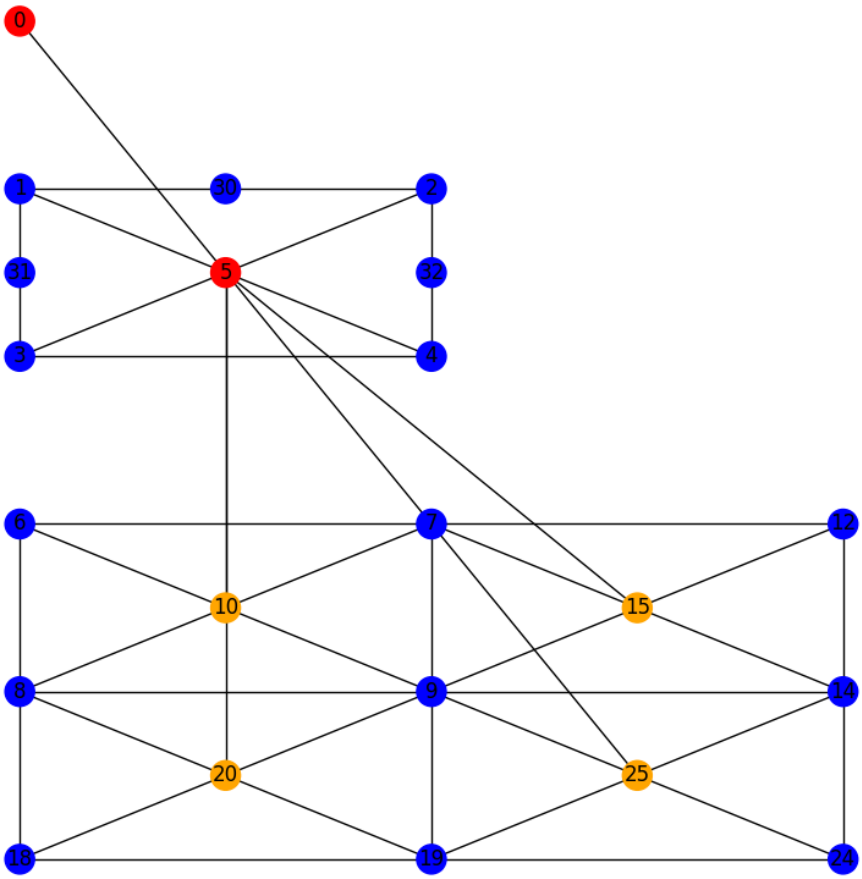
test7

b) czy wierzchołki są narysowane w poprawnych współrzędnych

Kopia 2. d)

```
for s in lower_squares:
    assert s.layer_number == lower_layer_number
```

```
for x in [0, 1, 2]:
    for y in [-3, -4, -5]:
        vertex = find_vertice_with_coordinates_and_remove_duplicates(x, y, lower_squares)
        assert vertex.label == VertexLabel.E
for x in [0.5, 1.5]:
    for y in [-3.5, -4.5]:
        vertex = find_vertice_with_coordinates_and_remove_duplicates(x, y, lower_squares)
        assert vertex.label == VertexLabel.I
```

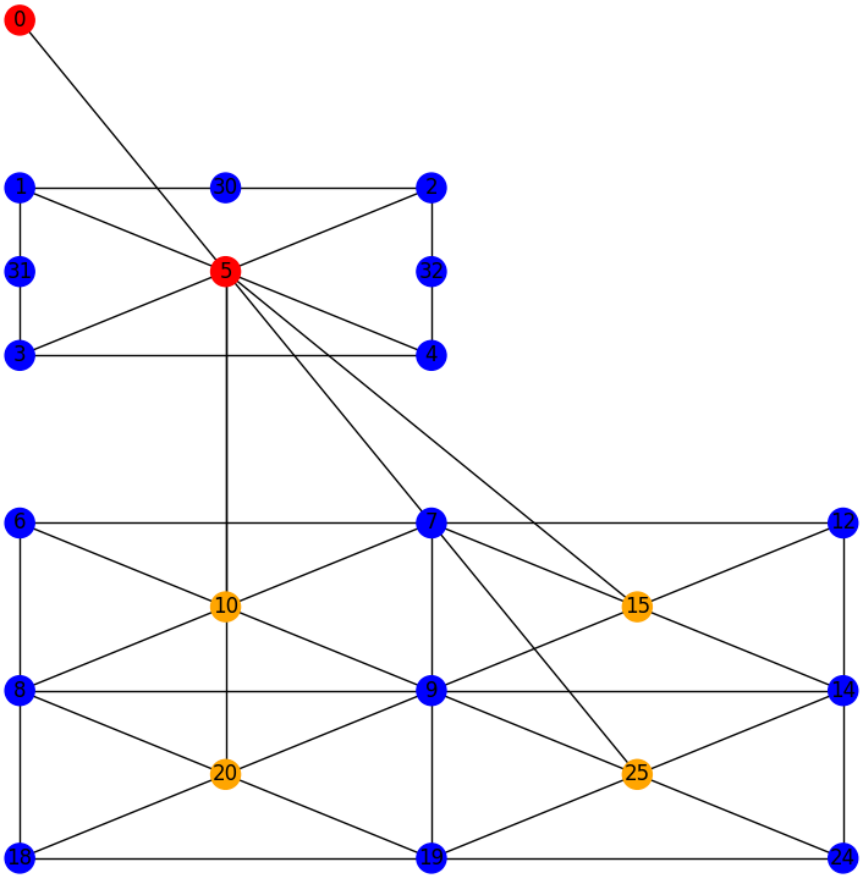


test7

c) czy da się wybierać poziom grafu do narysowania

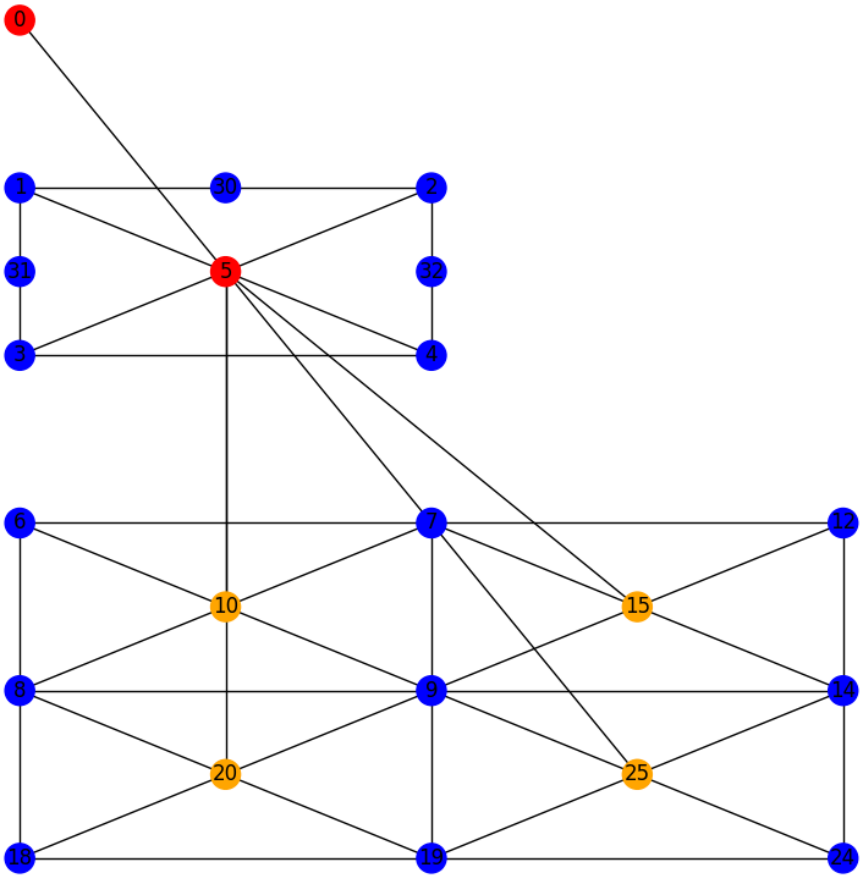
Niestety model tego nie oferuje

d) czy są narysowane etykiety wierzchołków



test7

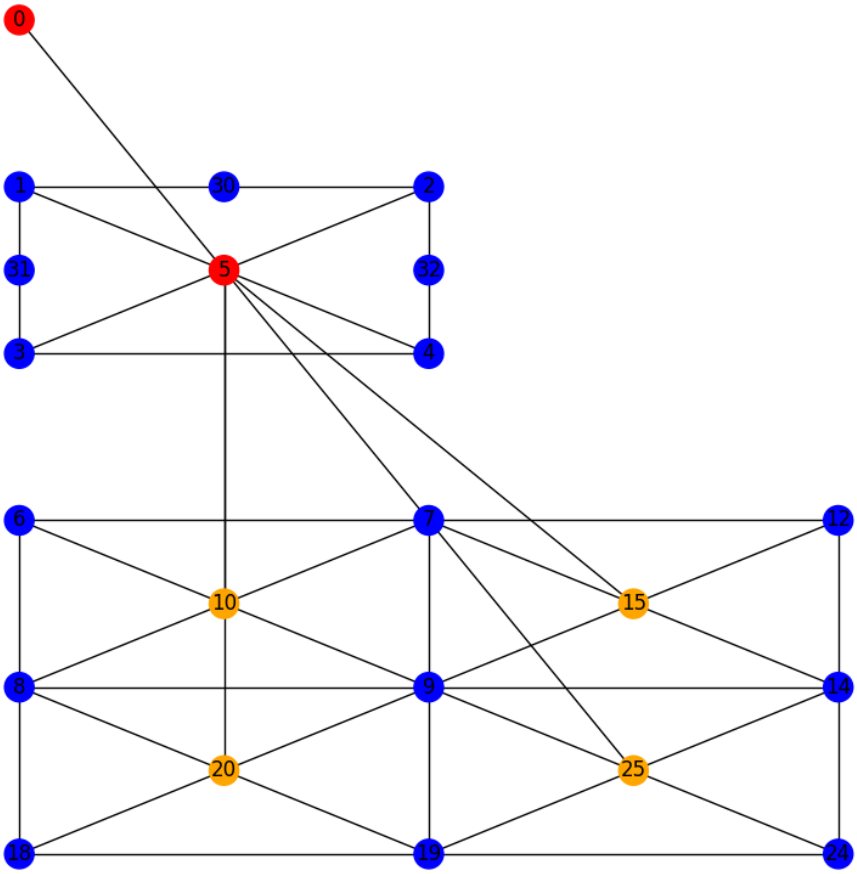
e) czy jest zaznaczone które wierzchołki mają linki do poprzedniego lub następnego poziomu



test7

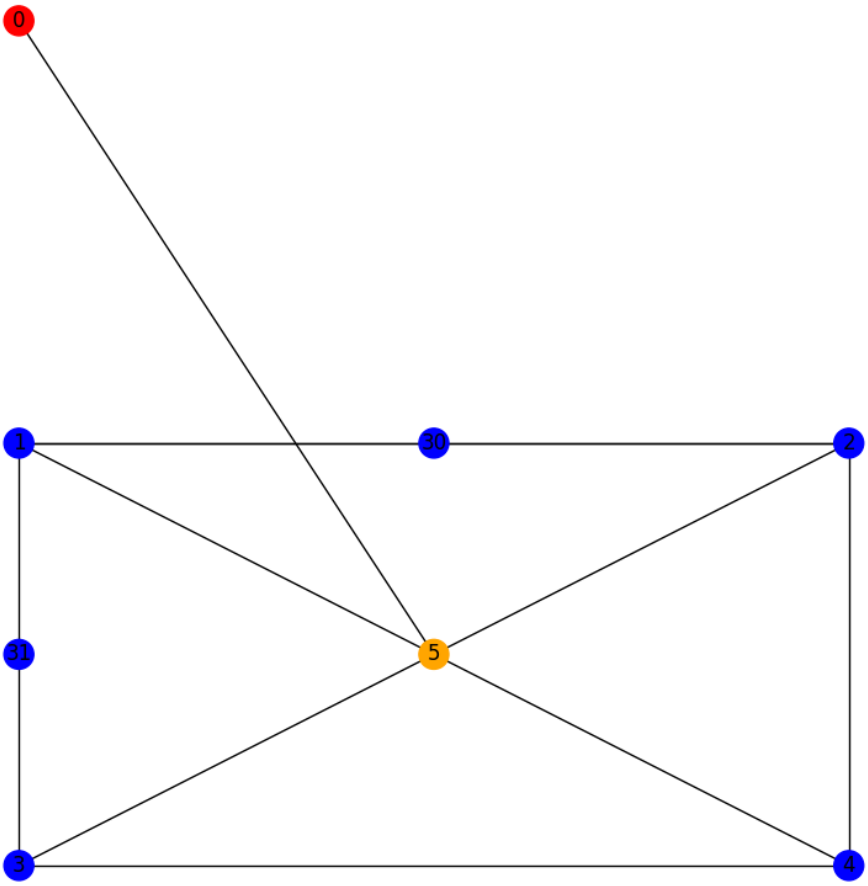
4. Czy zostały przygotowanie różne grafy do testowania

a) czy został przygotowany graf izomorficzny z grafem lewej strony produkcji, który jest poprawny



