1. Jak uruchomić program?

Z powodu użycia bibliotek `matplotlib` i `networkx` na początku należy stworzyć środowisko wirtualne i zainstalować wymagania z dołączonego pliku `requirements.txt`.

```
cd traces_theory/
python3 -m venv .
source bin/activate
pip3 install -r requirements.txt
```

W folderze z zadaniem załączone są przykładowe dane w folderach `input1` oraz `input2`. Faktyczny program można uruchomić za pomocą

```
python3 main.py input1/transactions.txt input1/alphabet.txt baadcb
python3 main.py input2/transactions.txt input2/alphabet.txt acdcfbbe
```

2. Zawartość programu

Rozwiązanie podzieliłem na plik `util.py` oraz `main.py`. `util.py` zawiera wszystkie funkcje pomocnicze używane w głównej części programu, czyli:

- `get_transactions(transaction_file: str) -> dict[str, list[str]]` - funkcja odczytująca transakcję z pliku o podanej ścieżce.
- `get_alphabet(alphabet_file: str) -> list[str]` - funkcja odczytująca alfabet z pliku o podanej ścieżce.
- `verify_input(transactions:dict[str, list[str]], alphabet:list[str]) -> bool` - funkcja sprawdzająca zgodność transakcji z alfabetem.
- `verify_word(alphabet:list[str], word: str) -> bool` - funkcja weryfikująca zgodność zadanego słowa z alfabetem.
- `get_dependent_transactions(transactions: dict[str, list[str]]) -> set[tuple[str, str]]` - funkcja zwracająca zależne od siebie transakcje.
- `get_dependencies(dependent:set[tuple[str, str]], alphabet:list[str]) -> dict[set[str]]` - funkcja zwracająca słownik mapujący transakcję na zbiór zależnych od nich transakcji.
- `fnf_calculation(dependencies: dict[set[str]], word: str, alphabet: list[str]) -> list[list[str]]` - funkcja obliczająca FNF dla zadanego słowa.

W `main.py` znajduje się tylko funkcja główna:

- `main() -> None` - funkcja która kolejno pobiera dane z plików tekstowych, oblicza zbiory D oraz I, oblicza FNF zadanego słowa i rysuje graf końcowy.

3. Wyniki programu

3.1. Przykład pierwszy

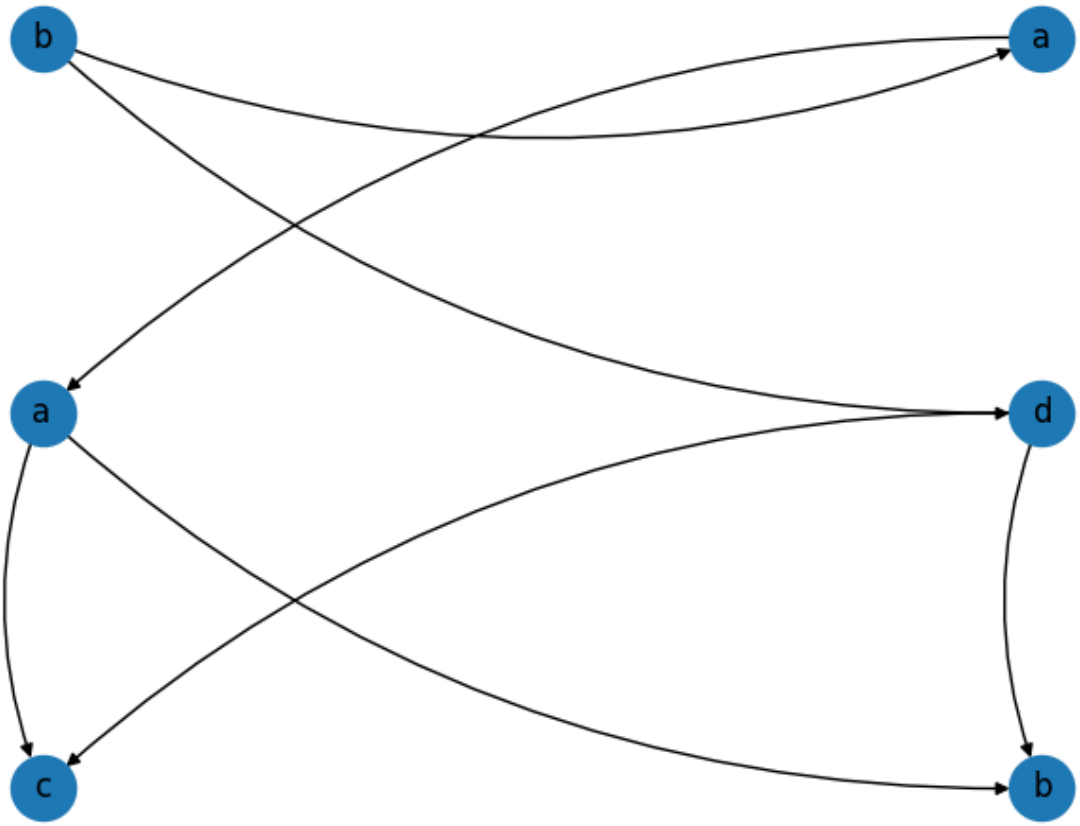
Dla wywołania

```
python3 main.py input1/transactions.txt input1/alphabet.txt baadcb
```

