Курсова работа

Тема: Итеративни и рекурсивни алгоритми: обзор и сравнение, приложения и примери

Изготвил: Г.Гайдаджиев 43група ТУ - София

I. Увод

Алгоритмите са основна движеща сила в компютърните науки, какро и в света около нас. Те представляват крайни и еднозначни инструкции за решаване на определен проблем.Или по-просто казано алгоритъмът е действие за решаване на проблем и получаване на желания резултат. Ще разгледаме итеративните и рекурсивните алгоритми, ще ги сравним и ще демонстрираме техните приложения чрез примери както в живота така и в програмирането(чисто теоритично).

II. Обзор на итеративни алгоритми

Итеративните алгоритми използват цикли като "for" и "while" за многократно изпълнение на инструкции. Те са ефективни и лесно се оптимизират за задачи като обхождане на масиви, списък и като цяло всякаква структура от данни заложена в програмния език. Също так често се използват и за изчисляване на сумарни стойности.
Пример:

```java
public class Primeri {


    public static int findSum(int[] array) {
        int sum = 0;
        for (int num : array) {
            sum += num;
        }
        return sum;
    }

    Run | Debug
    public static void main(String[] args) {

        int[] numbers = {1, 2, 3, 4, 5};
        System.out.println("Sum of array elements: " + findSum(numbers));
    }
}
```

3. Обзор на рекурсивни алгоритми

Рекурсивните алгоритми представляват функция, която се извиква сама себе си. Те са изградени от базов случай, който прекратява рекурсията(без него ще се получи безкрайно циклене и ще изпаднем в капана на безкрайния цикъл) , и рекурсивен случай, който представлява самото извиква на функцията отново.

Пример за рекурсивен алгоритъм:

```java
public class Primeri {
    public static int factorial(int n) {
        if (n == 0) {
            return 1; // bazov sluchai
        }
        return n * factorial(n - 1); // rekursiven sluchai
    }

    Run | Debug
    public static void main(String[] args) {
        int number = 5;
        System.out.println("Factorial of " + number + " is: " + factorial(number));
    }
}
```

4. Сравнение между итеративни и рекурсивни алгоритми

И двата вида функции имат своите плюсове и минуси. Итеративните алгоритми са по-ефективни по отношение на паметта, тъй като не изискват допълнителен стек за изпълнение. Те обаче могат да бъдат по-сложни за четене и имплементация при някои задачи.
Рекурсията от своя страна е удобна за задачи с дървовидна структура или подход "разделяй и владей", но може да доведе до препълване на стека. Също така трябва да се отбележи, че при работа с рекурсия има по-голяма вероятност да се объркa.

5. Приложения

Итеративни алгоритми са ни добре познати и в някои видове сортировки и при обработката на данни.

- Bubble Sort, Insertion Sort….
- Обработка на големи масиви от данни.

Също както при итеративните алгоритми рекурсивни такива също се срещат в различни видове сортировки, но за разлика от другите алгоритми те са фундаментални за работата с дървовидни структури в даден програмен език:

- Разделяй и владей (Merge Sort, Quick Sort).
- Работа с дървовидни структури (дървета и графи).

Това бяха приложения в теоретичното програмиране. Тези видове алгоритми обаче се използват много разпространено в заобикаящия ни свят. Примери за итеративни алгоритми срещаме например във финансовите приложения, като обработка на месечни транзакции или изчисляване на средни стойности (напр. лихвени проценти), игрите - обхождане на игрално поле в шахмат или проверка на всички възможни ходове на играч, също така се наблюдават и в навигация нашумелите самоуправляващи се дронове и прахосмукачки използват именно такива алгоритми.

От своя страна пък рекурсивните алгоритми намират приложение в работата с файлови системи , като служат за обхождане на директории и поддиректории, за да се намерят или обработят всички файлове в компютърна система. АЛгоритмите служат и за обработка на естествен език, анализирайки изречения, разделени на думи, и изграждане на синтактични дървета. Както и за графични интерфейси -> рекурсиите служат за създаване на вложени менюта в приложения (например падащи менюта).

6. Заключение

В тази курсова работа като заключение може да кажем, че итеративните и рекурсивните алгоритми са неизменна част от нашия живот и ние използваме всеки ден, без да се замислям колко често го правим.