Algorytmy i struktury danych - kolokwium semestr zimowy 2019/2020

Imię i nazwisko: _____

Numer indeksu: | | | | | | |

B

Zadanie:	1	2	3	4	5	6	7	8	9	10	**Łącznie**
Punktacja:	1	2	1	1	2	2	2	1	2	1	15
Wynik:											

1. (1 punkt) Klasa `std::vector` z biblioteki standardowej definiuje metody `size` i `capacity`. Proszę wyjaśnić znaczenie zwracanych przez nie wartości. Jaka jest złożoność czasowa wstawiania nowego elementu na pozycji środkowej `std::vector`? Jak relacja między `size` i `capacity` wpływa na złożoność czasową takiej operacji?

   `size` - liczba elementów, `capacity` - pojemność,
   wstawianie - $O(n/2) \approx O(n)$, jeżeli `size == capacity` to konieczna realokacja $O(n)$

2. (2 punkty) *Palindromem* określamy wyrażenie brzmiące tak samo, gdy jest czytane od lewej do prawej i od prawej do lewej (np. „*A kilku tu klika*”). Proszę zaproponować algorytm (wystarczy opis słowny), który dla listy jednokierunkowej przechowującej kolejne znaki wyrażenia (bez znaków białych) sprawdzi, czy wyrażenie jest palindromem. Algorytm powinien działać w czasie liniowym $O(n)$ i korzystać ze stałej ilości pamięci pomocniczej $O(1)$.

   Zarys algorytmu:

   1. Znajdź środek listy (dwa wskaźniki: szybki/wolny).
   2. Odwróć drugą połowę listy (odwracanie w miejscu).
   3. Sprawdź, czy pierwsza i druga połowa są identyczne.
   4. Odwróc drugą połowę listy ponownie, aby uzyskać listę początkową.

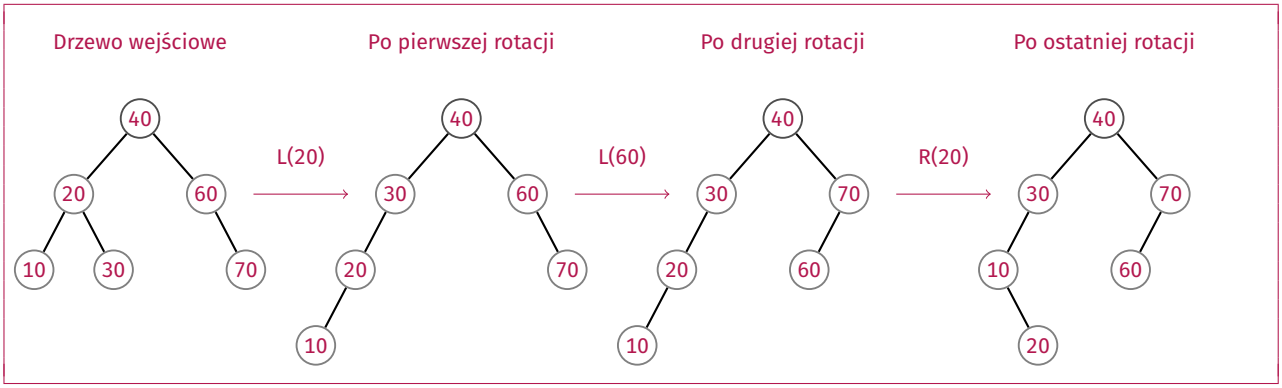
3. (1 punkt) Klasa `Stack` implementuje funkcjonalność stosu z metodami `top`, `push`, `pop`, `empty`, `size`. Obiekt `s1` klasy `Stack` powstał z pustego stosu przez wstawienie w kolejności elementów `{13, 17, 19, 23, 29}`. Proszę zapisać implementację (C++/pseudokod) odwrócenia kolejności elementów w obiekcie `s1` przy wykorzystaniu dwóch pustych stosów pomocniczych `s2` i `s3`. Jaką wartość zwróci wywołanie metody `s1.top()` przed i po odwróceniu stosu?

```
while (!s1.empty()) {
    s2.push(s1.top());
    s1.pop();
}
while (!s2.empty()) {
    s3.push(s2.top());
    s2.pop();
}
while (!s3.empty()) {
    s1.push(s3.top());
    s3.pop();
}
// Przed odwróceniem: s1.top() == 29, po odwróceniu: s1.top() == 13
```

4. (1 punkt) Proszę wyjaśnić, na czym polega zastosowanie wartownika (*sentinel*) w przeszukiwaniu liniowym listy i jak wpływa na liczbę wykonywanych operacji porównań.

