Algorytmy wyszukiwania i funkcje haszujące

Maciej Hojda

Uwaga: Słowa „dany", „zadany", „podany", „wybrany" itd. w kontekście parametrów (zmiennych) oznacza parametr zadany przez użytkownika (a nie na stałe, przez programistę), a implementacja wykorzystująca taki parametr powinna obsługiwać jego różne wartości.

Zadanie nr 0 – przygotowanie struktury danych

- Utwórz strukturę (może być klasa) do przechowywania opisów floty robotów mobilnych wykorzystywanych w zadaniach eksploracji i inspekcji. Pola składowe struktury to: `identyfikator`, `typ`, `masa`, `zasięg`, `rozdzielczość` (kamery).
- Zaimplementuj funkcję generującą opis pojedynczego robota (jeden obiekt) o parametrach losowanych jak następuje: `identyfikator` – ciąg znaków alfanumerycznych o długości N, `typ` – `,,AUV''`, `''AFV''`, `,,AGV''`, `masa` – od 50 do 2000 [dag], `zasięg` – od 1 do 1000 [km], `rozdzielczość` – od 1 do 30 [MP]. Wartości liczbowe są całkowite.
- Zaimplementuj funkcję generującą opisy M robotów o losowych parametrach. Opisy, w postaci struktur są zapisywane w M-elementowym wektorze.
- Zaimplementuj funkcję wyświetlającą wygenerowaną strukturę w postaci tabelki (jeden wiersz – jeden robot; równe odstępy między kolejnymi polami).
- Zaimplementuj funkcję zapisującą/odczytującą opisy do/z pliku.

Na tym **wektorze robotów** realizowane są kolejne zadania. **Uwaga:** nie zakładaj, że wartości parametrów są unikalne.

1 Zadanie nr 1 – wyszukiwanie liniowe

- Zaimplementuj funkcję wyszukującą jednego robota z **wektora robotów**. Poszukiwanie odbywa się względem jednego parametru (parametr wybierany przez użytkownika; szukana wartość wybierana przez użytkownika). Funkcja realizuje algorytm wyszukiwania liniowego i zwraca strukturę z pierwszym znalezionym robotem (lub `None`, jeśli robota nie znajdzie).
- Rozwiń napisaną funkcję tak, aby wyszukiwanie odbywało się po zbiorze parametrów. Zbiór i wartości są zadawane przez użytkownika w postaci struktury analogicznej do tej reprezentującej pojedynczego robota; parametry, po których wyszukiwanie się nie odbywa pozostają puste (`None`), np. `(None, ,,AUV'', 50, 10, None)`.
- Rozwiń napisaną funkcję tak, aby akceptowała też wektory wartości parametrów, np. `(None, ,,AUV'', [90, 51, 52, 53], [10, 11, 15], None)`. Zagnieźdź algorytmy wyszukiwania liniowego tak, aby parametr robota był wyszukiwany w liście dopuszczalnych wartości.

Funkcje wyszukiwania uzupełnij o tryb wizualizacji krok-po-kroku (tzn. wyświetlaj kolejne analizowane elementy **wektora robotów** i aktualnie poszukiwane wartości parametrów).

2 Zadanie nr 2 – wyszukiwanie binarne

- Zaimplementuj funkcję sortującą **wektor robotów** względem wskazanego parametru. Wykorzystaj wbudowaną funkcję `sort`.
- Zaimplementuj funkcję wyszukującą jednego robota o zadanym parametrze, przy założeniu, że wektor jest odpowiednio posortowany. Funkcja realizuje algorytm wyszukiwania binarnego i zwraca strukturę z pierwszym znalezionym robotem (lub `None`, jeśli robota nie znajdzie).
- Rozwiń napisaną funkcję tak, aby akceptowała wektor dopuszczalnych wartości parametrów (posortowany). Wyszukiwanie binarne wykorzystaj wielokrotnie.

Funkcje wyszukiwania uzupełnij o tryb wizualizacji krok-po-kroku.

