Algorytmy i Struktury Danych
Laboratorium nr 4- Kopce

Autorzy:

Dominik Wiącek

Ivan Ziubanav

1. Omówienie struktury oraz złożoności czasowych związanych z nią operacji

a) Czym jest kopiec

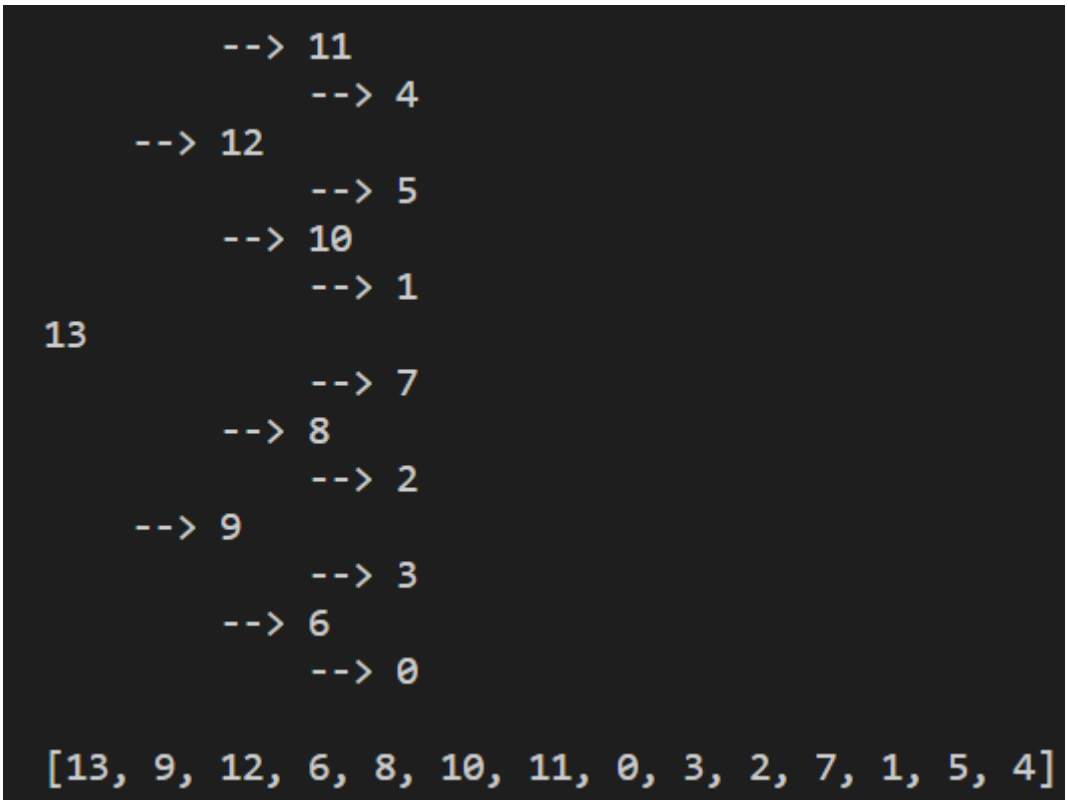
Kopiec n-arny- struktura danych bardzo podobna do drzew. Składa się z węzłów, gdzie każdy węzeł ma maksymalnie n dzieci. Co ważne, w przeciwieństwie do drzew, wszystkie dzieci posiadają tą samą zależność klucza względem rodzica, tj. albo każde dziecko ma klucz nie większy od rodzica (max heap) albo nie mniejszy (min heap). Ta własność sprawia, że kopiec jest bardzo optymalną strukturą pod tworzenie kolejki priorytetowej, ponieważ zawsze element o największym priorytecie jest na szczycie kopca (można bardzo szybko go wyekstraktować z kopca, zaś w jego miejsce automatycznie wejdzie element o następnym największym priorytecie). Kopiec też zawsze musi mieć dany poziom w pełni zapełniony zanim można wstawiać elementy na poziomie niższym, zaś poziom zapełnia się zawsze od lewej. Dzięki tym własnościom można bardzo wygodnie implementować kopiec w tablicy, a rodzica/dziecko zawsze można znaleźć w oparciu o indeks dziecka/rodzica.

