Politechnika Warszawska

Algorytmy i struktury danych

Laboratorium 3

Projekt "Kopce"

Informatyka – Inteligentne systemy

Paweł Sarnacki 305290

Piotr Niedziałek 304474

Prowadzący: dr inż. Łukasz Skonieczny

Warszawa 2023

1. Wstęp

Program został napisany w języku python z wykorzystaniem aplikacji Visual Studio Code. W programie tworzone są kopce 2-arne, 3-arne i 4-arne dla ‘n’ losowych liczb od 0 do 300 tysięcy. Ilość liczb wynosi od 10 tysięcy do 100 tysięcy z krokiem 10 tysięcy.
W programie zaimplementowano wstawianie liczb do kopca, usuwanie szczytu kopca i wyświetlanie kopca. Na wykresach przedstawiono pomierzony czas wstawiania liczb i usuwania szczytu kopców dla ‘n’ liczb. Program był pisany w wersji python 3.9.13, a wykorzystane biblioteki to: time, random, networkx, matplotlib.pyplot, pygraphviz.

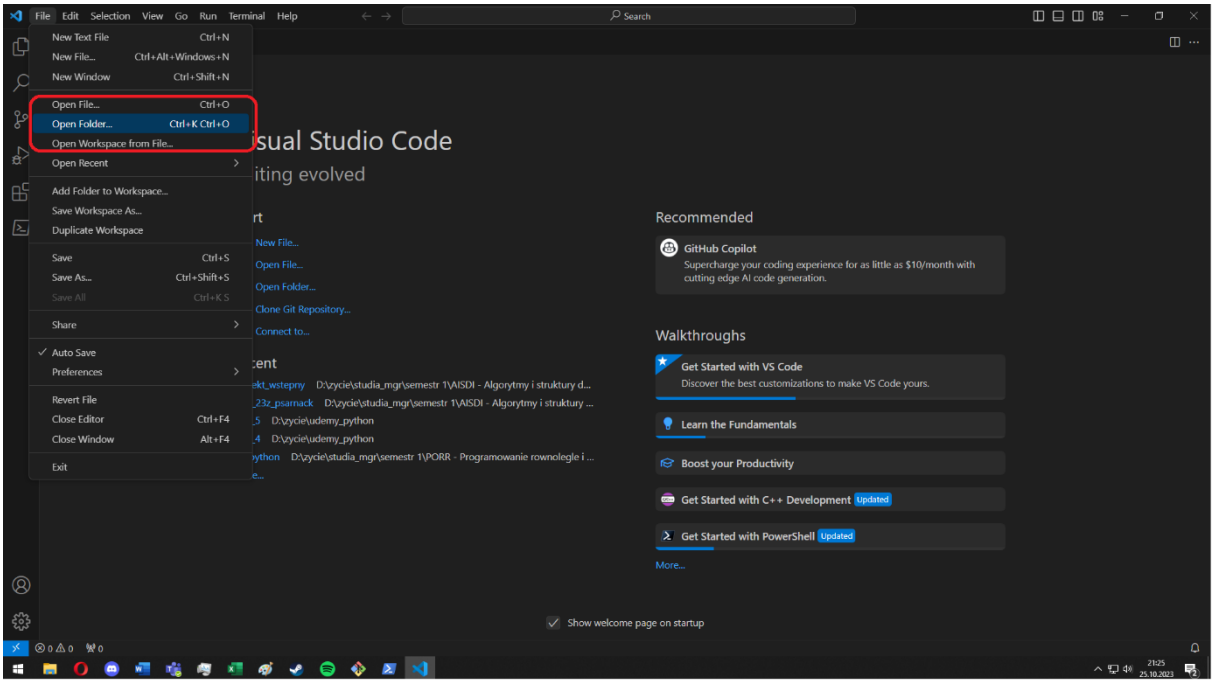
2. Struktura projektu

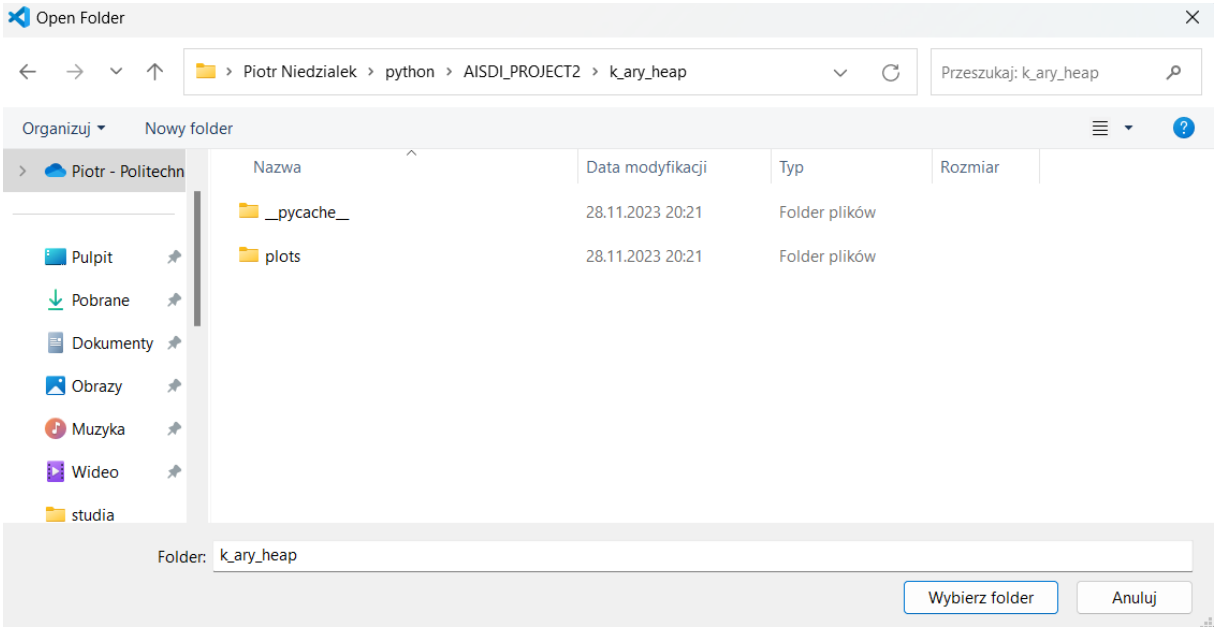
Projekt składa się z:

- Pliku main.py, który zawiera główny program wywołujący funkcje z plików i mierzący czas (Paweł Sarmacki)
- Pliku heap.py, który zawiera implementację kopca – klasę heap i metody służące do wstawiania liczb i usuwanie szczytu kopca (Piotr Niedziałek)
- Pliku draw_plots.py, który rysuje wykresy porównujące czas działania dla różnych rodzajów kopców i różnej ilości operacji wstawiania i usuwania szczytu (Paweł Sarnacki)
- Pliku draw_graphs.py, który rysuje reprezentacje kopców (Paweł Sarnacki)
- Katalogu plots, który zawiera wygenerowane wykresy