3 Zadanie nr 3 – wyszukiwanie z wykorzystaniem funkcji haszującej

- Zaproponuj i zaimplementuj funkcję haszującą działającą na pojedynczej strukturze robota. Zbiór wartości funkcji to liczba całkowita z zakresu od 0 do $H-1$. Funkcja powinna uwzględniać wszystkie parametry robota. Do ograniczenia wartości wykorzystaj operację modulo.
- Zaimplementuj funkcję generującą **wektor wartości** funkcji haszującej dla zadanego **wektora robotów**. **Wektor wartości** jest generowany jak następuje:

Algorithm 1 GWW – Generacja **wektora wartości**

- wejscie: **wektor robotów**, $H >> |$`wektor robotów`$|$
1 niech **wektor wartości** będzie $H-1$ elementowym wektorem wypełnionym pustymi elementami (`None`)
2 dla $n := 0$ do $|$`wektor robotów`$| - 1$
3 oblicz wartość h funkcji haszującej na $n$tym elemencie **wektora robotów**
4 na pierwszej pustej pozycji **wektora wartości** licząc od h tej pozycji (włącznie) wstaw n; jeśli do końca **wektora wartości** brakuje wolnych pozycji, to kontynuuj od jego początku
- wyjście: **wektor wartości**

- Zaimplementuj funkcję wyszukiwania robota o zadanych wszystkich parametrach, która korzysta z **wektora wartości** i funkcji haszującej.
- Zaproponuj i zaimplementuj metodę wyszukiwania robota po 2 parametrach: `masa` i `zasięg`, dopuszczając sytuację, gdy dowolny z parametrów jest nieznany, np. `(None, 40)`. Wykorzystaj (być może zmodyfikowaną) funkcję haszującą (generacja wektora wartości może odbywać się inaczej niż w algorytmie GWW). Zwracaj listę wszystkich robotów spełniających postawione wymagania.

Funkcje wyszukiwania i generacji **wektora wartości** uzupełnij o tryb wizualizacji krok-po-kroku.

4 Zadanie nr 4 – binarne wyszukiwanie zbioru wartości

- Zaimplementuj funkcję zwracającą zbiór indeksów wszystkich robotów, których wybrany parametr ma zadaną wartość.
  Wykorzystaj procedurę poszukiwania binarnego w celu znalezienia pierwszego robota spełniającego wymaganie. Zauważ, że w wyniku działania procedury znaleziony został też przedział, w którym znajdują się wszystkie poszukiwane roboty. Dokładniej, w ostatniej iteracji algorytmu, wykonano operację $mid =$ `round`$(\frac{max+min}{2})$, gdzie mid jest zwróconym indeksem robota, zaś min i max to krańce przedział

poszukiwania w ostatniej iteracji. Znając min i max, uruchom wyszukiwanie binarne na przedziałach $[min, mid]$ i $[mid, max]$ w celu znalezienia minimalnej i maksymalnej indeksów robotów spełniających wymaganie.

- Zaimplementuj funkcję zwracającą zbiór indeksów wszystkich robotów o parametrze liczbowym z zadanego przedziału. Ponownie, wykorzystaj wyszukiwanie binarne.

5 Zadanie nr 5 – wieloatrybutowe, wielowartościowe wyszukiwanie binarne

- Zaimplementuj funkcję tworzącą, dla każdego parametru robota **wektor pomocniczy**. Każdy wektor pomocniczy, o rozmiarze takim samym jak **wektor robotów**, zawiera indeksy z wektora robotów posortowane względem danego parametru.
- Zaimplementuj funkcję wyszukiwanie binarnego realizowaną po zbiorze parametrów (każdy z dopuszczalnymi wieloma wartościami lub `None`, tak jak na końcu zadania nr 1). Wyszukiwanie wykonaj po każdym parametrze z osobna, korzystając z procedury opracowanej w poprzednim zadaniu. Następnie weź część wspólną wszystkich zbiorów indeksów (wykorzystaj funkcję `intersection`). Nie zakładaj, że **wektor robotów** jest jakkolwiek posortowany, a korzystaj z **wektorów pomocniczych**.

Funkcje wyszukiwania uzupełnij o tryb wizualizacji krok-po-kroku.

6 Zadanie nr 6 (opcjonalne) – algorytm MD5

Zaimplementuj algorytm MD5 tak, aby wyliczał skrót dla dowolnego pliku zadanego przez użytkownika.