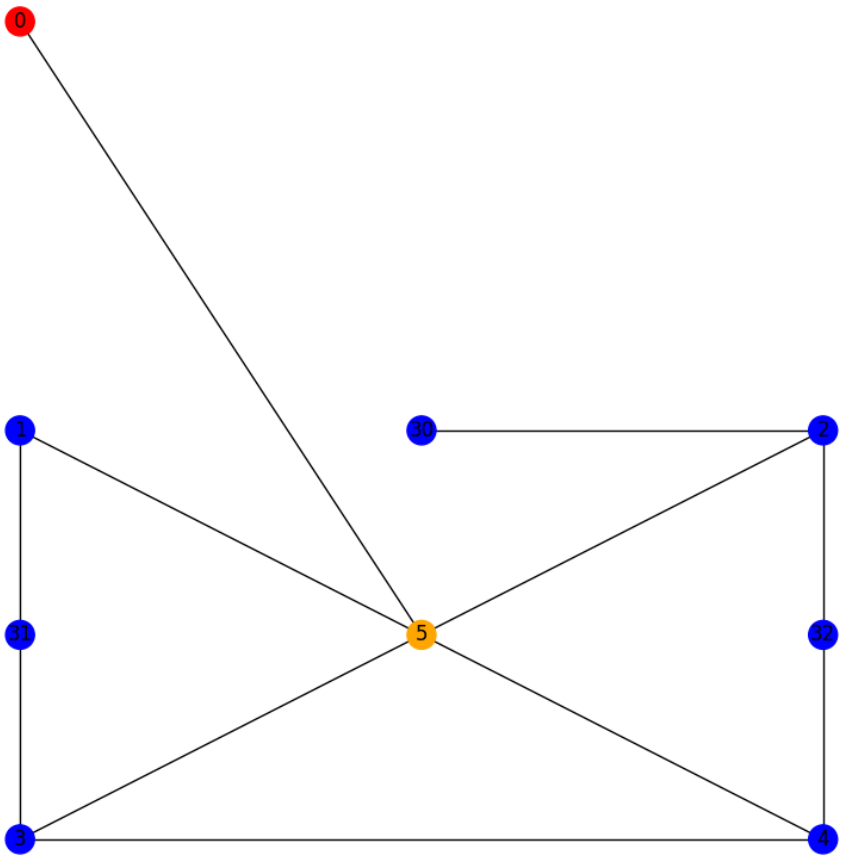
test1

b) czy został przygotowany graf izomorficzny z grafem lewej strony produkcji, który jest niepoprawny (bez jakiegoś wierzchołka)



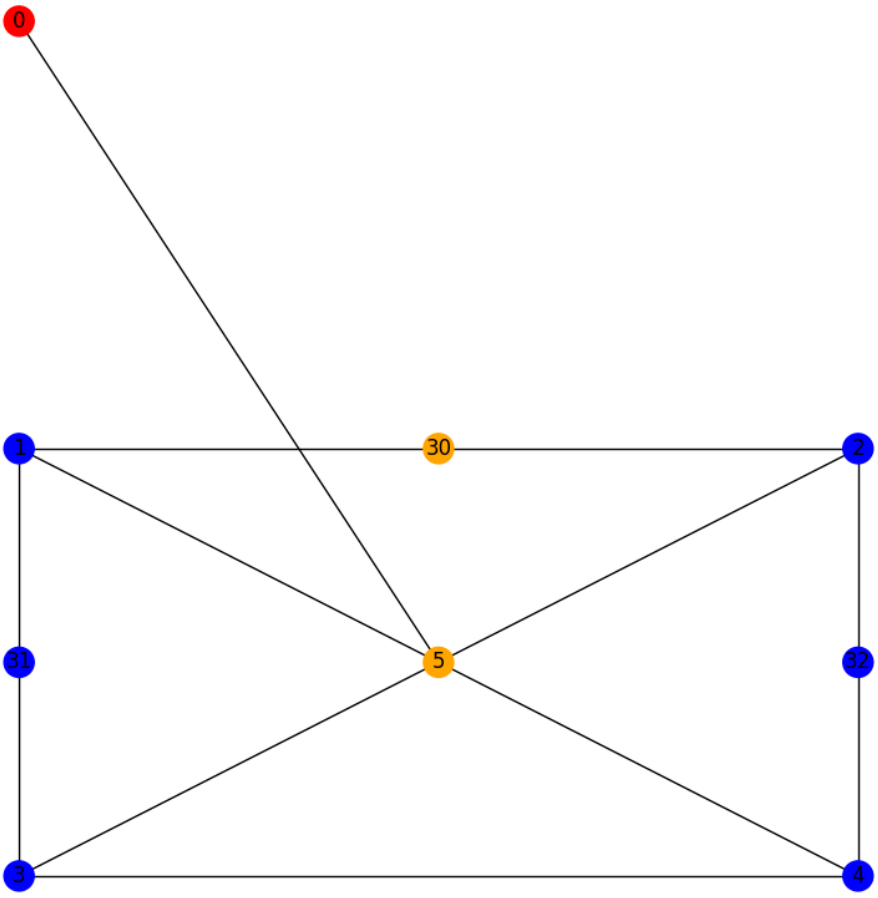
test2

c) czy został przygotowany graf izomorficzny z grafem lewej strony produkcji, który jest niepoprawny (bez jakiejś krawędzi)



test3

d) czy został przygotowany graf izomorficzny z grafem lewej strony produkcji, który jest niepoprawny (z niepoprawną etykietą)

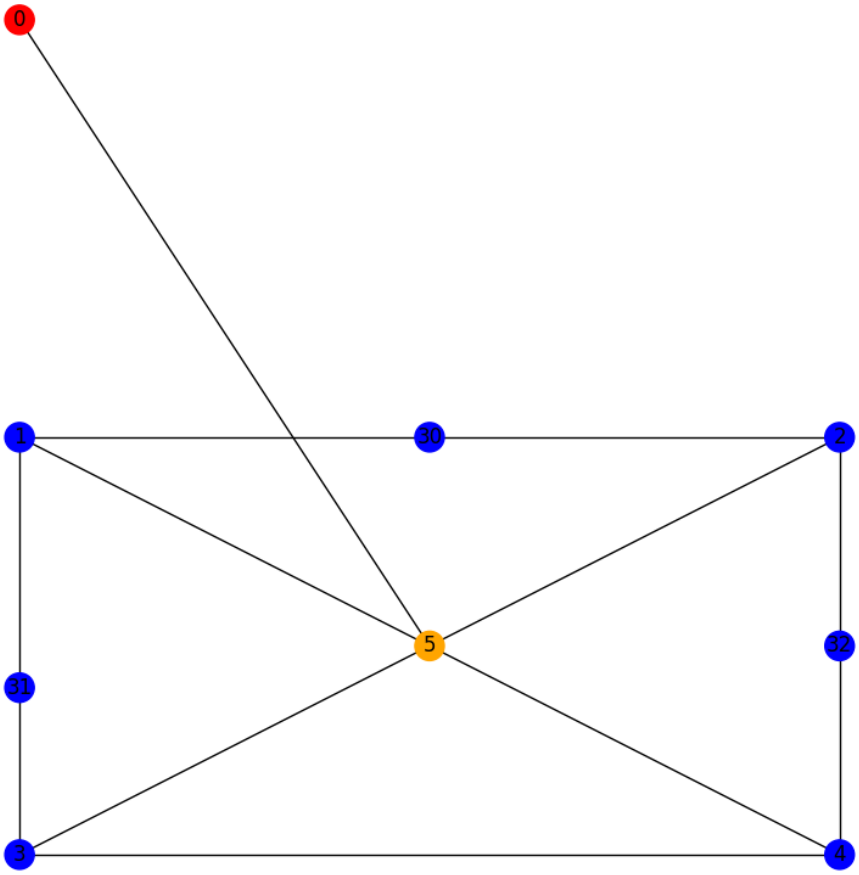


test4

e) czy został przygotowany graf izomorficzny z grafem lewej strony produkcji, który jest niepoprawny (z błędnymi współrzędnymi)

Współrzeðne wirzechołka `31` to `(0, -1.6)` zamiast `(0, -1.5)` przez co produkcja nie zostaje zastosowana.

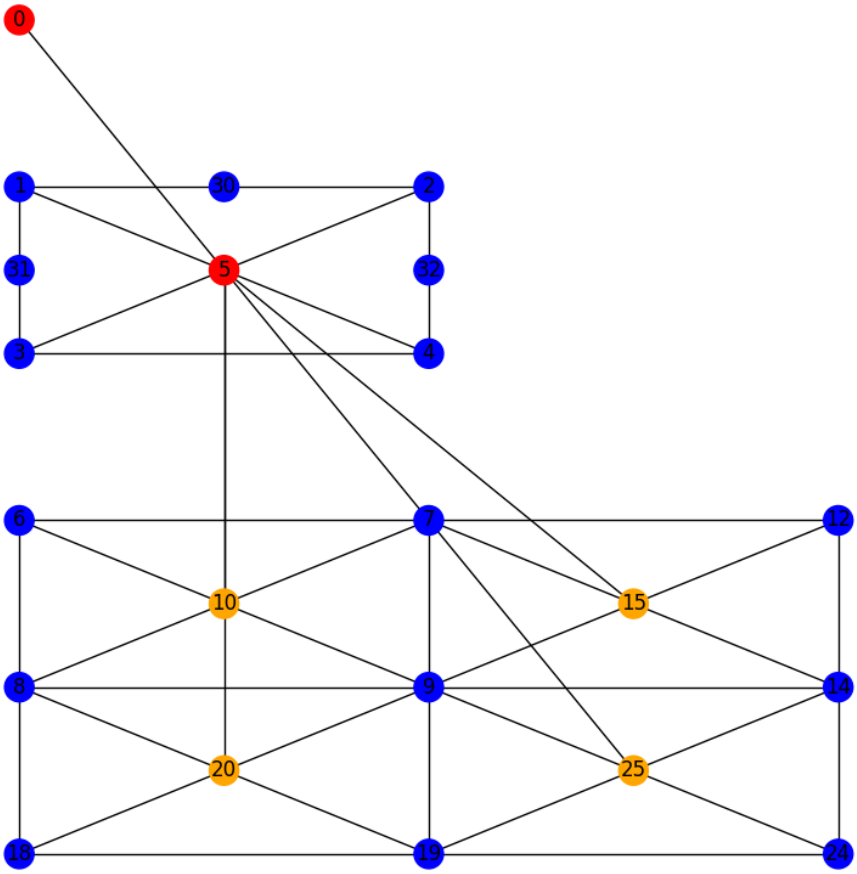
```
graph_fragment.vertices.extend(list([Vertex(0.5, -1, 30, VertexLabel(1)), Vertex(0, -1.6, 3
```



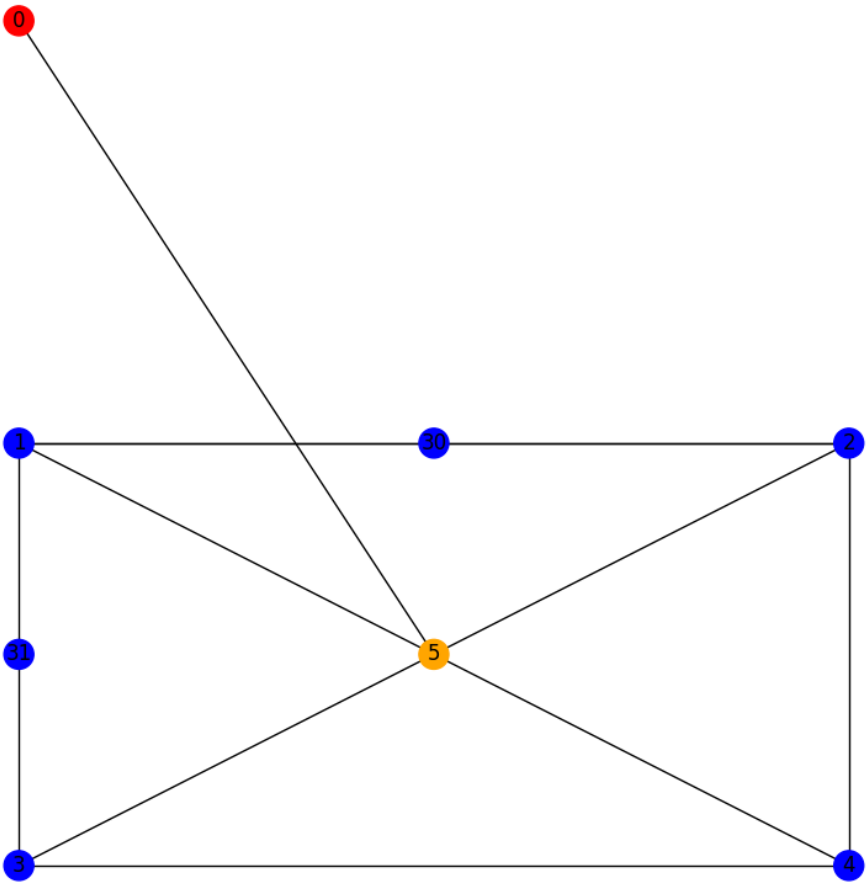
test8

5. Czy wynik produkcji został dobrze sprawdzony

a) czy zostało sprawdzone czy produkcja wykonała się na poprawnym grafie i nie została wykonana na niepoprawnym grafie?