b) Złożoności czasowe

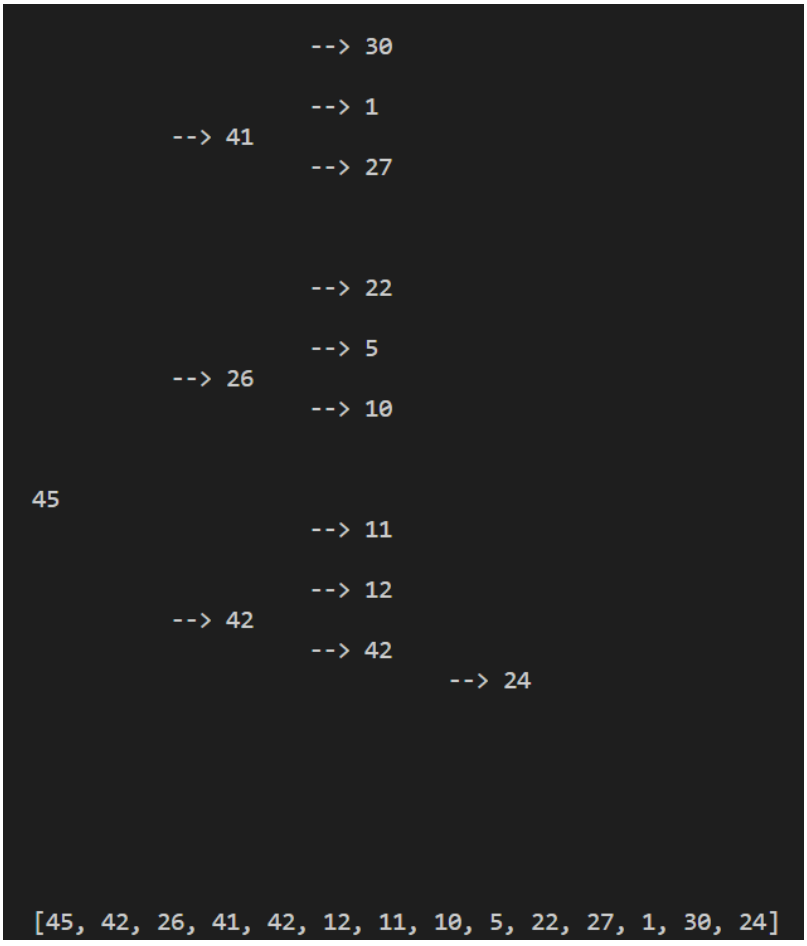
Kopca można używać do sortowania (algorytm Heap Sort). Polega on na stworzeniu kopca w oparciu o tablicę danych wejściowych (jeżeli chcemy posortować rosnąco zastosujemy max heap, jeżeli malejąco: min heap). Dzięki właściwościom kopca na jego szczycie będzie znajdował się element ekstremalny. Należy go zamienić z elementem ostatnim w tablicy (na samym końcu kopca) i zmniejszyć długość kopca o 1, dzięki czemu ten element (który jest na swojej posortowanej pozycji) nie będzie dalej rozważany. Operację należy powtarzać, aż nie skończą się elementy w kopcu. Wtedy tablica będzie posortowana. Złożoność czasowa heap sortu w każdym przypadku jest O(nlogn), ponieważ operacja usunięcia szczytu kopca ma złożoność O(logn) (musimy po usunięciu węzła przejść przez całą wysokość drzewa znajdując nowy ekstremalny element), a operację tą musimy wykonać tyle razy ile jest elementów w kopcu. Operacja dodania węzła do kopca ma, tak jak operacja usuwania szczytu złożoność O(logn), ponieważ dzieją się podobne czynności jak przy usuwaniu, z tym że najpierw dodajemy węzeł na koniec tablicy (kopca) i potem musimy przejść potencjalnie przez całą wysokość kopca, aby wstawić go na odpowiednią pozycję.

2. Przykład wypisania kopców na ekran

a) 2-arny


--> 11
--> 4
--> 12
--> 5
--> 10
--> 1
13
--> 7
--> 8
--> 2
--> 9
--> 3
--> 6
--> 0
[13, 9, 12, 6, 8, 10, 11, 0, 3, 2, 7, 1, 5, 4]

b) 3-arny


--> 30
--> 1
--> 41
--> 27
--> 22
--> 5
--> 26
--> 10
45
--> 11
--> 12
--> 42
--> 42
--> 24
[45, 42, 26, 41, 42, 12, 11, 10, 5, 22, 27, 1, 30, 24]

