AISDI zadanie 4 – Kopce

Mikołaj Rożek, Aleksandra Szymańska

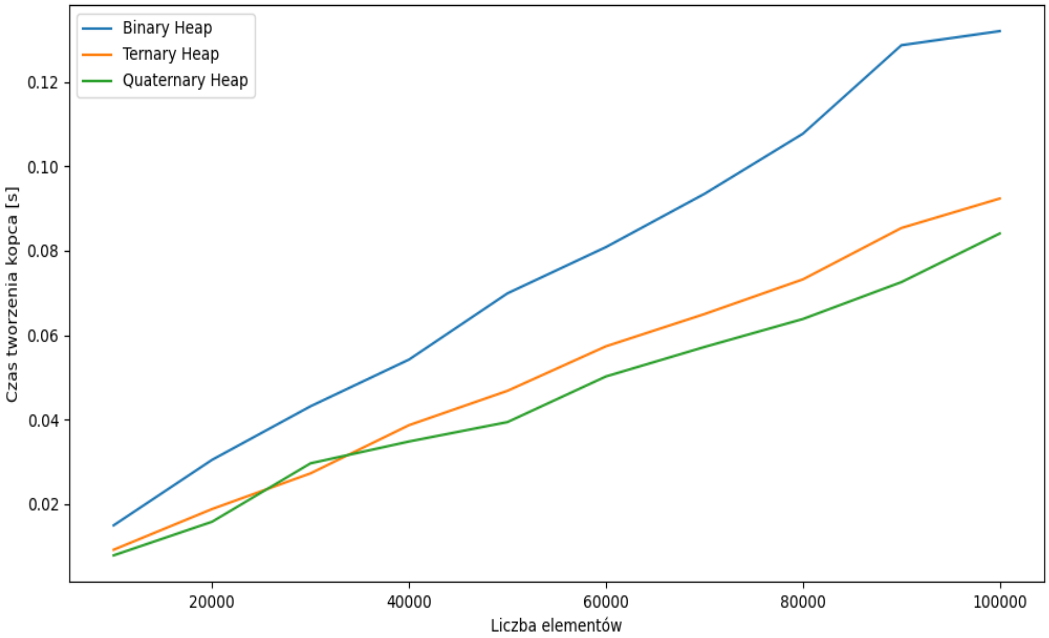
26. 04. 2023 r.

1.Opisy struktury kopca

Kopiec maksymalny n-narny to ma strukturę pełnego drzewa n-narnego, w którym rodzic jest zawsze większy od jego n-dzieci i każdy poziom oprócz ostatniego jest pełny. Ostatni poziom zapełniany jest od lewej strony. W odróżnieniu od drzew może być zaimplementowany w liście. Dziecko węzła o pozycji pos ma pozycję nnarność * pos + i, gdzie i jest numerem dziecka (<=nnarność). Rodzic dziecka o pozycji pos ma natomiast pozycję (pos – 1) / nnarność. Można zaobserwować regularność w powstawaniu drzew n-narnych, dlatego stworzyliśmy jedną klasę o parametrach arity – 'nnarność' kopca, heap_as_list – liście węzłów reprezentujących kopiec oraz current_position – pozycja ostatniego elementu (równoznaczne z długością listy reprezentującej kopiec pomniejszonej o 1). Nasza klasa posiada metody push(value) – dodającą wartość z argumentu do kopca, pop_root() – usuwającą największą wartość kopca, print_uni() – wypisującą zdeformowany, ale przedstawiający relacje węzłów, obraz kopca. Zaimplementowaliśmy też metody pomocnicze change_elements(position_1, position_2) – zamieniającą elementy na pozycjach podanych w argumentach oraz find_max_child(pos) – znajdującą numer dziecka (jeśli istnieje) o wartości większej niż rodzica na pozycji podanej w argumencie.

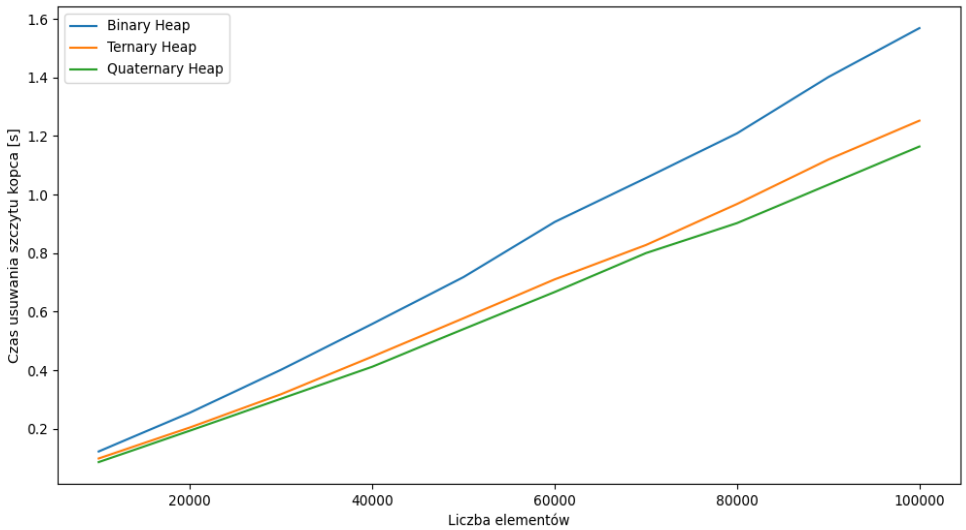
2.1 Porównanie szybkości wstawiania wartości do kopca

Wstawianie do kopca odbywa się tak, że najpierw sprawdzane jest, czy w kopcu znajduje się jakaś wartość. Jeśli nie, to jest dodawana i aktualna pozycja jest ustawiana na 0. Jeśli tak, to wartość jest dodawana na koniec listy, inkrementowana jest wartość current_position i sprawdzany oraz korygowany jest warunek, który mówi, że nowowstawiony węzeł ma być mniejszy od swojego rodzica. Jeśli nie to jest z nim zamieniany, aż warunek będzie spełniony.