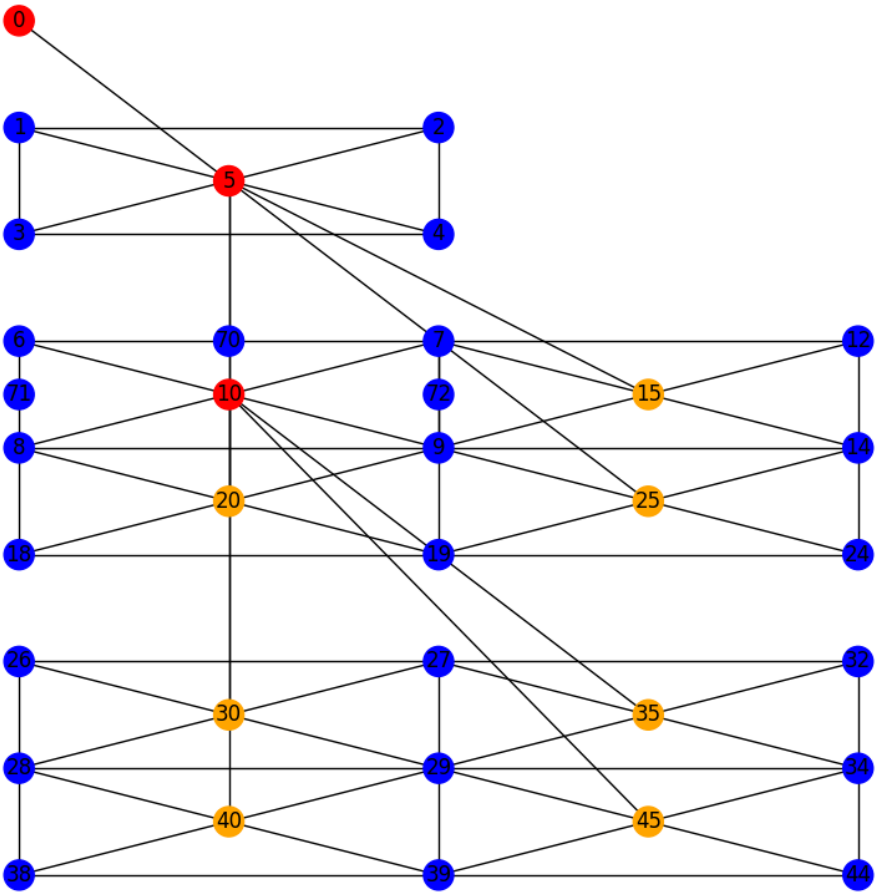


test1



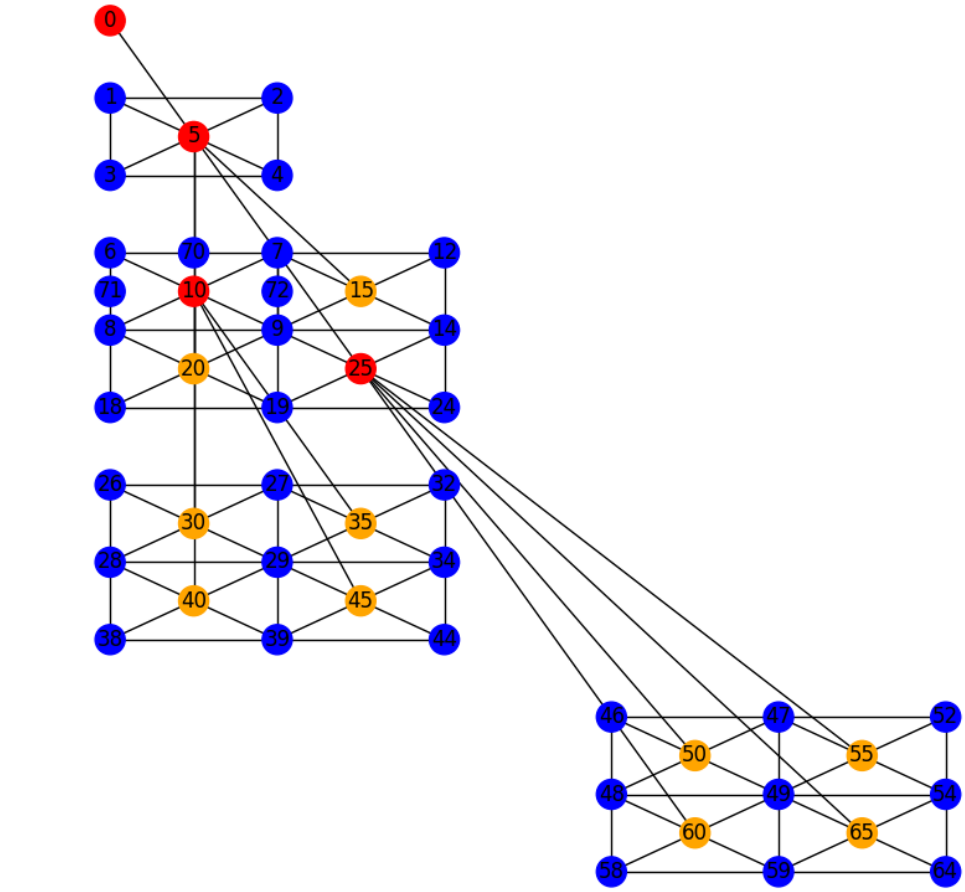
test2

b) czy zostało sprawdzone czy jeśli graf lewej strony jest umieszczony w jako podgraf większego grafu, to czy produkcja nie uszkadza większego grafu



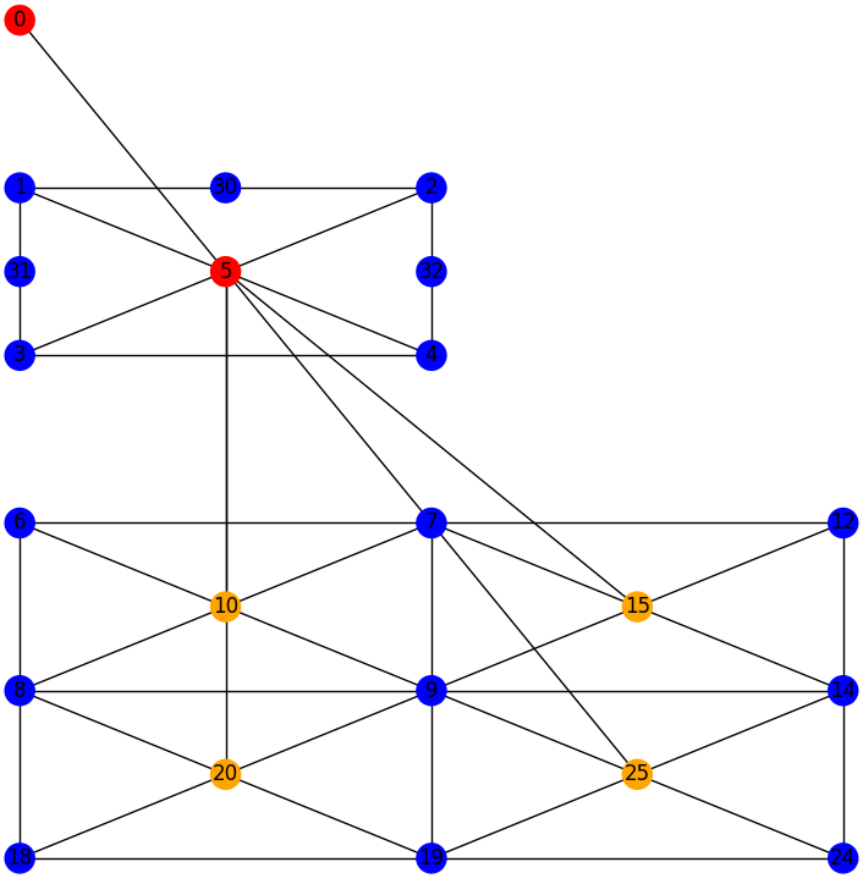
test5

c) czy zostało sprawdzone czy jeśli graf lewej strony jest umieszczony jako podgraf większego grafu, to czy produkcja dobrze transformuje osadzenie



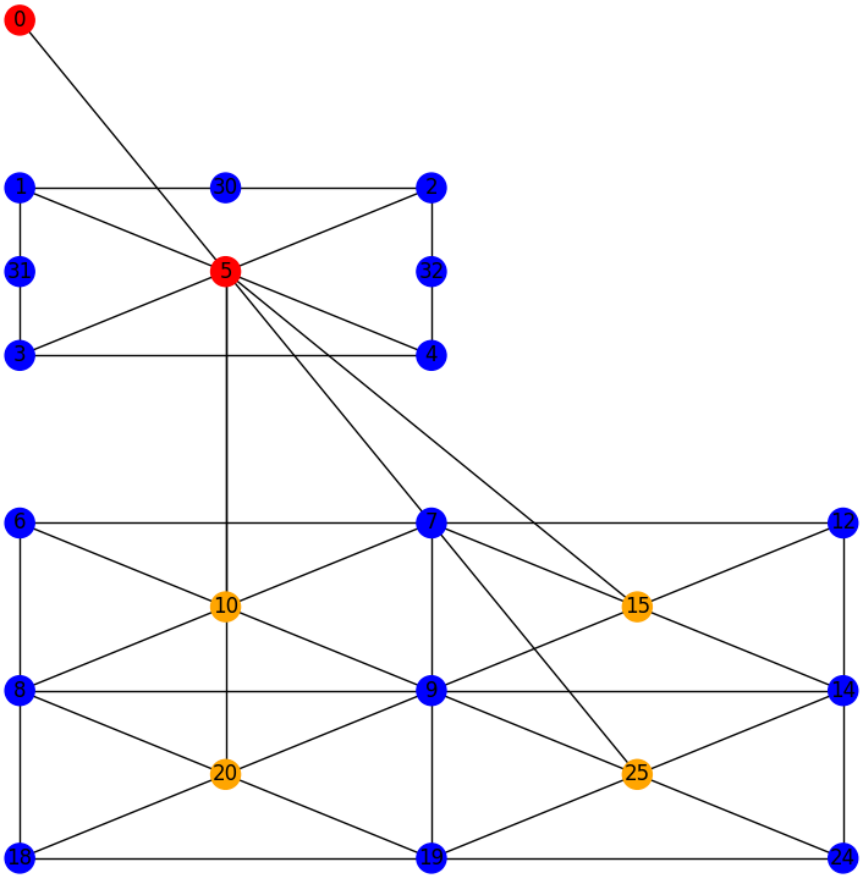
test6

d) czy zostało sprawdzone czy graf prawej strony jest poprawny (czy ma wszystkie wierzchołki, krawędzie i poprawne etykiety)