3. Uruchomienie projektu z środowiska Visual Studio Code

3.1. Otworzenie folderu, wybór ścieżki i kliknięcie wybierz folder

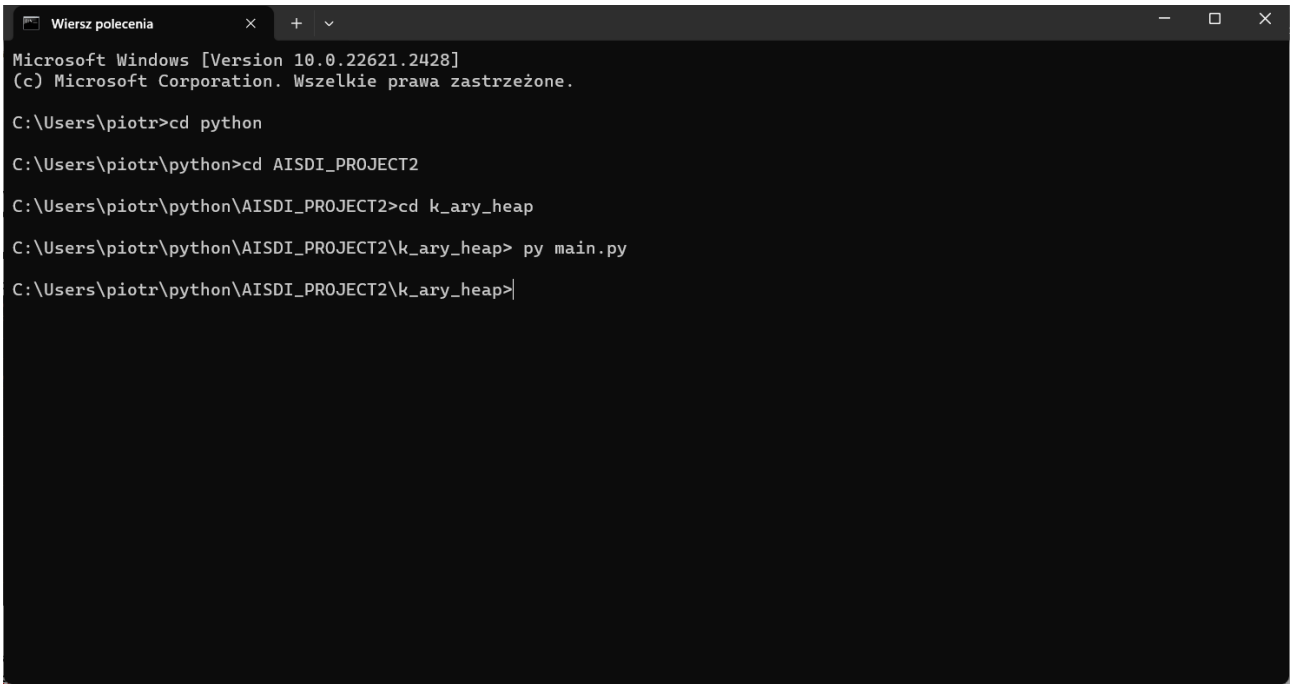




3.2. Wybranie terminalu bash, wpisanie „py main.py”, lub kliknięcie F5, wyświetlone i zapisane zostaną wykresy i grafy

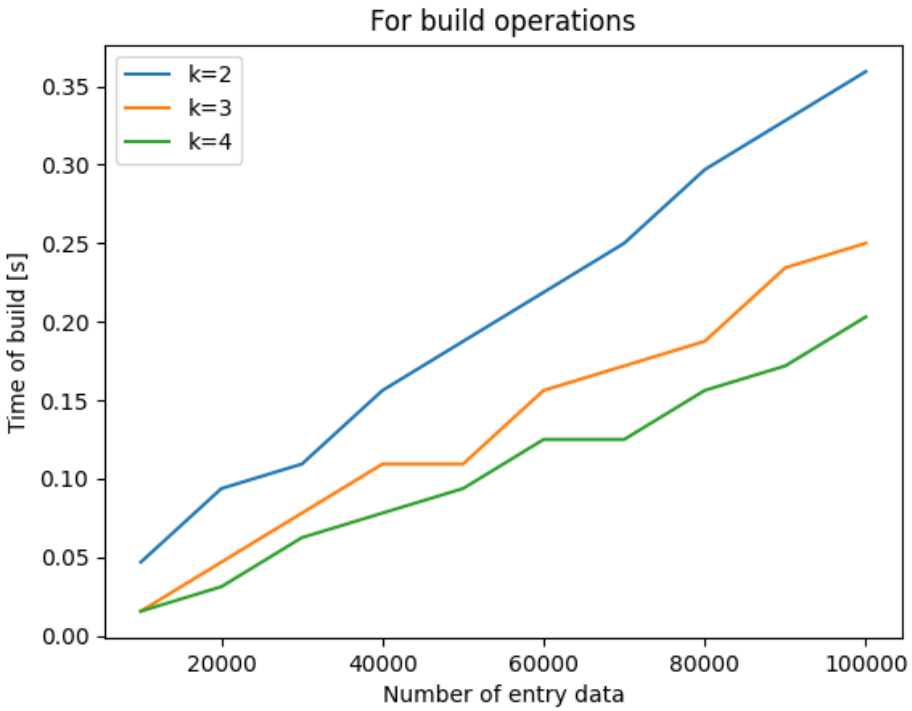
4. Uruchomienie projektu bez środowiska Visual Studio Code

Przejście do odpowiedniego folderu z plikami projektu i wpisanie w cmd **„py main.py”**

