**1 - Výhody použití tabulky (memoizace) pro Fibonacciho posloupnost:**

1. **Rychlost** – Ukládá již spočítané hodnoty, takže se nepočítají znovu.
2. **Úspora času** – Místo exponenciální složitosti **O(2ⁿ)** máme **O(n)**.
3. **Jednoduchost** – Stačí se podívat do tabulky místo opakovaných výpočtů.

**2 - Jak se tabulka používá a plní daty?**

1. **Struktura**: Používáme slovník (HashMap<Integer, Integer>), kde klíč je n a hodnota je fib(n).
2. **Plnění**:
   * Pokud fib(n) není v tabulce, spočítáme rekurzivně fib(n-1) + fib(n-2).
   * Výsledek uložíme do tabulky.
   * Při dalším volání se hodnota pouze načte z tabulky.
3. **Inicializace**: Na začátku uložíme základní případy fib(0)=0 a fib(1)=1.

**Příklad kódu**:

java

if (!tabulka.containsKey(n)) {

int vysledek = fib(n-1) + fib(n-2);

tabulka.put(n, vysledek);

}

return tabulka.get(n);

**3 - Význam tabulky v tomto problému**

* **Optimalizace rekurze** – Zabrání zbytečnému opakování stejných výpočtů.
* **Umožňuje počítat velká čísla** – Např. fib(100) by bez tabulky trvalo roky, s tabulkou milisekundy.
* **Příklad dynamického programování** – Ukazuje, jak ukládání mezivýsledků zrychlí algoritmus.

**Příklad výstupu pro fib(5)**:

Tabulka: {0=0, 1=1, 2=1, 3=2, 4=3, 5=5}

Každé číslo se spočte jen **jednou**, pak se bere z paměti.