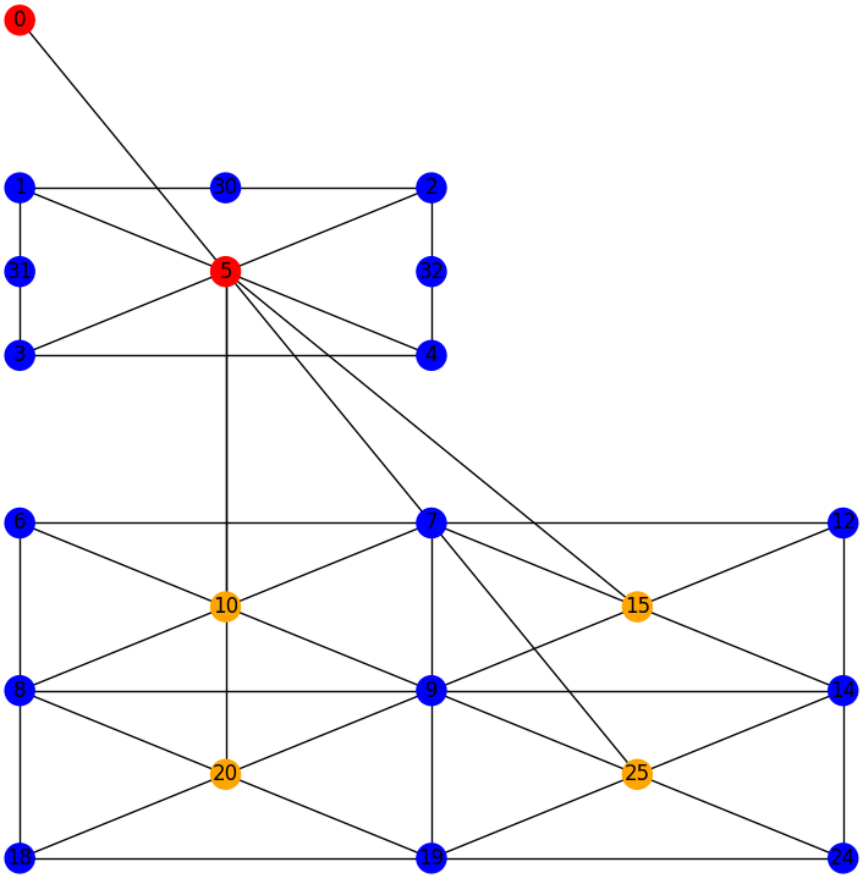
test7

e) czy zostało sprawdzone czy współrzędne nowych wierzchołków są poprawne



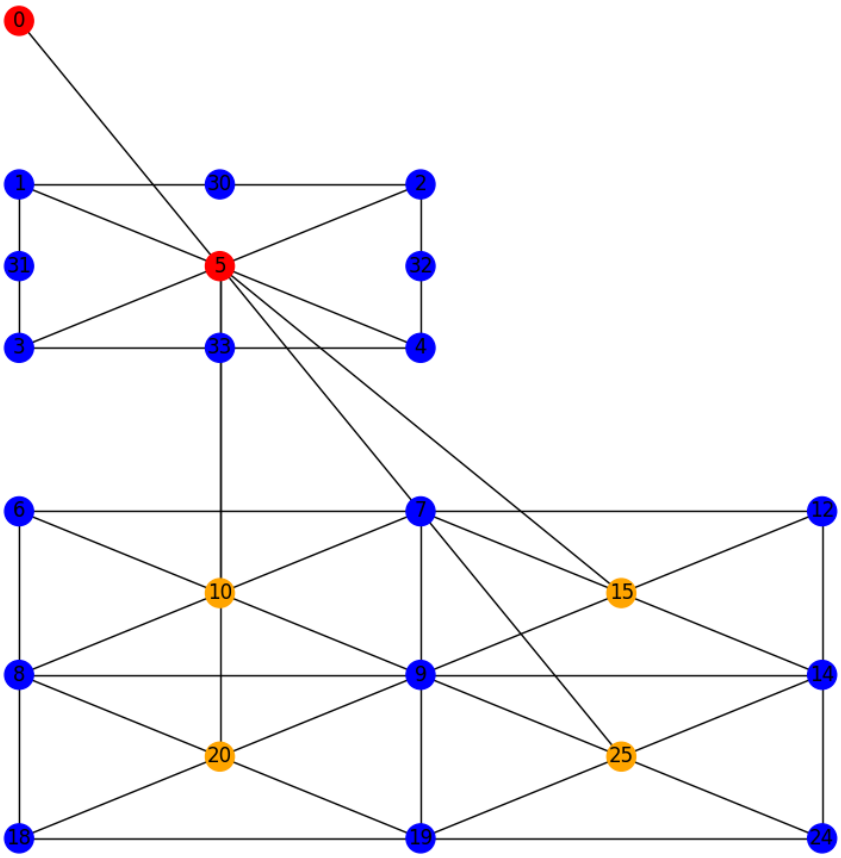
test7

f) czy zostało sprawdzone czy nowy graf umieszczony jest na poprawnym poziomie



test7

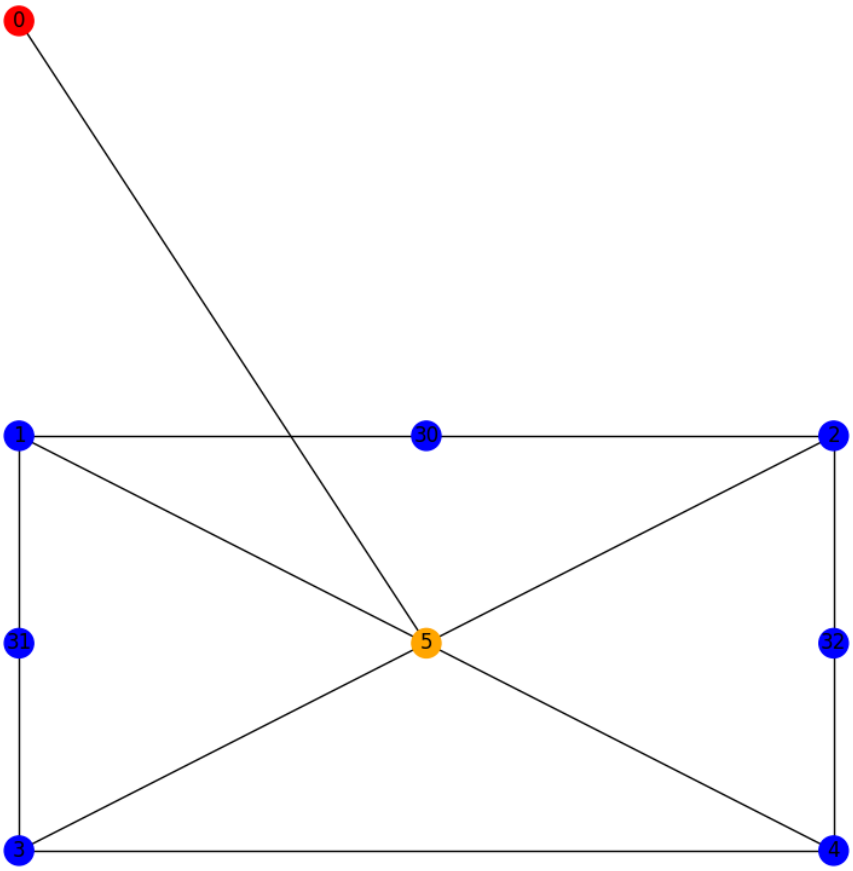
P6



test1

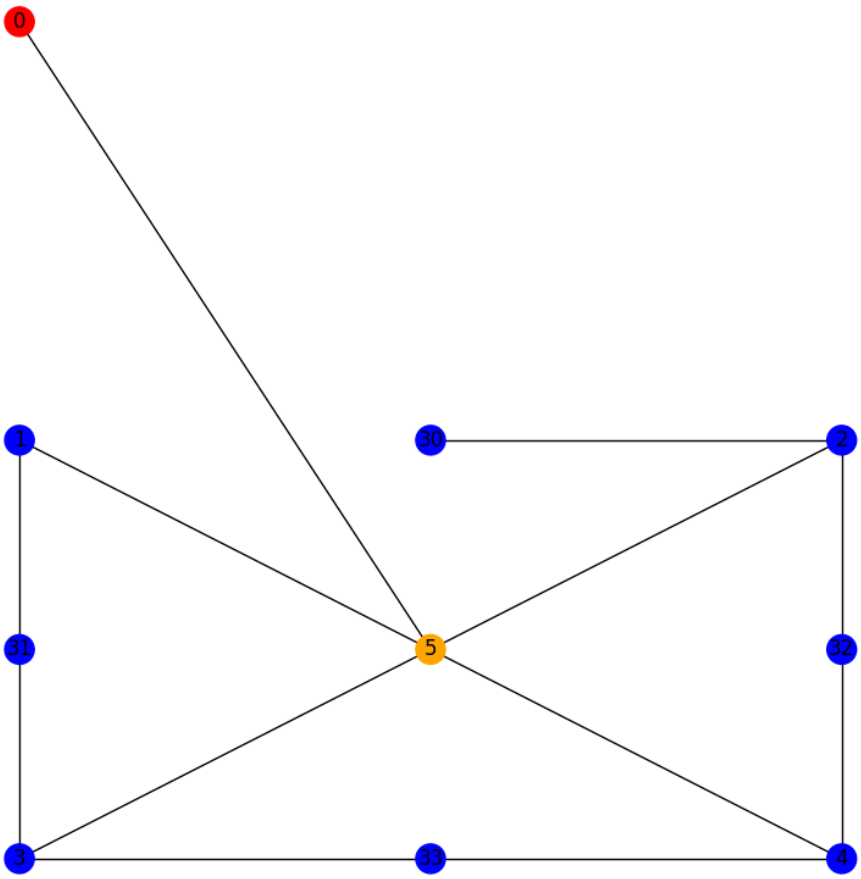
1. Czy produkcja sprawdza czy graf (podgraf grafu) do którego chcemy zastosować produkcję jest izomorficzny z grafem lewej strony produkcji (czy da się ją wykonać)?

a) czy zmiana grafu do którego stosujemy produkcję poprzez usunięcie losowego wierzchołka nie psuje tego mechanizmu



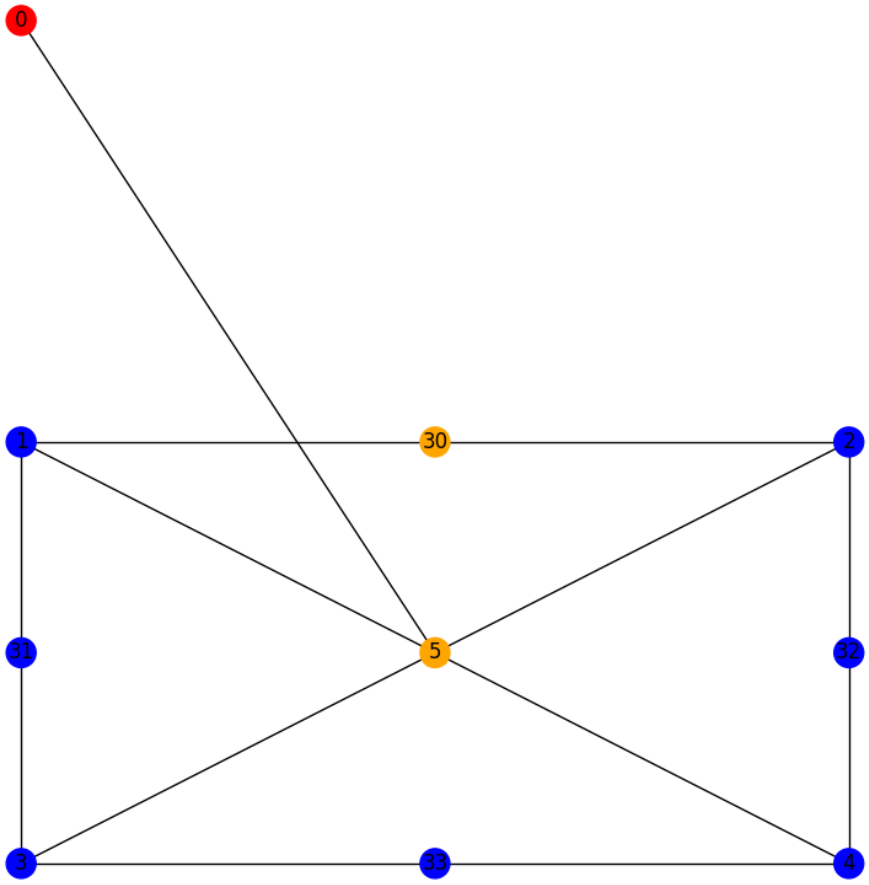
test2

b) czy zmiana grafu do którego stosujemy produkcję poprzez usunięcie losowej krawędzi nie psuje tego mechanizmu



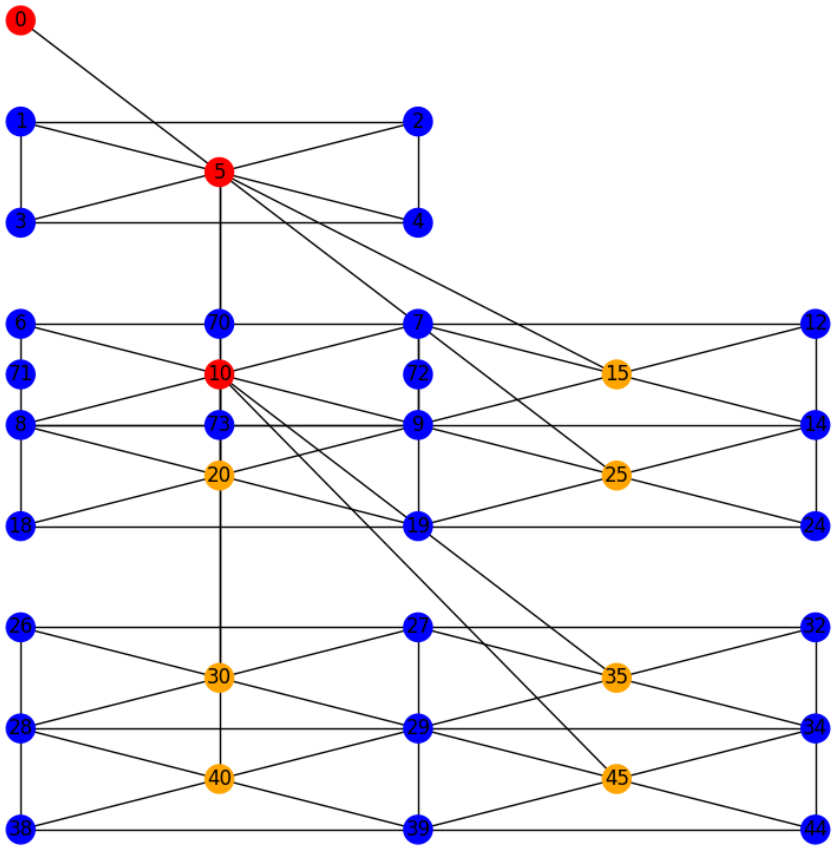
test3

c) czy zmiana grafu do którego stosujemy produkcję poprzez zmianę etykiety losowego wierzchołka nie psuje tego mechanizmu



test4

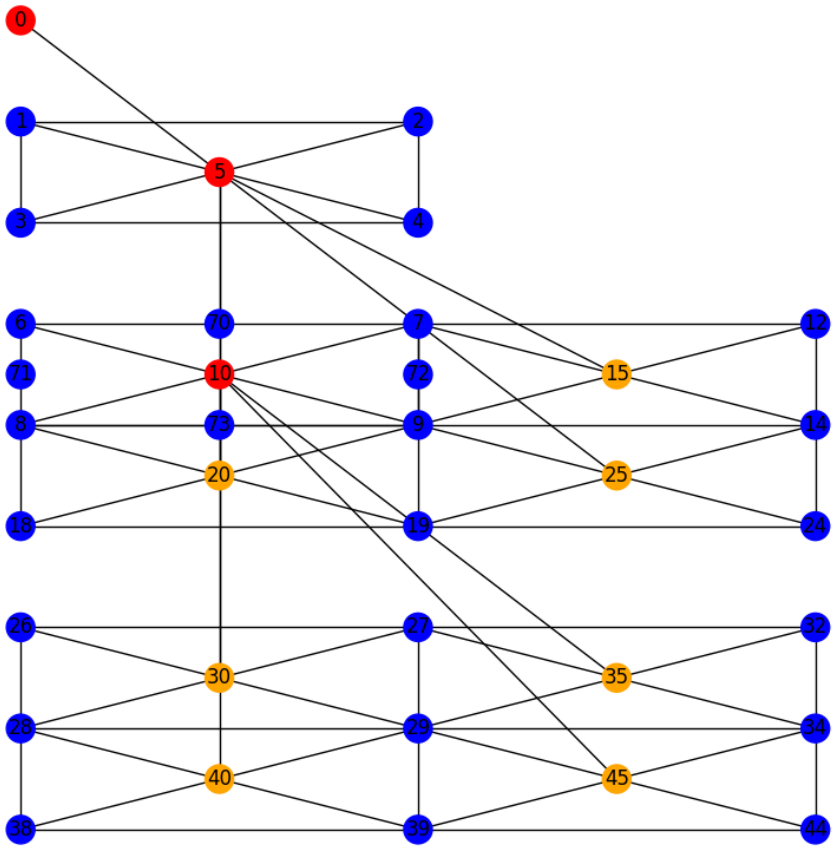
d) czy umieszczenie grafu izomorficznego z grafem lewej strony jako podgrafu większego grafu nie psuje tego mechanizmu