Otrzymujemy

```
D = {('a', 'a'),('a', 'b'),('a', 'c'),('b', 'a'),('b', 'b'),('b', 'd'),('c',
'a'),('c', 'c'),('c', 'd'),('d', 'b'),('d', 'c'),('d', 'd')}
I = {('a', 'd'),('b', 'c'),('c', 'b'),('d', 'a')}
```

```
FNF([baadcb]) = (b)(ad)(a)(bc)
```



Rysunek 1: Graf dla danych z folderu `input1`

3.2. Przykład drugi

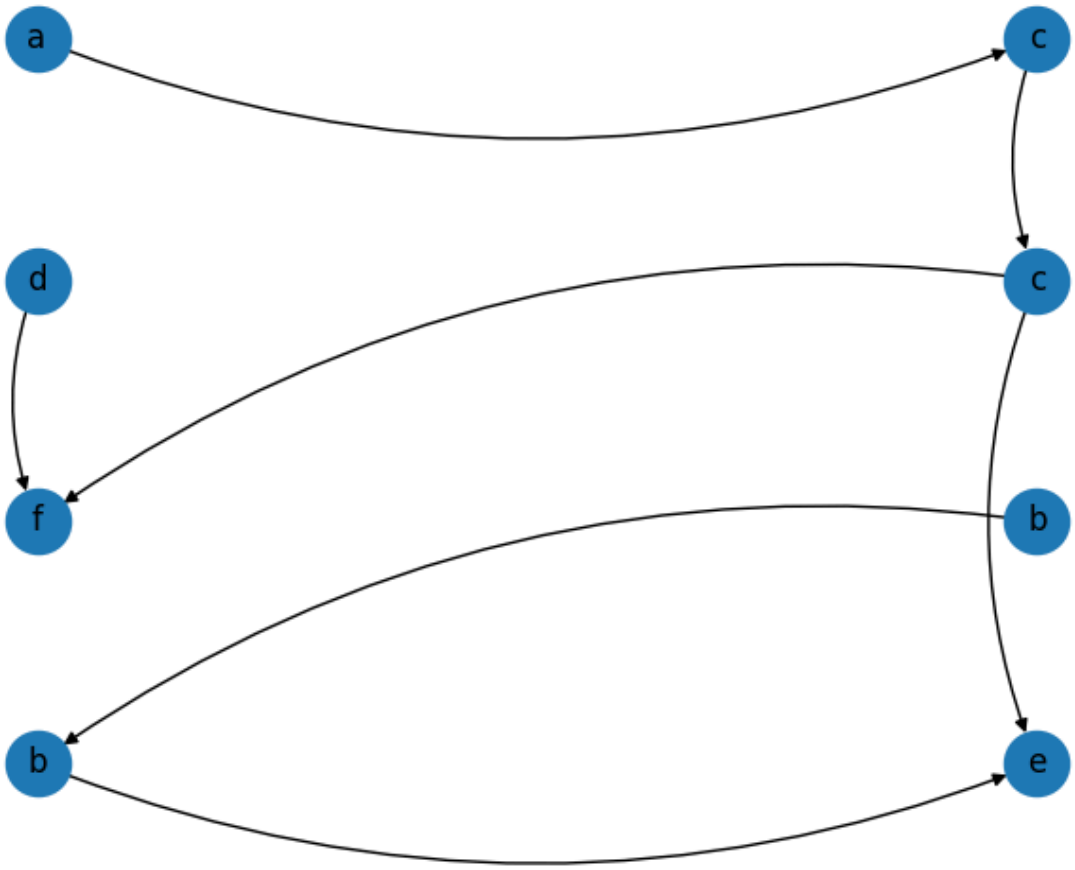
Dla wywołania

```
python3 main.py input2/transactions.txt input2/alphabet.txt acdcfbbe
```

Otrzymujemy

```
D = {('a', 'a'),('a', 'c'),('a', 'f'),('b', 'b'),('b', 'e'),('c', 'a'),('c',
'c'),('c', 'e'),('c', 'f'),('d', 'd'),('d', 'f'),('e', 'b'),('e', 'c'),('e',
'e'),('f', 'a'),('f', 'c'),('f', 'd'),('f', 'f')}
I = {('a', 'b'),('a', 'd'),('a', 'e'),('b', 'a'),('b', 'c'),('b', 'd'),('b',
'f'),('c', 'b'),('c', 'd'),('d', 'a'),('d', 'b'),('d', 'c'),('d', 'e'),('e',
'a'),('e', 'd'),('e', 'f'),('f', 'b'),('f', 'e')}
```

```
FNF([acdcfbbe]) = (abd)(bc)(c)(ef)
```



Rysunek 2: Graf dla danych z folderu `input2`