Wartownik - specjalny element oznaczający koniec listy, zastępuje wykorzystanie **`nullptr`**. Wstawienie wartości szukanej do wartownika pozwala wykonywać jedno porównanie mniej na iterację, bo nie ma konieczności sprawdzania warunku końca listy (mamy pewność, że w najgorszym wypadku zatrzymamy się na strażniku).

5. (2 punkty) Proszę uzupełnić ciąg rotacji przekształcających drzewo wejściowe w wyjściowe. Dla każdego kroku proszę naszkicować stan drzewa, określić kierunek przeprowadzonej rotacji i wskazać wierzchołek będący korzeniem rotacji.



6. (2 punkty) Proszę wyjaśnić zasadę działania algorytmu sortowania przez wybieranie (*selection sort*) (opis lub pseudokod). Jaka jest złożoność czasowa działania algorytmu (średnia, najlepsza, najgorsza) i pamięciowa? Czy jest to algorytm stabilny? Proszę zilustrować działanie algorytmu dla przedstawionej tablicy.

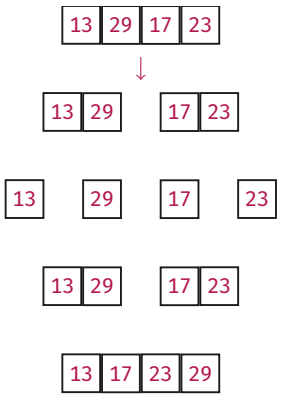
Powiększa obszar posortowany dobierając na koniec najmniejszy element z obszaru nieposortowanego.

- Złożoność czasowa: $O(n^2)$, $O(n^2)$, $O(n^2)$
- Złożoność pamięciowa: $O(1)$
- Stabilność: nie

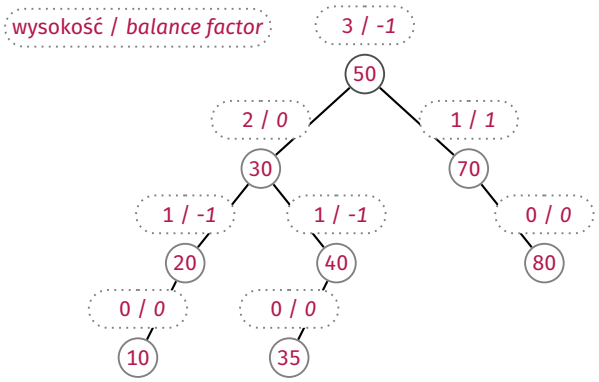
7. (2 punkty) Proszę wyjaśnić zasadę działania algorytmu sortowania przez scalanie (*merge sort*) (opis lub pseudokod). Jaka jest złożoność czasowa działania algorytmu (średnia, najlepsza, najgorsza) i pamięciowa? Czy jest to algorytm stabilny? Proszę zilustrować działanie algorytmu dla przedstawionej tablicy.

Sortuje zakres dzieląc go rekursywnie na mniejsze fragmenty i następnie scalając posortowane części.

- Złożoność czasowa: $O(n \log n)$, $O(n \log n)$, $O(n \log n)$
- Złożoność pamięciowa: $O(n)$
- Stabilność: tak