test5

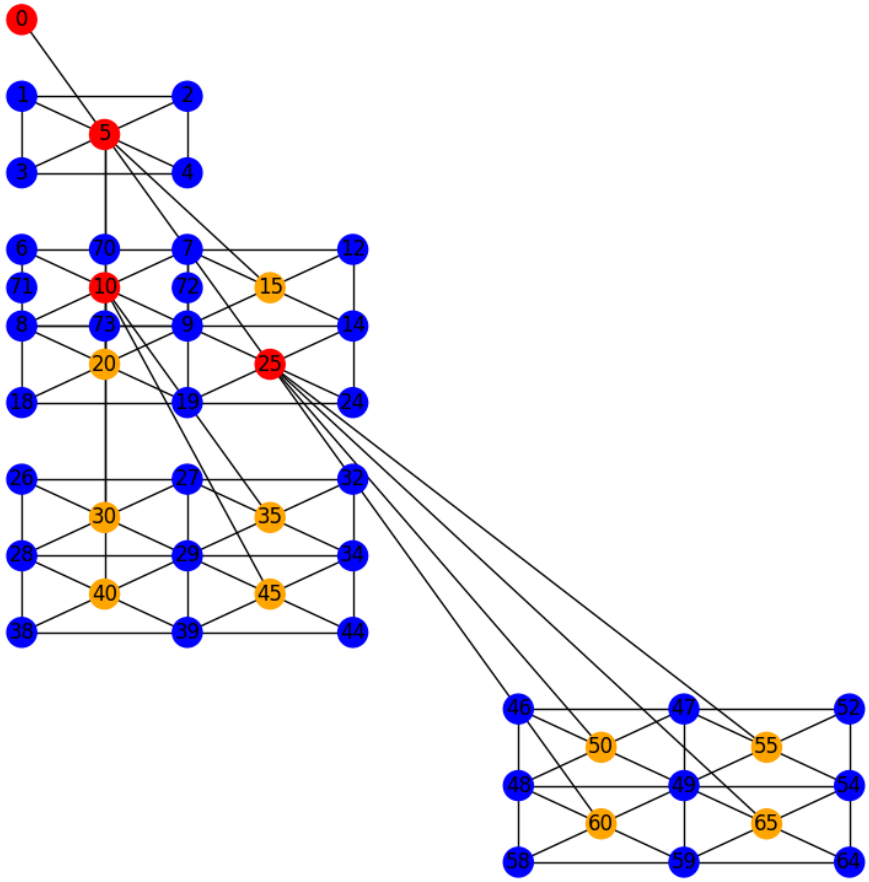
2. Czy produkcja dobrze się wykonała?

a) czy jeśli graf izomorficzny z grafem lewej strony jest umieszczony jako podgraf większego grafu, to czy produkcja nie „uszkadza" większego grafu



test5

b) czy jeśli graf izomorficzny z grafem lewej strony jest umieszczony w jako podgraf większego grafu, to czy produkcja dobrze transformuje osadzenie



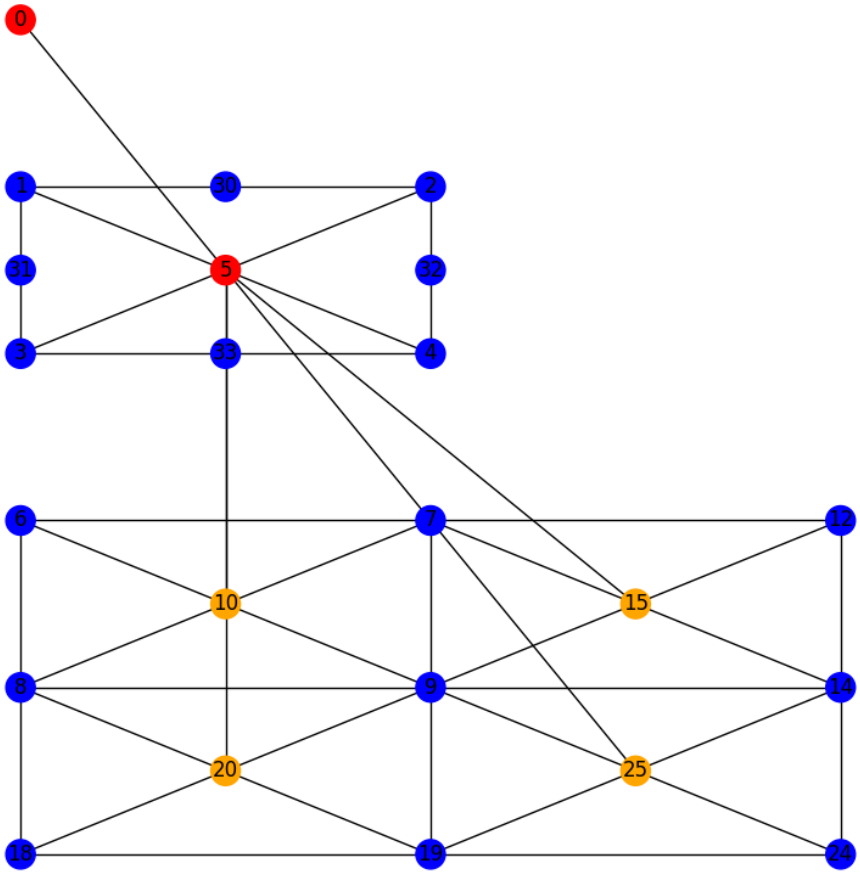
test6

c) czy graf izomorficzny z grafem prawej strony jest poprawny (czy ma wszystkie wierzchołki, krawędzie i poprawne etykiety)

Z powodu braku połączeń pomiędzy małymi kwadratami połaczenie musi wystarczyć jako istnienie wierzchołków na pozycji x, y i połączeń wewnątrz mniejszych kwadratów

```
for x in [0, 1, 2]:
    for y in [-3, -4, -5]:
        vertex = find_vertice_with_coordinates_and_remove_duplicates(x, y, lower_squares)
        assert vertex.label == VertexLabel.E
for x in [0.5, 1.5]:
    for y in [-3.5, -4.5]:
        vertex = find_vertice_with_coordinates_and_remove_duplicates(x, y, lower_squares)
        assert vertex.label == VertexLabel.I
```

```
assert lower_squares[0].edges == [(9, 8), (8, 6), (6, 7), (7, 9), (9, 10), (8, 10), (6, 10),
assert lower_squares[1].edges == [(9, 7), (7, 12), (12, 14), (14, 9), (9, 15), (7, 15), (12,
assert lower_squares[2].edges == [(18, 8), (8, 9), (9, 19), (19, 18), (18, 20), (8, 20), (19
assert lower_squares[3].edges == [(9, 14), (14, 24), (24, 19), (19, 9), (9, 25), (14, 25),
```



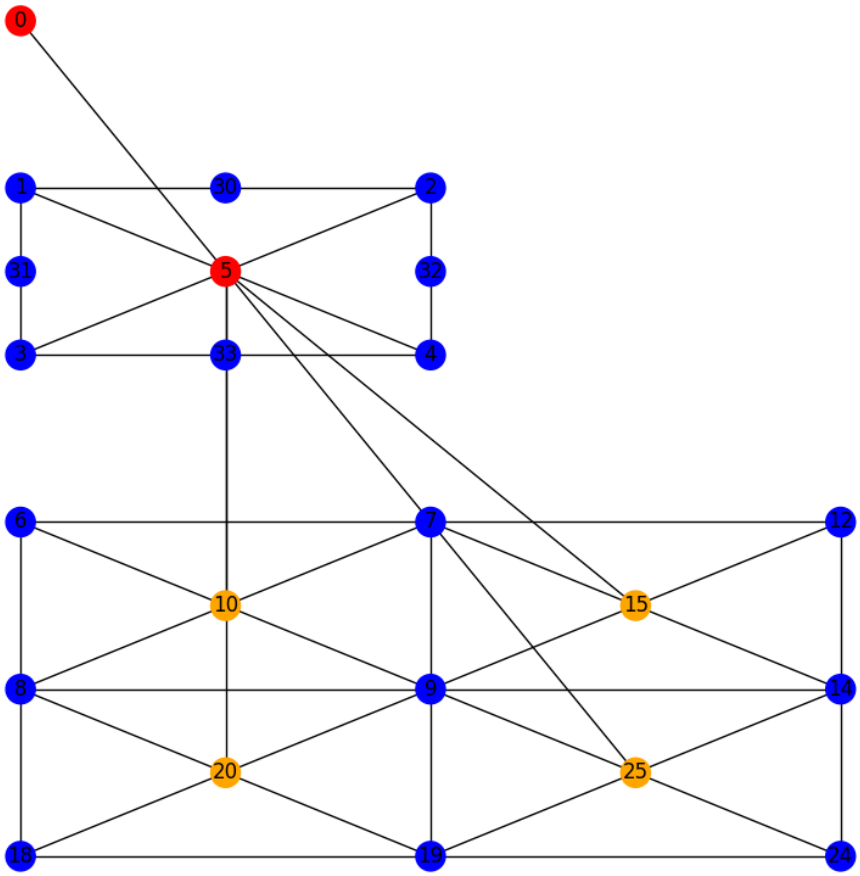
test7

d) czy współrzędne nowych wierzchołków w tym grafie są poprawne

Wierzchołki da się wyliczyć za pomocą wzoru:

- kolumna: od `0` do `2 ^ poziom`
- rząd: od `-sum for i in range(0, poziom): 2^poziom` do `-(sum for i in range(0, poziom) 2^poziom) - (2 ^ poziom)` (intuicja: suma wszystkich coraz większych kwadratów)

```
for s in lower_squares:
    assert s.layer_number == lower_layer_number
for x in [0, 1, 2]:
    for y in [-3, -4, -5]:
        vertex = find_vertice_with_coordinates_and_remove_duplicates(x, y, lower_squares)
        assert vertex.label == VertexLabel.E
for x in [0.5, 1.5]:
    for y in [-3.5, -4.5]:
        vertex = find_vertice_with_coordinates_and_remove_duplicates(x, y, lower_squares)
        assert vertex.label == VertexLabel.I
```

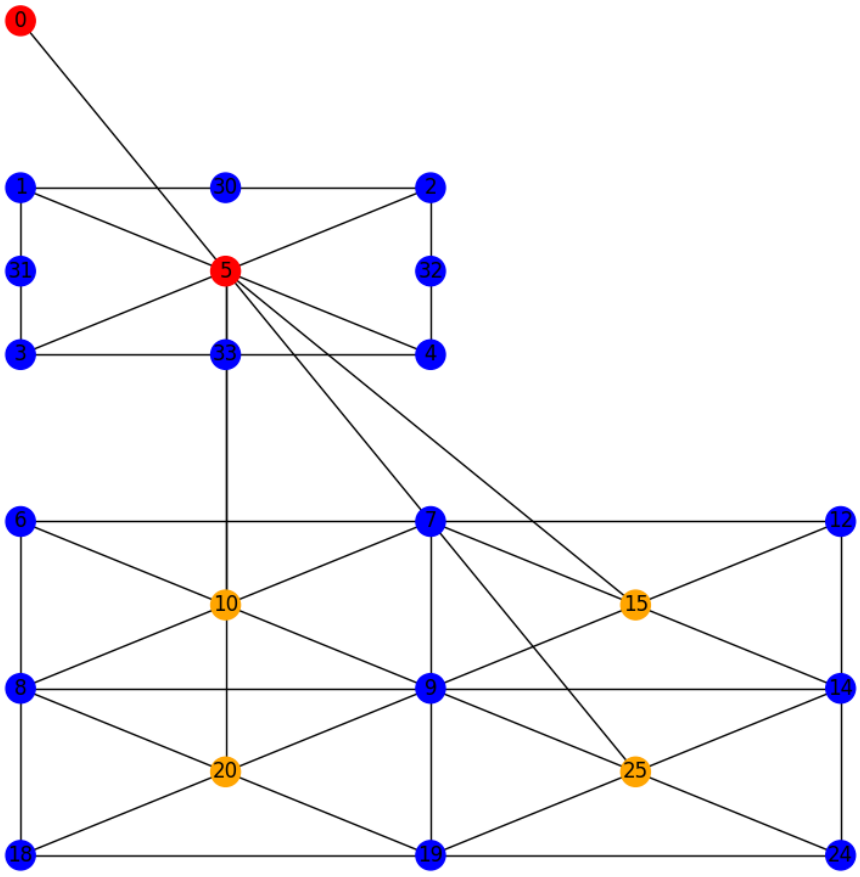


test7

e) czy nowy graf umieszczony jest na poprawnym poziomie

```
upper_layer_number = graph_fragment.layer_number
lower_layer_number = upper_layer_number + 1
```

```
lower_squares = [verticies_graph_fragment.get(x) for x in [10, 15, 20, 25]]
for s in lower_squares:
    assert s.layer_number == lower_layer_number
```



test7

3. Czy graf dobrze się rysuje?

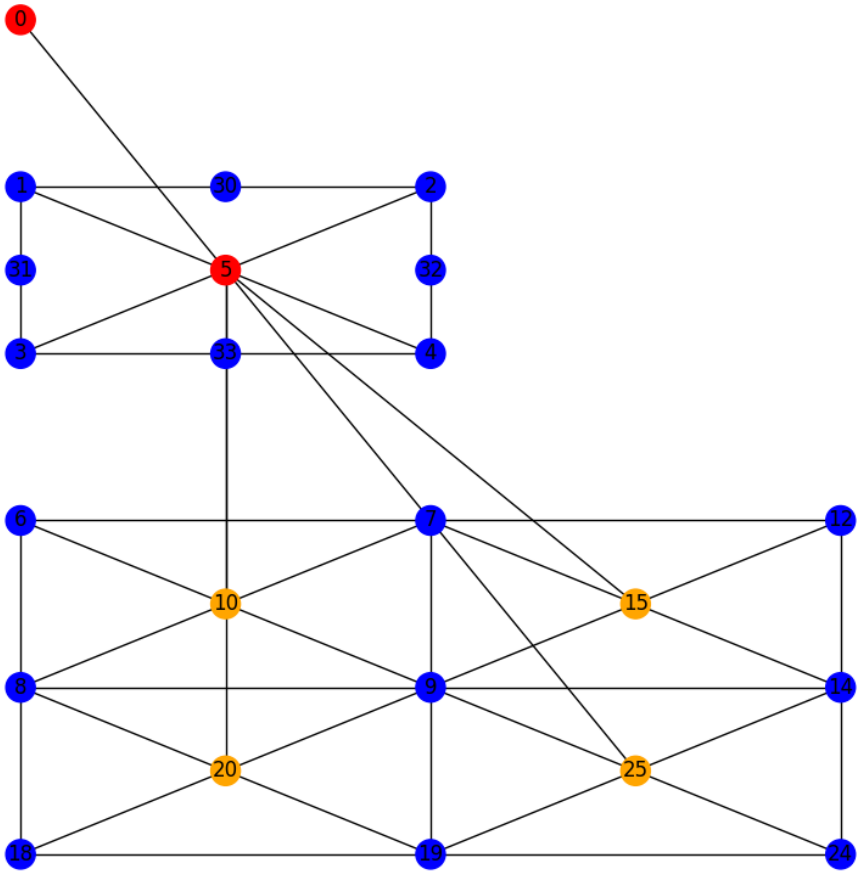
a) czy są wszystkie wierzchołki i krawędzie

Kopia 2. c)

```
for x in [0, 1, 2]:
    for y in [-3, -4, -5]:
        vertex = find_vertice_with_coordinates_and_remove_duplicates(x, y, lower_squares)
        assert vertex.label == VertexLabel.E
for x in [0.5, 1.5]:
```

```
    for y in [-3.5, -4.5]:
        vertex = find_vertice_with_coordinates_and_remove_duplicates(x, y, lower_squares)
        assert vertex.label == VertexLabel.I

assert lower_squares[0].edges == [(9, 8), (8, 6), (6, 7), (7, 9), (9, 10), (8, 10), (6, 10),
assert lower_squares[1].edges == [(9, 7), (7, 12), (12, 14), (14, 9), (9, 15), (7, 15), (12,
assert lower_squares[2].edges == [(18, 8), (8, 9), (9, 19), (19, 18), (18, 20), (8, 20), (19
assert lower_squares[3].edges == [(9, 14), (14, 24), (24, 19), (19, 9), (9, 25), (14, 25),
```



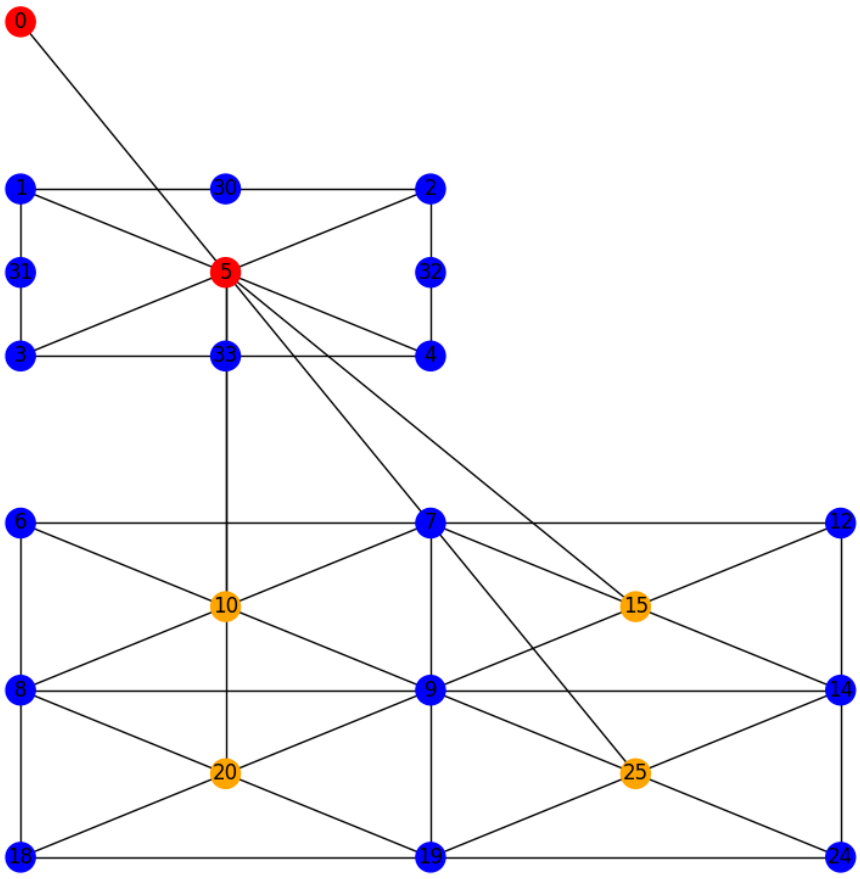
test7

b) czy wierzchołki są narysowane w poprawnych współrzędnych

Kopia 2. d)

```
for s in lower_squares:
    assert s.layer_number == lower_layer_number
```

```
for x in [0, 1, 2]:
    for y in [-3, -4, -5]:
        vertex = find_vertice_with_coordinates_and_remove_duplicates(x, y, lower_squares)
        assert vertex.label == VertexLabel.E
for x in [0.5, 1.5]:
    for y in [-3.5, -4.5]:
        vertex = find_vertice_with_coordinates_and_remove_duplicates(x, y, lower_squares)
        assert vertex.label == VertexLabel.I
```

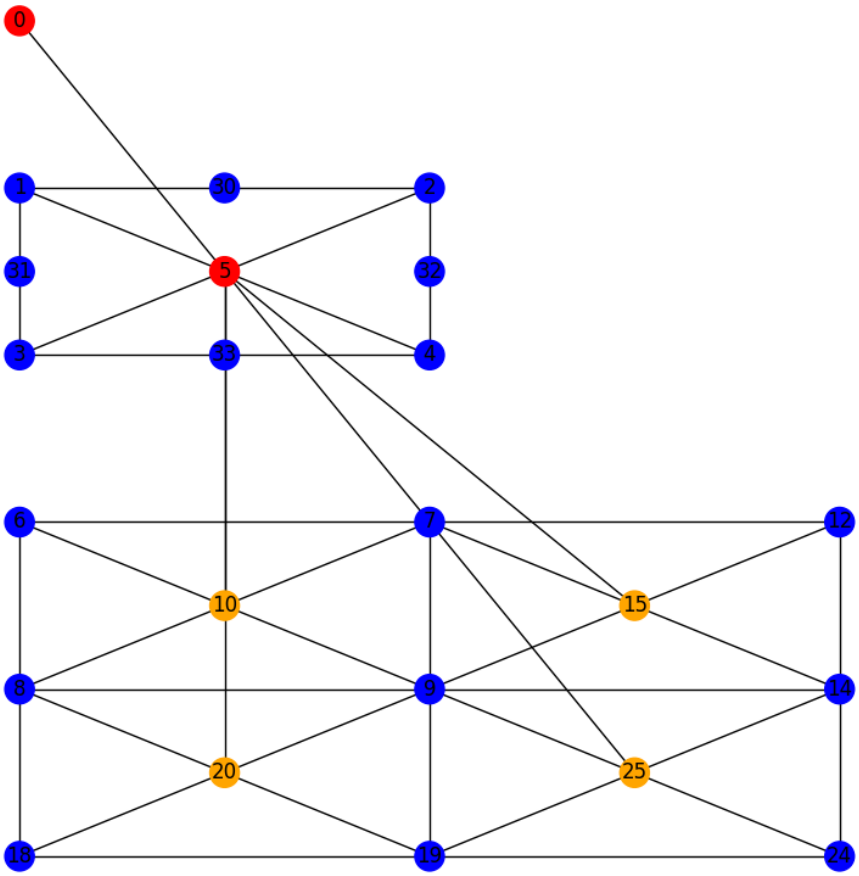


test7

c) czy da się wybierać poziom grafu do narysowania

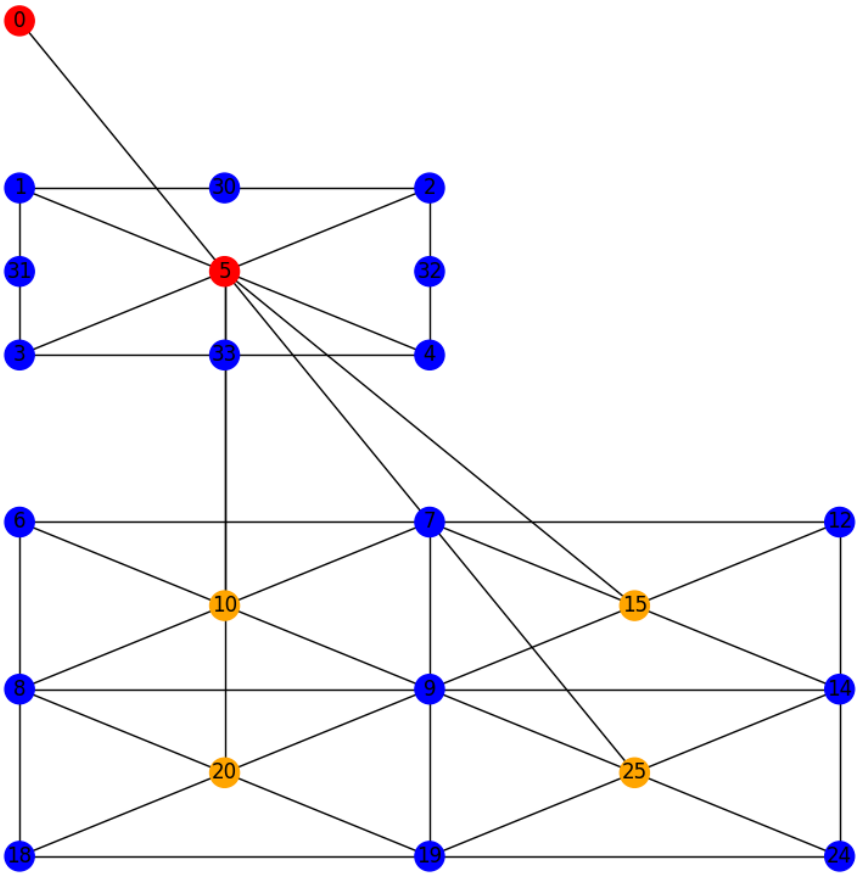
Niestety model tego nie oferuje

d) czy są narysowane etykiety wierzchołków



test7

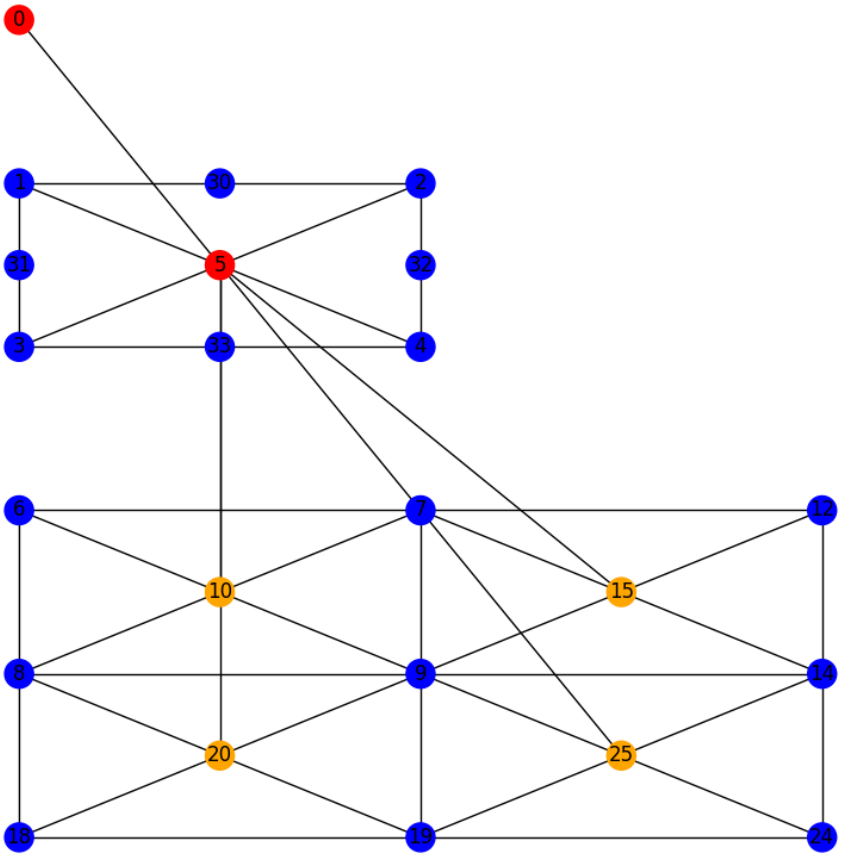
e) czy jest zaznaczone które wierzchołki mają linki do poprzedniego lub następnego poziomu



test7

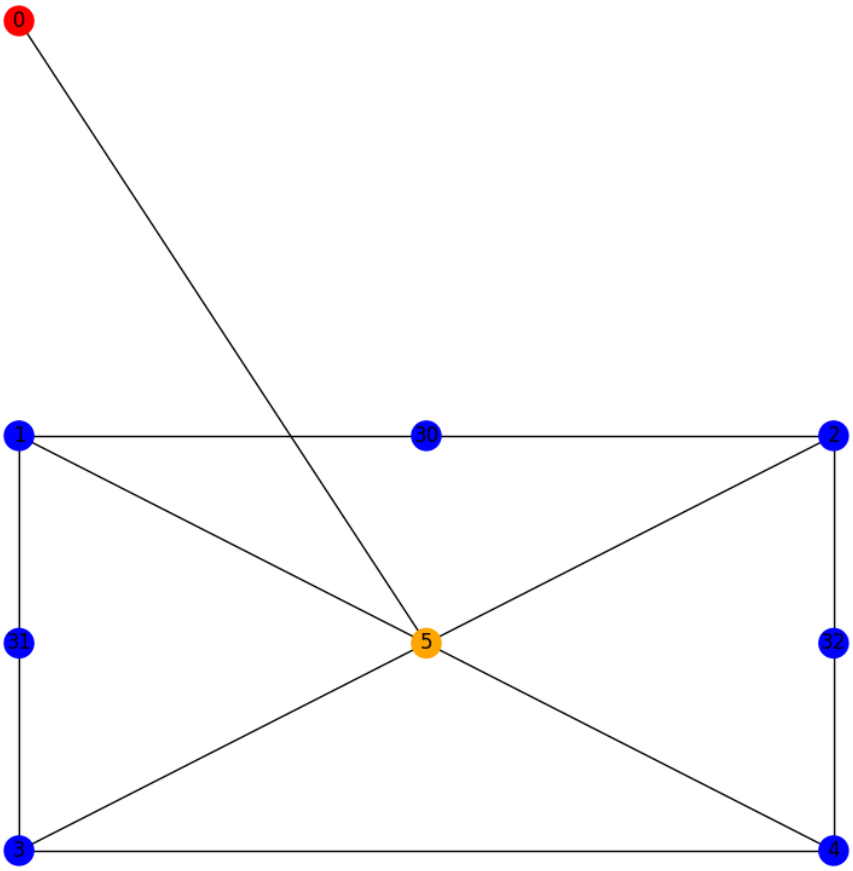
4. Czy zostały przygotowanie różne grafy do testowania

a) czy został przygotowany graf izomorficzny z grafem lewej strony produkcji, który jest poprawny



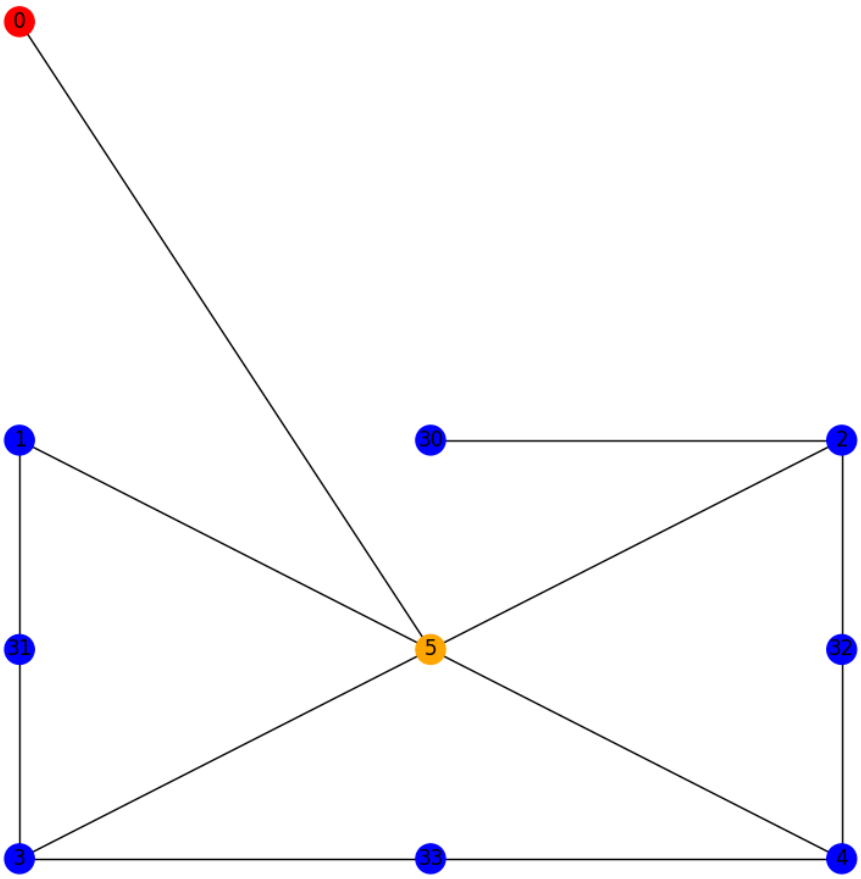
test1

b) czy został przygotowany graf izomorficzny z grafem lewej strony produkcji, który jest niepoprawny (bez jakiegoś wierzchołka)



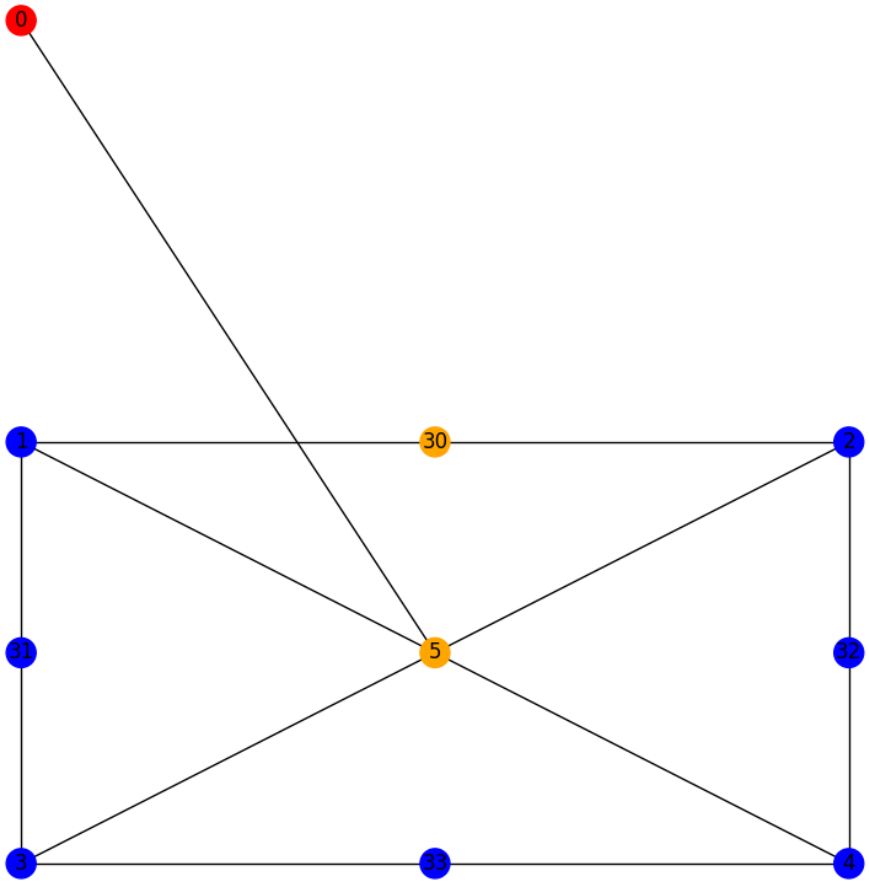
test2

c) czy został przygotowany graf izomorficzny z grafem lewej strony produkcji, który jest niepoprawny (bez jakiejś krawędzi)



test3

d) czy został przygotowany graf izomorficzny z grafem lewej strony produkcji, który jest niepoprawny (z niepoprawną etykietą)

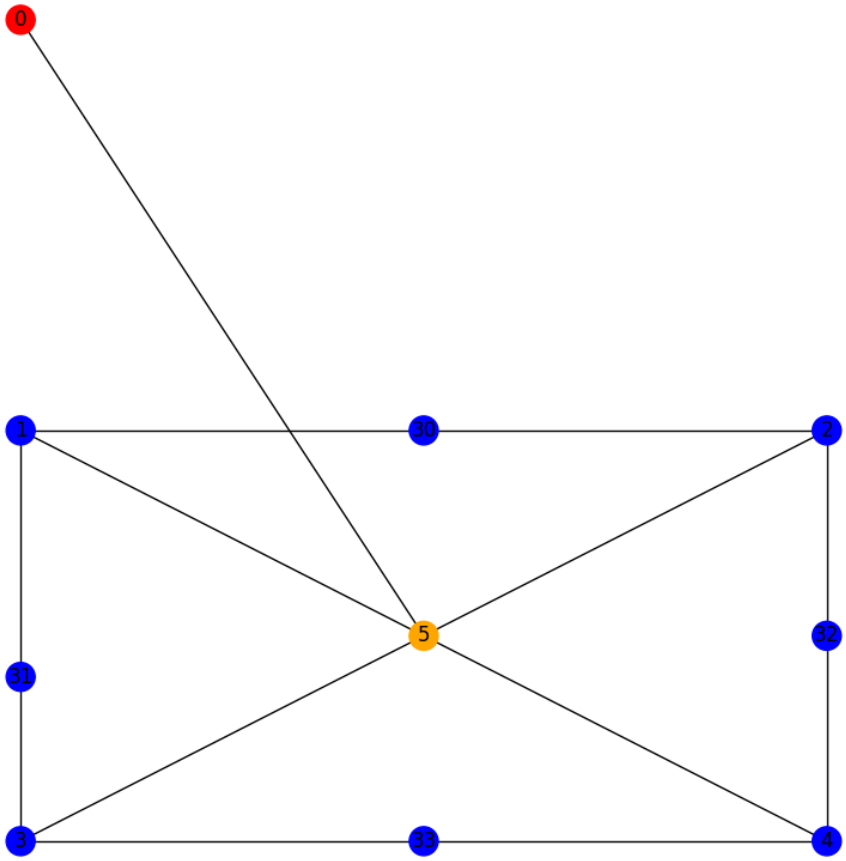


test4

e) czy został przygotowany graf izomorficzny z grafem lewej strony produkcji, który jest niepoprawny (z błędnymi współrzędnymi)

Współrzeđne wirzechołka `31` to `(0, -1.6)` zamiast `(0, -1.5)` przez co produkcja nie zostaje zastosowana.

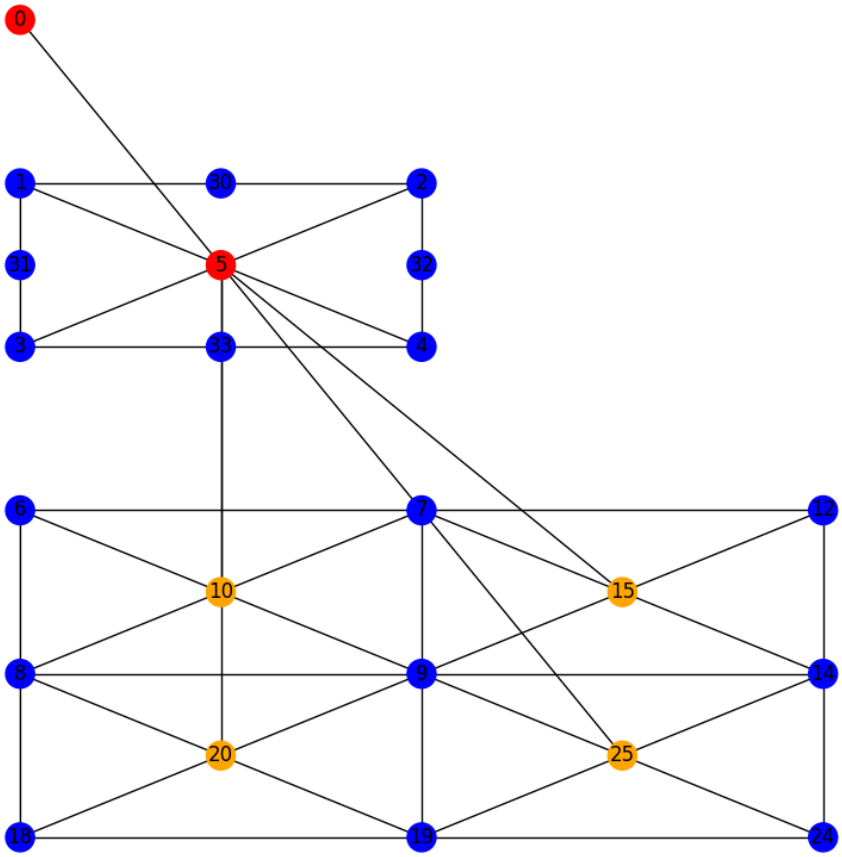
```
graph_fragment.vertices.extend(list([Vertex(0.5, -1, 30, VertexLabel(1)), Vertex(0, -1.6, 31
```



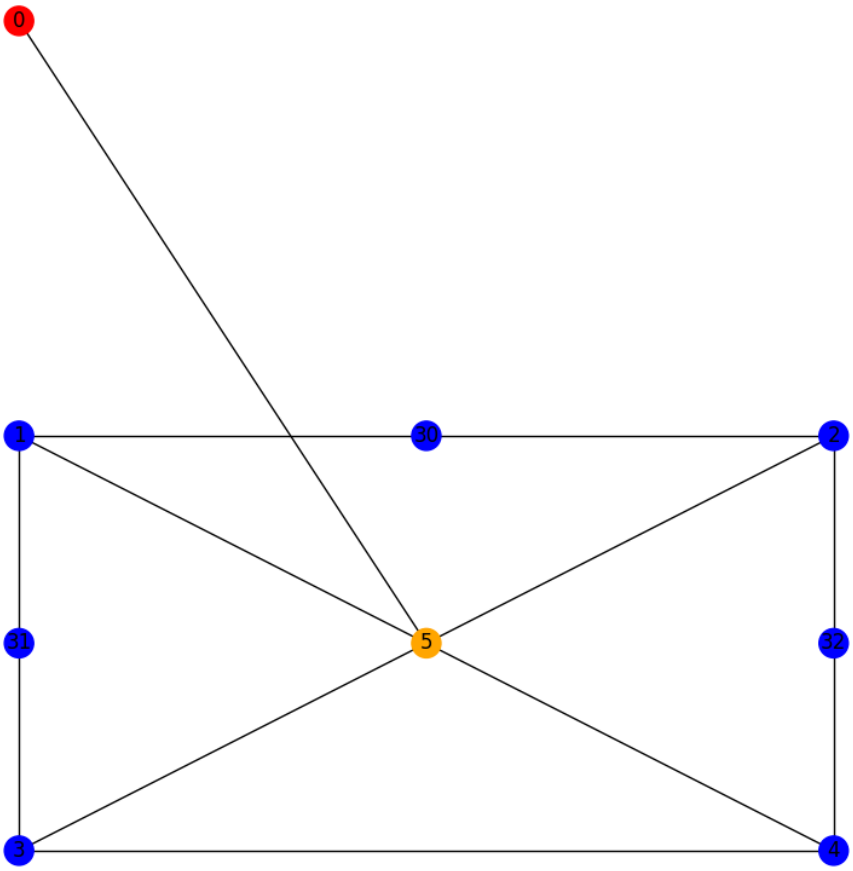
test8

5. Czy wynik produkcji został dobrze sprawdzony

a) czy zostało sprawdzone czy produkcja wykonała się na poprawnym grafie i nie została wykonana na niepoprawnym grafie?