2.2 Porównanie szybkości usuwania szczytu kopca

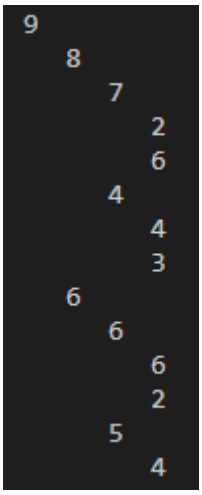
Usuwanie szczytu kopca polega na przypisaniu do szczytu kopca ostatniej wartości z listy kopca, usunięciu ostatniego elementu z listy, zdekrementowaniu current_position oraz naprawieniu warunku mówiącego o tym, że na rodzic ma być największy od dziecka. W tym celu wśród dzieci węzła szukany jest większy od rodzica. Jeśli taki istnieje, to jest on zamieniany z rodzicem i warunek sprawdzany jest dalej wgłąb, jeśli nie to znaczy, że warunek jest już spełniony.

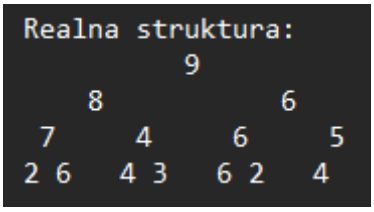


2.3 Przedstawienie wyświetlania kopców w konsoli

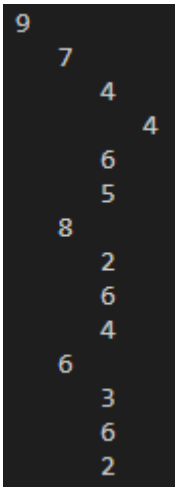
Print_uni() po lewej wyświetla większe węzły (rodziców węzłów znajdujących się po prawej i niżej). W jednej kolumnie wyświetlane są węzły znajdujące się w kopcu na jednakowej głębokości. Dodatkowo wyżej znajdują się węzły, które na danym poziomie są bardziej na lewo. Oznacza to, że szczyt kopca znajduje się najwyżej - w pierwszej kolumnie, a jego dzieci są w kolejnej kolumnie, zaczynając najwyżej od tych najbardziej na lewo.

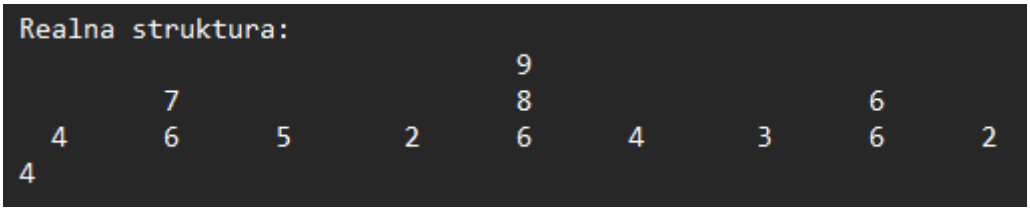
Dla kopca 2-narnego:





Dla kopca 3-narnego:





Dla kopca 4-narnego:

```
9
   8
      6
      5
      2
      7
   6
      4
      3
      6
      2
   6
      4
   4
```

```
Realna struktura:
                  9
   8        6        6        4
6 5 2 7  4 3 6 2   4
```

3. Wnioski

Na podstawie wyników przedstawionych na wykresach, można wysnuć następujące wnioski dotyczące wydajności różnych typów kopców:

Poczwórny kopiec (quaternary heap) okazał się najszybszy podczas tworzenia kopca, co jest zgodne z oczekiwaniami, gdyż ma mniejszą wysokość w porównaniu z binarnym i ternarym. Potrójny kopiec (ternary heap) osiągnął lepsze wyniki niż klasyczny kopiec binarny (binary heap), ale gorsze niż poczwórny kopiec. Wynika to z faktu, że potrójny kopiec ma mniejszą wysokość niż binarny, ale większą niż poczwórny.

Podobnie jak w przypadku tworzenia kopca, poczwórny kopiec okazał się najszybszy podczas usuwania szczytu kopca. Potrójny kopiec uzyskał wyniki lepsze niż binarny, ale gorsze niż poczwórny. Również w tym przypadku mniejsza wysokość kopca poczwórnego wpływa na jego szybsze usuwanie szczytu.

Wnioski z obu wykresów wskazują, że poczwórny kopiec jest ogólnie najszybszy, zarówno podczas tworzenia, jak i podczas usuwania szczytu. Potrójny kopiec ma średnią wydajność, a binarny jest najwolniejszy. To sugeruje, że poczwórny kopiec może być preferowanym wyborem pod względem wydajności dla dużych zbiorów danych.