**Lucrare de laborator**

**la disciplina**

**Structuri de Date și Algoritmi**

**Lucrare de laborator nr. 5:**

**Tema:**

Implementarea și analiza algoritmilor de sortare și de căutare în limbajul C.

**Scopul lucrării**

Scopul lucrării este de a familiariza studentul cu implementarea și analiza algoritmilor, utilizînd pentru aceasta limbajul C.

**Problema**

1. **Să se elaboreze un program ce va aloca dinamic un tablou unidimensional de numere întregi și va implementa următoarele funcții, funcțiile vor fi organizate sub forma unui meniu:**

* inițializarea tabloului cu numere aleatorii;
* afișarea tabloului;
* eliberarea memoriei tabloului;
* sortarea tabloului utilizînd una din metode ( selection sort, insertion sort, bubble sort ), se va afișa în cît timp a fost executată sortarea;
* sortarea tabloului utilizînd una din metode ( merge sort, quick sort, heap sort ), se va afișa în cît timp a fost executată sortarea.
  1. **Să se realizeze o analiză empirică a timpului necesar pentru a sorta un tablou aleator de lungimea N unde N va avea valori [100, 1000, 10000, 100000], rezultatele obținute se vor plasa în tabelul de mai jos:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **N** | **100** | **1000** | **10000** | **100000** |
| **Metoda de sortare 1 ( ex. selection sort)** |  |  |  |  |
| **Metoda de sortare 2 ( ex. heap sort)** |  |  |  |  |

1. **Să se elaboreze un program ce va aloca dinamic un tablou unidimensional de numere întregi și va implementa următoarele funcții, funcțiile vor fi organizate sub forma unui meniu:**

* inițializarea tabloului cu numere aleatorii;
* afișarea tabloului;
* eliberarea memoriei tabloului;
* căutarea unui număr din tablou utilizînd metoda de căutare linear search, se va afișa în cît timp se execută căutarea.
* căutarea unui număr din tablou utilizînd metoda de căutare binary search, se va afișa în cît timp se execută căutarea.

**2.2 Să se realizeze o analiză empirică a timpului necesar pentru a efectua 10000 de căutări într-un tablou aleator de lungimea N unde N va avea valori [100, 1000, 10000, 100000], rezultatele obținute se vor plasa în tabelul de mai jos:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **N** | **100** | **1000** | **10000** | **100000** |
| **Linear search** |  |  |  |  |
| **Binary search** |  |  |  |  |

**3.1 Să se realizeze un algoritm de calcul a unui element din șirul lui Fibonaci pe baza poziției acestuia N, de exemplu:**

**- fib(0) == 0;**

**- fib(1) == 1;**

**- fib(5) == 3;**

**- fib(100) == 354224848179261915075;**

**- fib(500) == 139423224561697880139724382870407283950070256587697307264108962948325571622863290691557658876222521294125;**

**Poziția unui element N, va fi în diapazonul [50;1000000] , asta înseamnă că algoritmul trebuie să poată calcula minim fib(50) și maxim fib(1000000).**

**3.2 Să se realizeze o analiză empirică a timpului necesar pentru a calcula numărul lui fibonaci de pe poziția N, unde N va avea valori [50, 100, 1000, 10000, 100000, 1000000], rezultatele obținute se vor plasa în tabelul de mai jos:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **N** | **50** | **100** | **1000** | **10000** | **100000** | **1000000** |
| **Fibonacci** |  |  |  |  |  |  |

**Barem de notare:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Problema** | **1.1, 1.2** | **1.1, 1.2, 2.1** | **1.1, 1.2, 2.1, 2.2** | **1.1, 1.2, 2.1, 2.2, 3.1,**  **3.2** | **Toate exercițiile și algoritm pentru fibonaci pînă la 1000** | **Toate exercițiile și algoritm pentru fibonaci de la 1000000** |
| **Nota** | **5** | **6** | **7** | **8** | **9** | **10** |

**Rezultatul**

Rezultaul lucrării se va plasa într-un raport, unde va fi inclusă foaia de titlu și codul sursă a programului cu o poză de execuție a acestuia și tabelele cu analiza empirică efectuată.