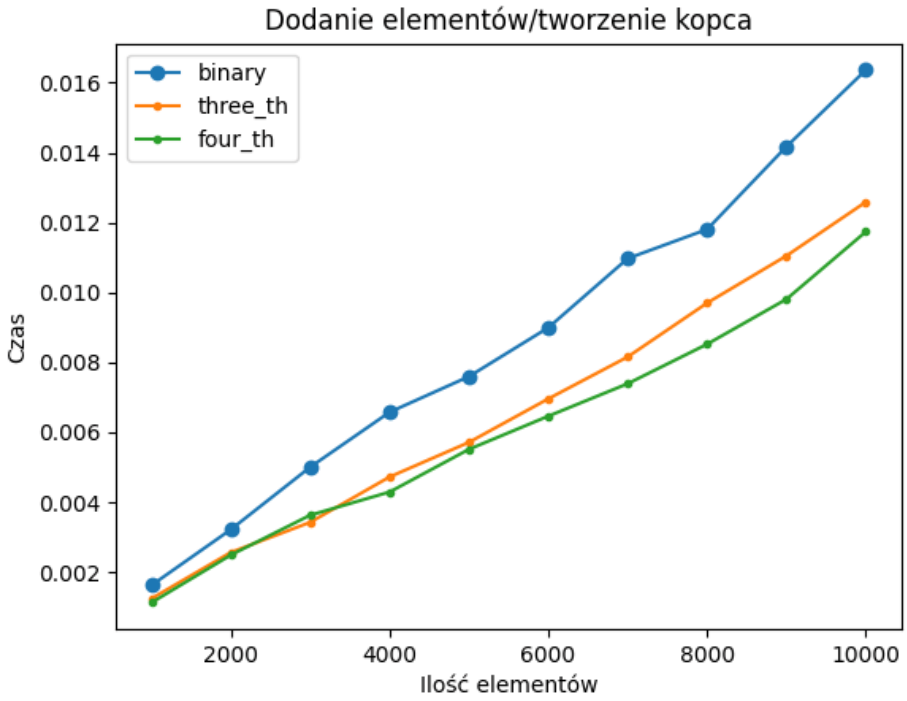
c) 4-arny

```
    --> 3
    --> 12
        --> 2
13
        --> 10
        --> 9
    --> 11
        --> 8
        --> 1
        --> 6
        --> 5
    --> 7
        --> 4
        --> 0

[13, 7, 11, 12, 3, 0, 4, 5, 6, 1, 8, 9, 10, 2]
```

3. Prezentacja danych z pomiarów wydajności kopców

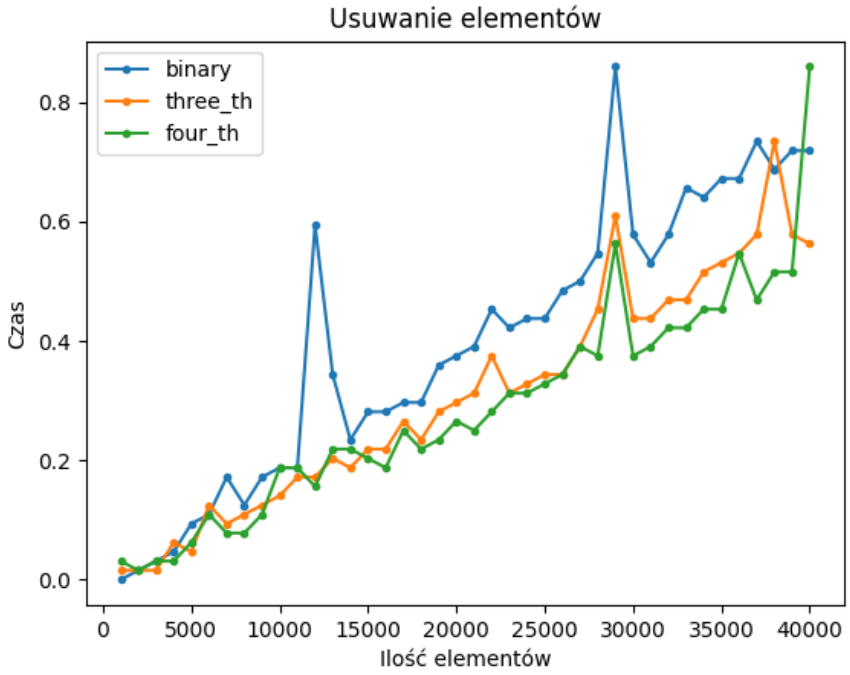
a) Operacja dodawania elementu do kopca



W przybliżeniu można przyjąć, że są to złożoności logarytmiczne dla każdego kopca, co się zgadza z rzeczywistością. Im większe n dla n-arnego kopca tym czas jest mniejszy,

ponieważ wtedy jest on bardziej rozpięty, więc dla tych samych elementów ma on mniejszą wysokość, więc po dodaniu musi w najgorszym wypadku mniej razy zamieniać się z elementami położonymi wyżej.

b) Operacja usunięcia szczytu kopca



W przybliżeniu czas jest logarytmiczny, tak jak w przypadku operacji dodania elementu. Wszystkie charakterystyki oraz powody są takie same jak w przypadku tamtej operacji.