Projekt

Algorytmy i strukury danych I

Można zaproponować własny projekt lub wybrać zadanie z poniższej listy. Wybór zadania należy wcześniej zgłosić. Co najwyżej trzy osoby mogą rozwiązać to samo zadanie (oprócz zagadnienia 1.). Należy wykorzystać własne implementacje algorytmów i struktur danych omawianych na zajęciach. Programy powinny odczytywać dane wejściowe ze standardowego wejścia i zapisywać dane wyjściowe na standardowe wyjście, chyba że dla danego zadania wyraźnie napisano inaczej.

1. Własny projekt (20 - 25 pkt)

Można zaproponować własny projekt.
Powinien on korzystać z algorytmów i struktur danych omawianych na zajęciach.

2. Asymmetric Numeral Systems (25 pkt)

Napisać narzędzie do bezstratnej kompresji i dekompresji oparte o metodę *Asymmetric Numeral Systems* w wariancie stablicowanym (tANS).

Przydatne materiały:

- Asymmetric numeral systems II
- Prezentacja dr. Jarosława Dudy nt. ANS (`Materiały/Projekt/ANSsem_mat.pdf`)
- Praca magisterska w języku polskim dot. implementacji algorytmu tANS na układy FPGA (`Materiały/Projekt/tANS.pdf`)
- www.ezcodesample.com
- Asymmetric numeral systems

3. Punkty w przestrzeni (25 pkt)

Proszę opracować i zaimplementować algorytm, który przechowuje bazę współrzędnych punktów w przestrzeni d-wymiarowej i umożliwia *dodanie kolejnego punktu do bazy* oraz *znalezienie najbliższego punktu* spośród znajdujących się w bazie dla wskazanego punktu (wg metryki Euklidesa) z **logarytmiczną złożonością czasową**.

Napisać program, który znajduje najbliższy punkt metodą *brute-force*. Przetestować programy sprawdzając czy dają identyczne wyniki. Empirycznie sprawdzić złożoność czasową programów.

4. Słownik T9 (20 pkt)

Napisać program używający słownika T9 (dla języka polskiego lub angielskiego) do znajdywania wyrazów wraz z graficzną nakładką (imitującą klawiaturę numeryczną).

5. Wizualizacja operacji na drzewach (25 pkt)

Wizualizacja operacji na drzewach BST oraz AVL. Operacje do wykonania zadawane są z klawiatury.

6. Słownik w oparciu o drzewo AVL (20 pkt)

> ***Drzewo AVL*** *(od twórców G.* ***A****delson-*** V****elsky and E.*** L****andis) to drzewo wyszukiwań binarnych zrównoważone po wysokościach. Wysokości lewego i prawego poddrzewa każdego wierzchołka różnią się co najwyżej o 1.*
> *Szczegółowe informacje można znaleźć na stronie.*

Zaimplementuj słownik przy pomocy drzewa AVL oraz operacje charakterystyczne dla tej struktury:

- `insert(x)` - wstawia klucz `x` do drzewa
- `rotateLeft()` i `rotateRight()` - funkcje pomocnicze, wykonują rotację w lewo/prawo na węźle `x`. Jest to lokalna operacje na drzewie, która zachowuje uporządkowanie *inorder*.
- `erase(n)` - usuwa węzeł `n` z drzewa
- `search(x)` - zwraca klucz `x` w drzewie (wskaźnik do węzła), albo `nullptr` jeśli tego klucza nie ma
- `size()` - zwraca liczbę kluczy w drzewie
- `depth()` - zwraca wysokość drzewa
- `join(S1, S2)` - łączy dwa drzewa `S1` i `S2` w jedno przy założeniu, że wszystkie klucze w `S1` są mniejsze niż w `S2`
- `split(x)` - dzieli drzewo na dwa drzewa: pierwszy złożony z elementów mniejszych bądź równych `x` i drugi złożonych z elementów większych od `x`

Elementy słownika są liniowo uporządkowane.

Napisać program, który wczyta słowa z pliku words.txt (1.8MB) i wstawi je do słownika. Napisać graficzny interfejs użytkownika, który sprawdzi czy dane słowo jest w słowniku.

Pytania do zadania

1. Podać przykład konkretnego poprawnego drzewa AVL i elementu, którego dodanie wymusi pojedynczą rotację. Przedstawić proces.
2. Podać przykład konkretnego poprawnego drzewa AVL i elementu, którego dodanie wymusi podwójną rotację. Przedstawić proces.

Wskazówki

> *Aby wyłączać synchronizację strumieni wejścia/wyjścia (`cin`, `cout`) z `stdio` należy na samym początku programu użyć* `ios_base::sync_with_stdio(false)`. *W przeciwnym razie programy mogą działać wolniej.*

7. Prostokąt (25 pkt)

Zadanie

Dany jest zbiór n punktów na płaszczyźnie o współrzędnych całkowitych. Znaleźć maksymalną liczbę punktów jaką może obejmować prostokąt o szerokości w i wysokości h.

Wejście

W pierwszym wierszu zapisano trzy dodatnie liczby całkowite oddzielone pojedynczym odstępem

- w $(1 \leq w \leq 10000)$ - szerokość prostokąta

- $h\,(1 \leq h \leq 10000)$ - wysokość prostokąta
- $n\,(1 \leq n \leq 15000)$ - liczba punktów

W kolejnych n wierszach zapisane są współrzędne punktów. Każdy z tych wierszy zawiera dwie liczby całkowite x i y ($-30000 \leq x, y \leq 30000$). **Rozwiązanie ma mieć złożoność czasową** $O(n \cdot \log n)$, także w przypadku pesymistycznym.

Napisać program, który rozwiązuje problem metodą *brute-force*. Przetestować programy sprawdzając czy dają identyczne wyniki. Empirycznie sprawdzić złożoność czasową programów.

Wyjście

Wypisać maksymalną liczbę punktów, którą może obejmować prostokąt.

8. Kopiec Fibonacciego (20 pkt)

Zaimplementować kopiec Fibonacciego z uwzględnieniem następujących operacji:

- `push` - wstawia element do kopca
- `pop` - usuwa i zwraca wartość najmniejszego elementu
- `top` - zwraca wartość najmniejszego elementu
- `size` - zwraca liczbę elementów na stosie
- `decrease-key` - zmniejsza wartość elementu
- `remove` - usuwa elementu

Szczegółowe informacje można znaleźć w pliku fibonacci-heap.pdf.

Pytania do zadania: Proszę przedstawić proces dodawania elementów 1, 2, . . . , 10 a następnie operacji `remove` (`delete-min`).

9. Lempel–Ziv–Welch (20 pkt)

Napisać narzędzie do bezstratnej kompresji oparte o metodę Lempel–Ziv–Welch.

10. Triangulacja Delaunay (25 pkt)

Napisać program, który dla zbioru punktów na płaszczyźnie, np. w kwadracie $(0, 1)^2$ lub na torusie T^2, generuje triangulację Delaunay. Napisać również wersję, która generuje losowy zbiór punktów **jednorodnie** rozrzuconych na sferze S^2, i tworzy dla niego triangulację Delaunay *na tej sferze*.

Program powinien wizualizować triangulację zapisując ją do pliku graficznego w formacie wektorowym np. `PostScript`, `PDF` lub `SVG`. Można wykorzystać np. bibliotekę `cairo`.

Andrzej Görlich