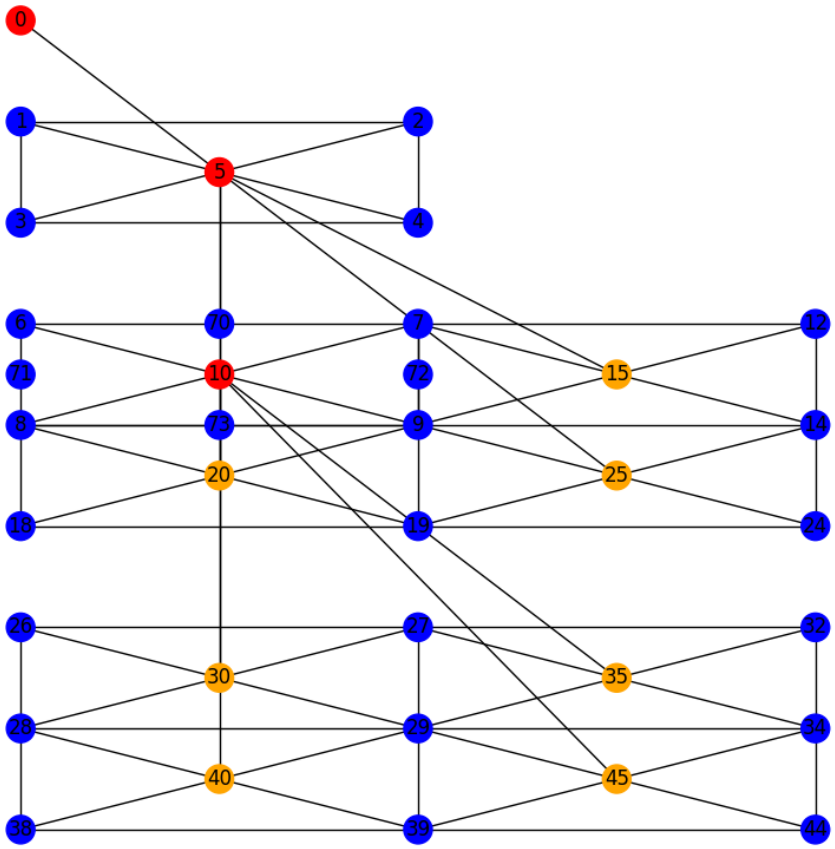


test1



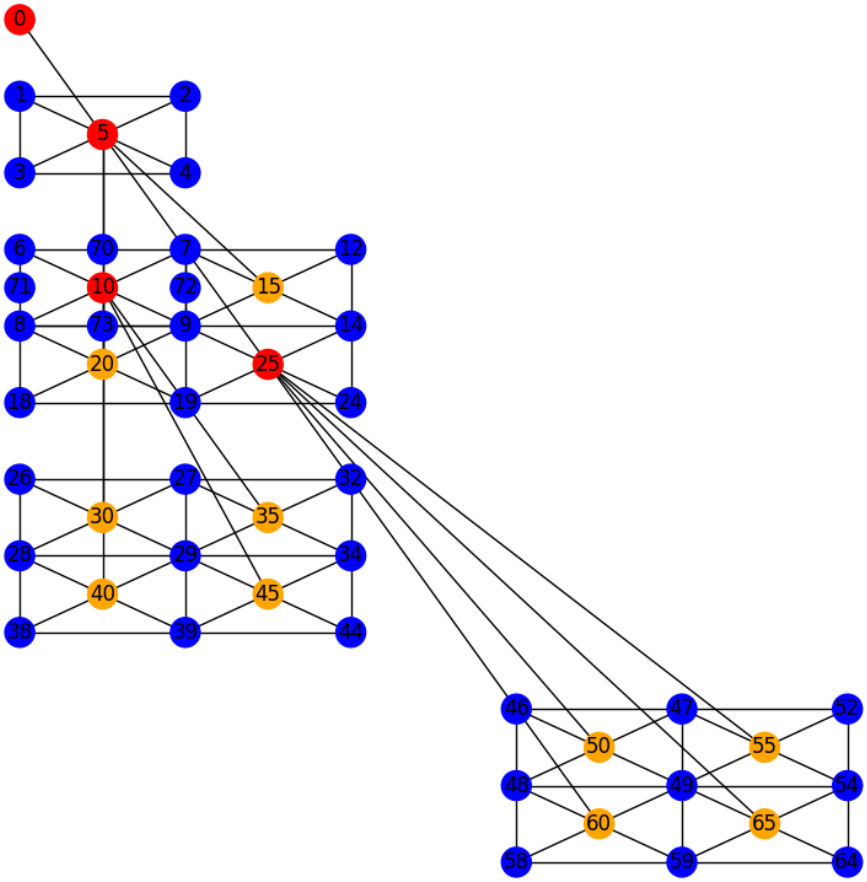
test2

b) czy zostało sprawdzone czy jeśli graf lewej strony jest umieszczony w jako podgraf większego grafu, to czy produkcja nie uszkadza większego grafu



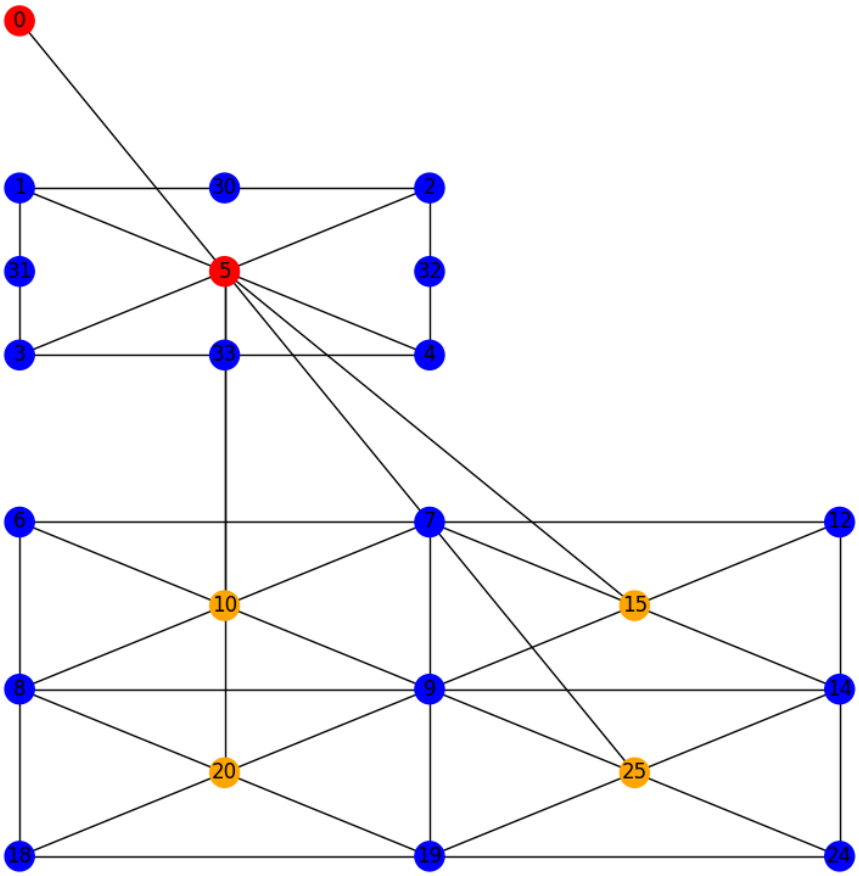
test5

c) czy zostało sprawdzone czy jeśli graf lewej strony jest umieszczony jako podgraf większego grafu, to czy produkcja dobrze transformuje osadzenie



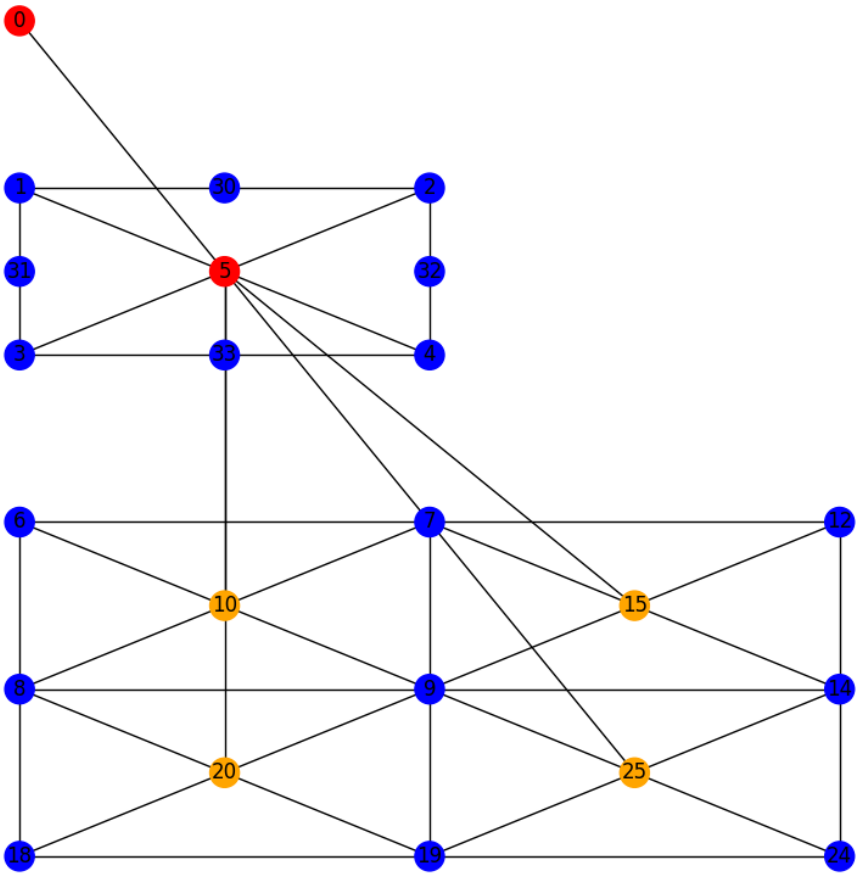
test6

d) czy zostało sprawdzone czy graf prawej strony jest poprawny (czy ma wszystkie wierzchołki, krawędzie i poprawne etykiety)



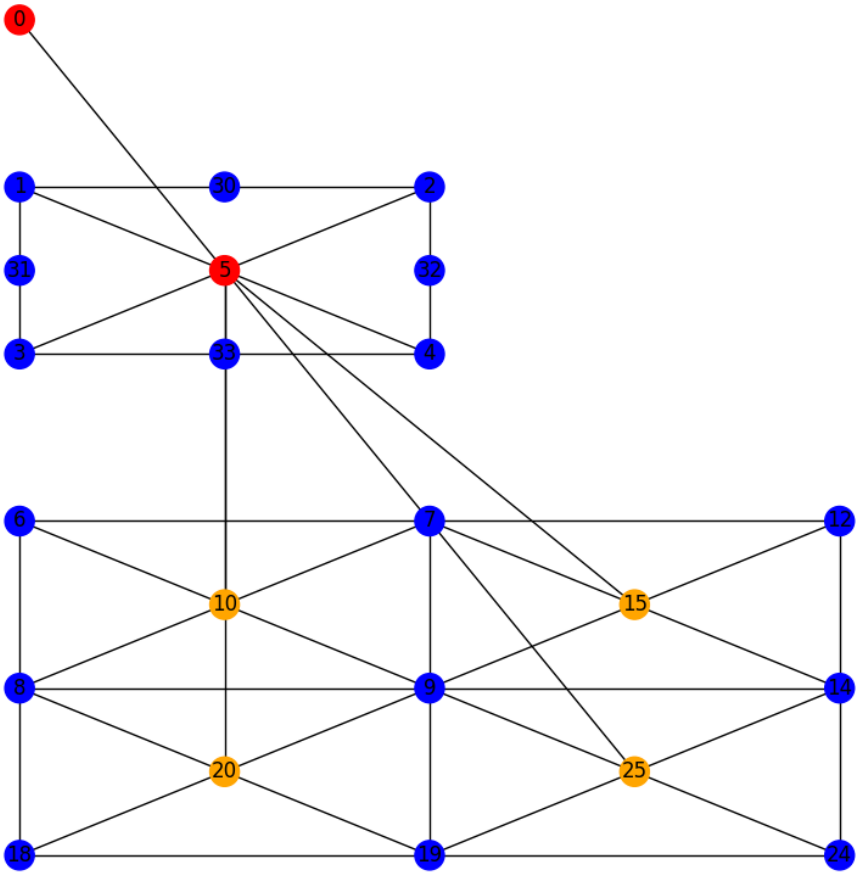
test7

e) czy zostało sprawdzone czy współrzędne nowych wierzchołków są poprawne



test7

f) czy zostało sprawdzone czy nowy graf umieszczony jest na poprawnym poziomie



test7