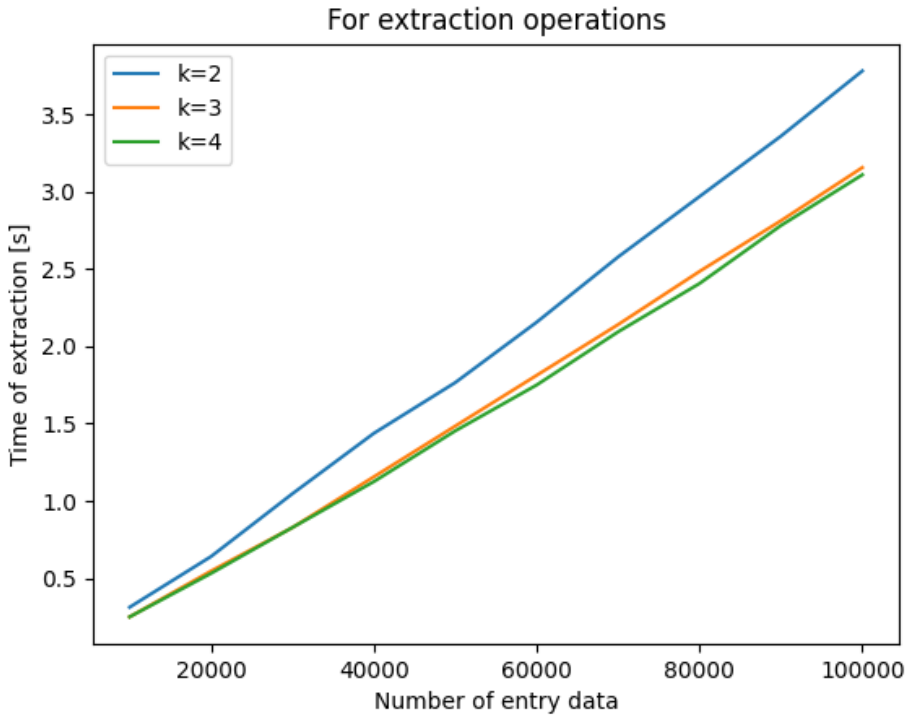


5. Otrzymane wyniki

Czas budowania kopców 2-arnych, 3-arnych i 4-arnych dla kolejnych wartości ‘n’, czyli ilości elementów:

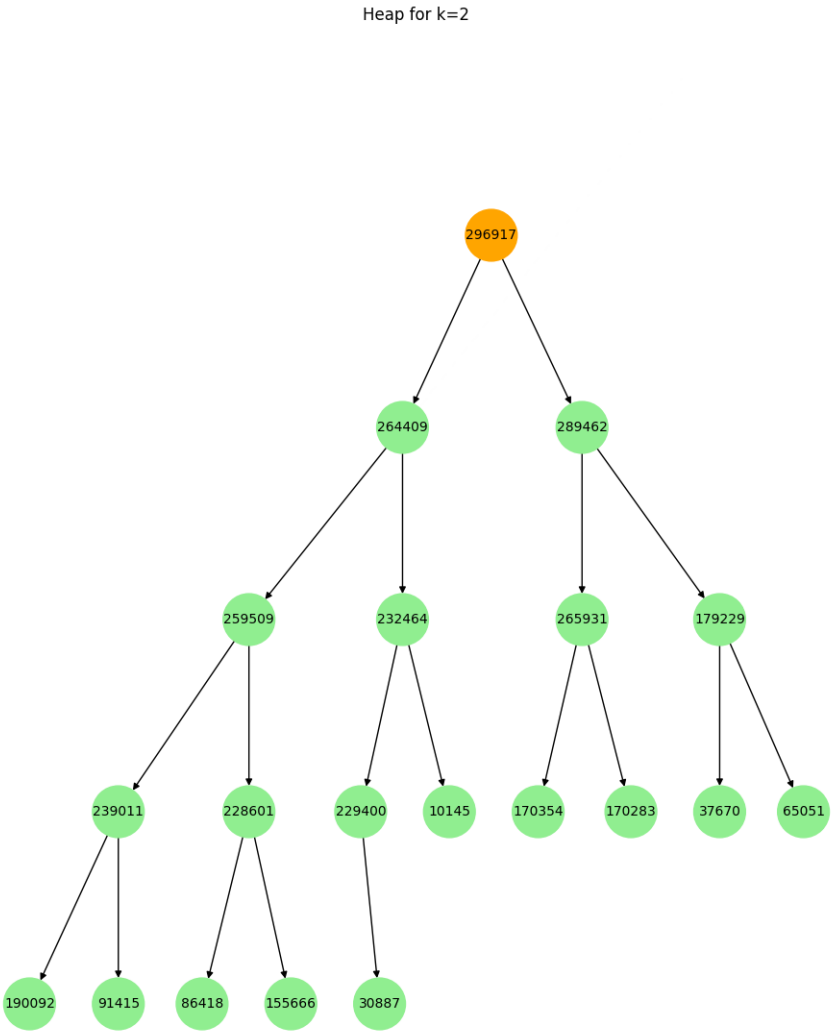


Czas usuwania szczytów kopców dla kolejnych wartości ‘n’:

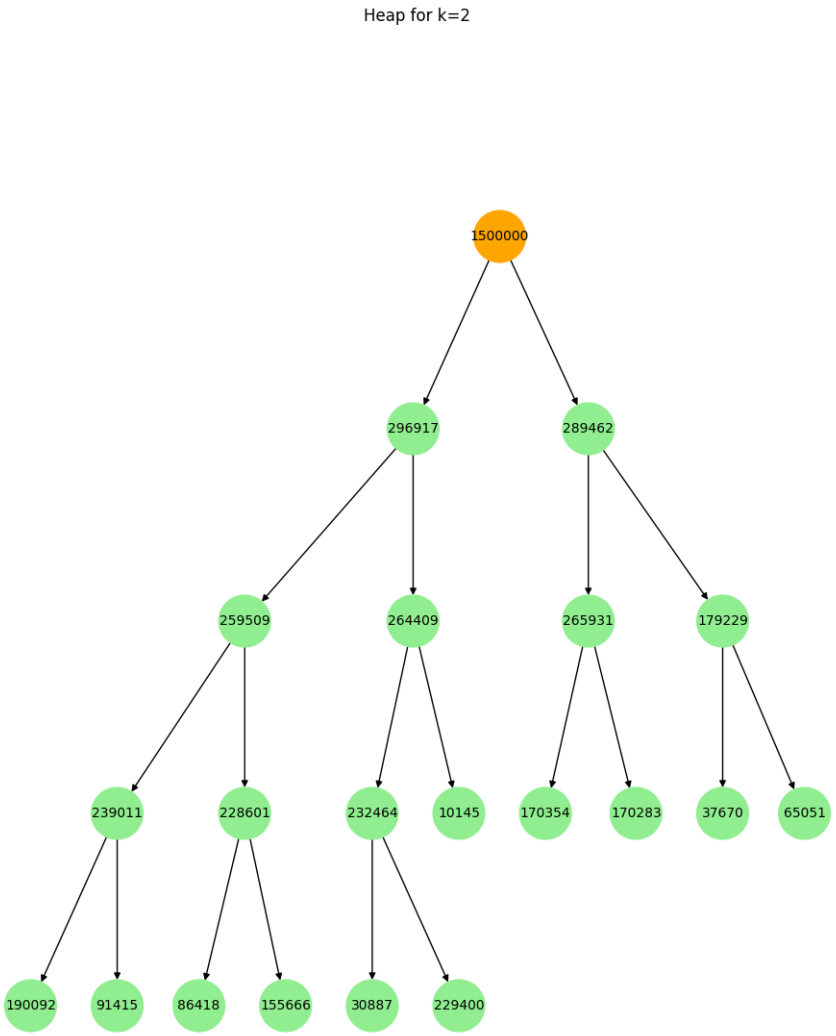


Przykłady wyświetlania kopców:

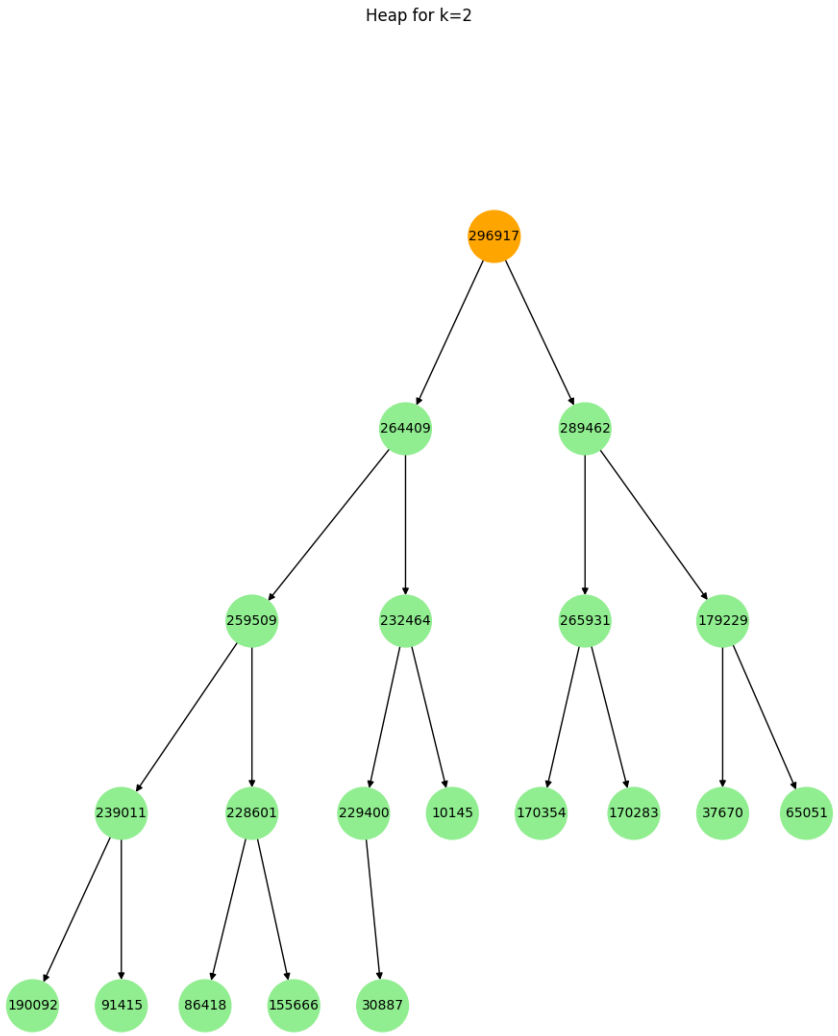
- 2-arny



Narysowany został również graf po dodaniu wielkiego elementu:

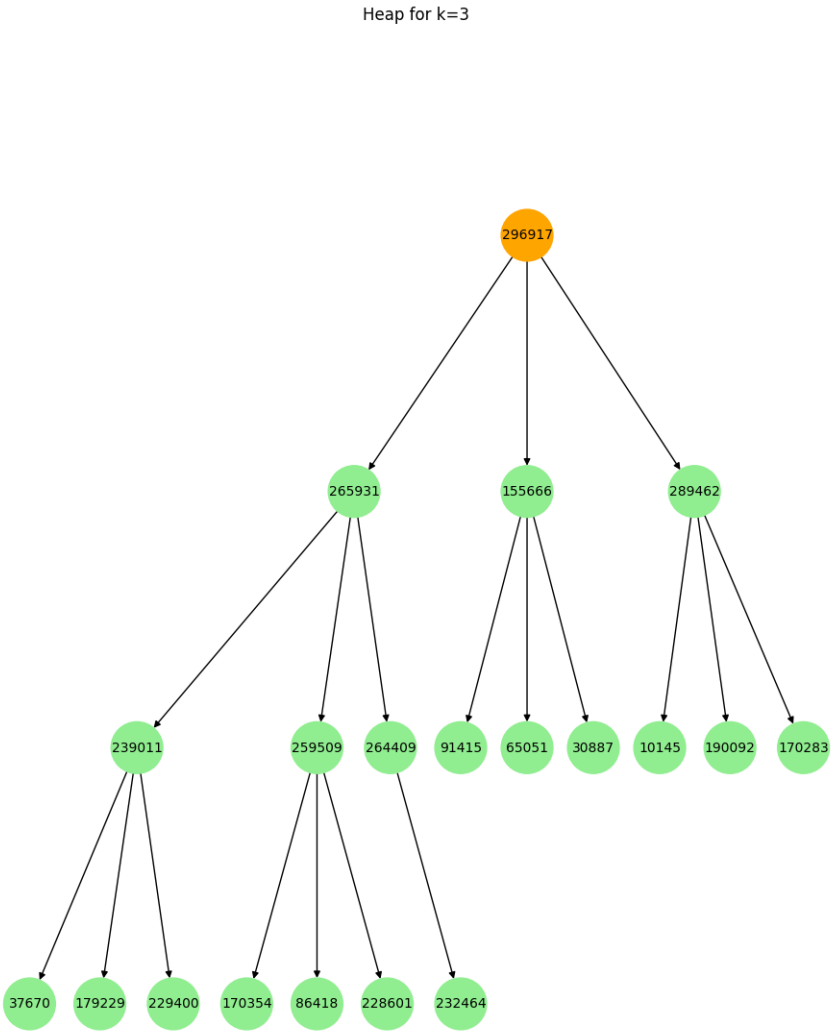


Kopiec został poprawnie przebudowany. Sprawdzono również usuwanie elementu z góry:

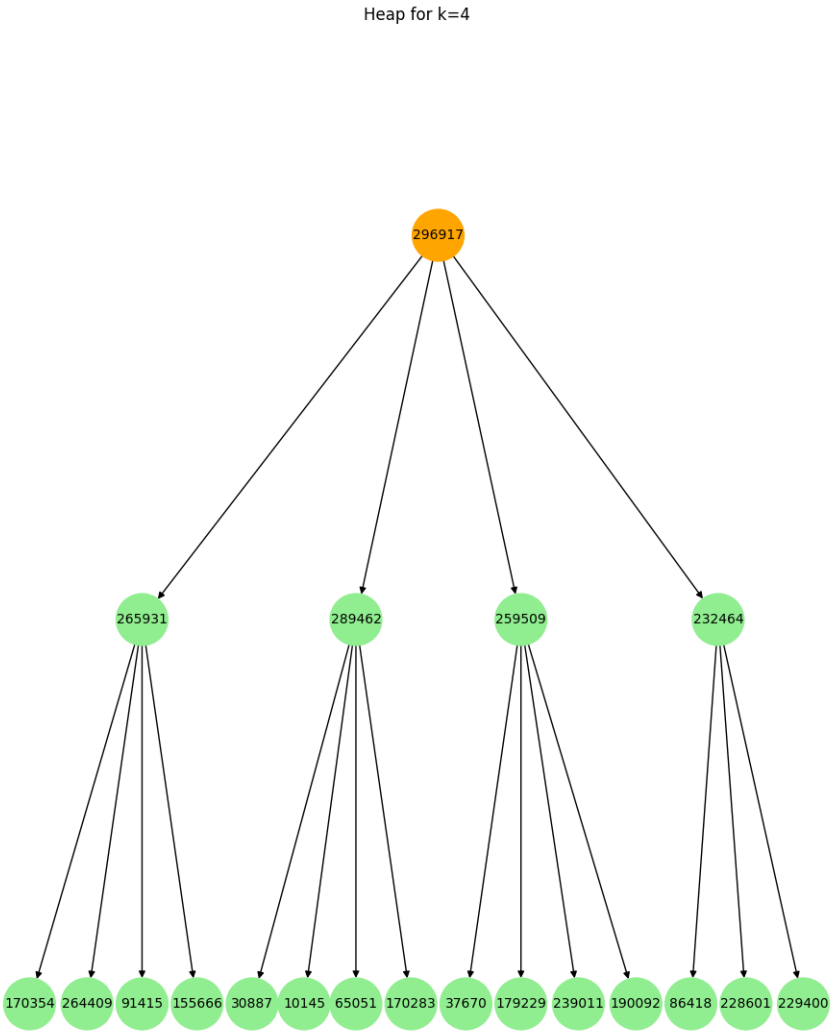


Kopiec powrócił do pierwotnej postaci – wstawianie i usuwanie odbyło się poprawnie.

- 3-arny


Heap for k=3
296917
265931
155666
289462
239011
259509
264409
91415
65051
30887
10145
190092
170283
37670
179229
229400
170354
86418
228601
232464

- 4-arny



6. Wnioski

- Implementacja budowania, dodawania elementów i usuwania szczytu kopca została zaimplementowana poprawnie. Widoczne jest to na narysowanych grafach. Każdy z kopców na grafach został poprawnie stworzony. Sprawdzenie dodania elementu i usunięcia szczytu przyniosło pozytywny rezultat.

- Czas budowania rośnie zgodnie z notacją O(n). Widoczne są zachwiania liniowości, które mają związek z brakiem kontroli nad innymi procesami odbywającymi się w tle.

- Czas usuwania odbywa się zgodnie z notacją O($klog_k n$), gdzie k to k-arny kopiec.

- Kopiec 2-arny potrzebuje więcej czasu przy budowaniu jak i usuwaniu niż pozostałe dwa rodzaje kopców.

- Przy budowaniu kopiec 4-arny ma przewagę w czasie wykonywania operacji nad pozostałymi; przy usuwaniu różnica między 3-arnym a 4-arnym jest minimalna

- Można podsumować, że wraz ze wzrostem k kopiec k-arny cechuje się mniejszym czasem budowania i mniejszym czasem usuwania szczytu