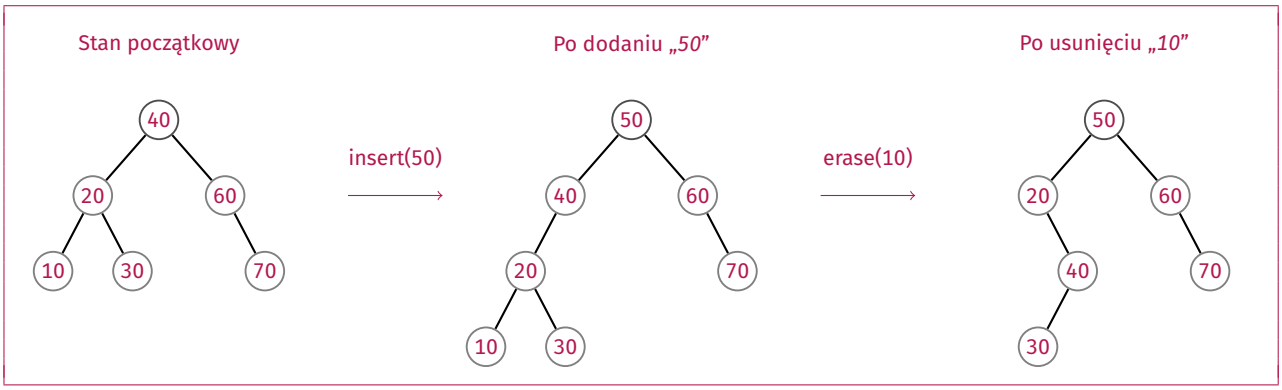


8. (1 punkt) Do każdego z węzłów drzewa proszę dopisać wysokość w danym węźle i współczynnik wyważenia (*balance factor*). Czy przedstawione drzewo jest drzewem AVL?

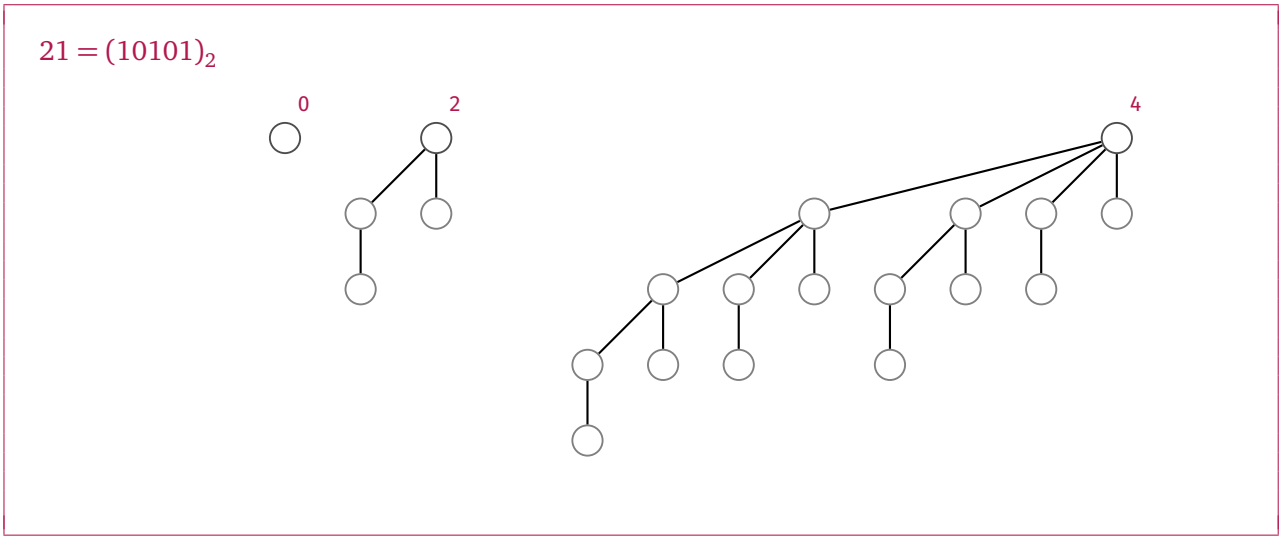


Jest AVL.

9. (2 punkty) Dla zadanego drzewa *splay* proszę naszkicować stany drzewa po sukcesywnie wykonywanych operacjach dodawania elementu o wartości 50 i usuwania elementu o wartości 10.



10. (1 punkt) Proszę naszkicować kopiec dwumianowy zawierający 21 elementów. Na rysunku proszę oznaczyć rzędy poszczególnych drzew dwumianowych.



Miejsce na notatki: