# Specifikace

GraphIt je knihovna funkcí v jazyce Python, která zprostředkovává základní operace pro práci s grafy, jako vytvoření nového grafu, přidávání/odebírání hran a vrcholů. Dále pak implementace algoritmů pro nalezení minimální kostry, nejkratší cesty mezi dvěma vrcholy a nalezení Eulerovského tahu. Podporované je také načítání grafů ze souboru v předepsaném formátu.

Uživatel má k dispozici jak samotnou knihovnu, tak i konzolové UI, skrz které ji lze využívat v plném rozsahu. Vzhledem ke striktnímu oddělení API a konzolového UI má knihovna potenciál pro budoucí implementaci do grafického prostředí.

# Dokumentace

## API

### Objekty

* + ***Graph***
    - Popis: Objekt pro práci s grafem
    - Props:
      * + ***matrix***

Matice sousednosti

* + - * + ***vertices***

Slovník vrcholů, klíčem je id, hodnotou je pak objekt Vertex

* + - * + ***edges***

Pole hran

* + - Metody:
      * + ***load()***

Načte graf z textového vstupu

* + - * + ***set\_edge()***

Přidá hranu mezi dva vrcholy, tj. udělá záznam do matice sousednosti

* + - * + ***remove\_edge()***

Odstraní hranu mezi dvěma vrcholy

* + - * + ***set\_edges()***

Přidá hrany z pole na vstupu

* + - * + ***set\_vertex()***

Přidá vrchol, rozšíří matici sousednosti

* + - * + ***set\_vertices()***

Přidá vrcholy z pole na vstupu

* + - * + ***get\_minSpanningTree()***

Vytvoří nový graf, který bude kostrou původního

* + - * + ***is\_connected()***

Vrací boolean, zda je graf souvislý

* + - * + ***is\_tree()***

Vrací boolean, zda je graf strom

* + - * + ***find\_route ()***

Vrací list of tuples, kde každá položka listu obsahuje id vrcholu a vzdálenost k němu

* + - * + ***import\_file()***

Načítá graf ze souboru podle definovaného formátu

Formát:

{

"graph": {

"vertices": [

{

"id": "id1",

"name": "nazev1"

},

{

"id": "id2",

"name": "nazev2"

},

],

"edges": [

{

"vertex\_1": {

"id": "id1"

},

"vertex\_2": {

"id": "id2"

},

"weight": 120

}

* + - * + ***export\_file()***

Načítá graf do souboru podle definovaného formátu

* + - * + ***contains\_cycle()***

Vrací, zda obsahuje graf cyklus

* + - * + ***get\_neighbours ()***

Vrací sousedy vrcholu a ohodnocení hran k nim

* + - * + ***get\_vertexEdges ()***

Vrací hrany vedoucí od vrcholu

* + - * + ***get\_vertexByIndex ()***

Vrací vrchol podle indexu v matici sousednosti

* + - * + ***clear()***

Vymaže všechny vrcholy a hrany grafu

* + - * + ***print()***

Vypíše graf do konzole po vrcholech a hranách

* + - * + ***clone()***

Deepcopy instance

* + ***Edge***
    - Popis: Objekt hrany grafu
    - Props:
      * + ***vertex\_1***

Vrátí jeden z vrcholů hrany

* + - * + ***vertex\_2***

Vrátí druhý z vrcholů hrany

* + - * + ***weight***

Vrátí hodnotu (váhu) hrany v ohodnoceném grafu. Pokud není ohodnocený, pak vrací 1

* + ***Vertex***
    - Popis: Objekt vrcholu grafu
    - Props:
      * + ***\_index***

Pozice vrcholu v matici sousednosti

* + - * + ***id***

Id vrcholu

* + - * + ***name***

Název vrcholu

* + ***Matrix***
    - Popis: Objekt pro práci s maticemi
    - Props:
      * + ***array***

Pole matice

* + - Metody:
      * + ***load()***

Načte matici z textového vstupu

* + - * + ***can\_multiply()***

Vrací, zda lze matice násobit

* + - * + ***multiply\_left()***

Násobit matici zleva

* + - * + ***multiply\_right()***

Násobit matici zprava

* + - * + ***add()***

Přičíst matici

* + - * + ***substract()***

Odečíst matici

* + - * + ***get\_value()***

Hodnota matice v konkrétním řádku a sloupci

* + - * + ***multiply\_constant()***

Vynásobit matici konstantou

* + - * + ***transpose()***

Transponovat matici

* + - * + ***to\_string()***

Převede matici na textový výstup

* + - * + ***clone()***

Deepcopy instance

## UI příkazy

* stop
* matrix.new – nová instance matice
* matrix.print – zobrazí aktuální instanci
* matrix.load – načte matici ze vstupu ve formátu [1,1,1;2,2,2]
* matrix.tr – transpozice matice
* matrix.mlp.left […] – vynásobení aktuální instance zadanou maticí zleva
* matrix.mlp.right […] – vynásobení aktuální instance zadanou maticí zprava
* matrix.add – přičtení matice k aktuální instanci
* matrix.sub – odečtení matice od aktuální instance
* graph.new – nová instance grafu
* graph.print – zobrazí aktuální instanci grafu
* graph.import *filename –* načtení grafu ze souboru
* graph.export *filename –* načtení grafu do souboru
* graph.vertex.set *id;name –* přidání vrcholu grafu
* graph.vertex.remove *id –* odebrání vrcholu grafu
* graph.edge.set *id1;id2;weight –* přidání hrany
* graph.edge.remove *id1;id2 –* odebrání hrany
* graph.findroute *id1;id2* – cesta se vzdálenostmi mezi dvěma vrcholy
* graph.minspntree – minimální kostra grafu

## Příklady

### Práce s aritmetickými výrazy a maticemi

(5+4)\*9/(80+1)

[1,2;1,2;1,2]\*[3;1]

[1,3;5,2;2,4]\*[1,1,3;3,2,4]

matrix.tr

### Vytvoření nového grafu v konzolovém UI

graph.new

graph.vertex.set 1;Praha

graph.vertex.set 2;Plzeň

graph.vertex.set 3;”Kostelec nad Černými Lesy”

graph.vertex.set 4;”Bratislava”

graph.vertex.remove 4

graph.edge.set 1;2;90

graph.edge.set 1;3;90

graph.edge.remove 1;3

graph.edge.set 1;3;40

graph.print

### Základní operace s grafem v konzolovém UI

graph.new

graph.import import

graph.findroute A;T

graph.findroute K;Z

graph.minspntree

graph.export export