Gemini Chat

Dokaž mi prosím, že každý n-tý člen Fibonacciho posloupnosti je menší než 2^n.

Jasně, dokážeme, že každý n-tý člen Fibonacciho posloupnosti, označený F_n, je menší než 2^n. Použijeme k tomu metodu matematické indukce.
Fibonacciho posloupnost je definována takto:

- $F_0 = 0$
- $F_1 = 1$
- $F_n = F_{n-1} + F_{n-2}$ pro $n \geq 2$

Důkaz matematickou indukcí

1. Báze indukce:
Ověříme tvrzení pro první dva členy posloupnosti:

- Pro $n = 1$: $F_1 = 1$. A $2^1 = 2$. Platí $1 < 2$.
- Pro $n = 2$: $F_2 = F_1 + F_0 = 1 + 0 = 1$. A $2^2 = 4$. Platí $1 < 4$.

Tvrzení platí pro $n = 1$ a $n = 2$.

---

2. Indukční předpoklad:
Předpokládejme, že tvrzení platí pro všechna přirozená čísla k až do n, tedy že:

- $F_k < 2^k$ pro všechna $k \leq n$.
  Konkrétně pro náš další krok to znamená:
- $F_{n-1} < 2^{n-1}$
- $F_n < 2^n$

---

3. Indukční krok:
Musíme dokázat, že tvrzení platí i pro $n + 1$, tedy že $F_{n+1} < 2^{n+1}$.
Víme, že $F_{n+1} = F_n + F_{n-1}$.
Použijeme náš indukční předpoklad:

- $F_n < 2^n$
- $F_{n-1} < 2^{n-1}$

Sečtením těchto dvou nerovností dostaneme:

$F_{n+1} = F_n + F_{n-1} < 2^n + 2^{n-1}$
Nyní upravíme pravou stranu nerovnosti:
$2^n + 2^{n-1} = 2 \cdot 2^{n-1} + 2^{n-1} = (2+1) \cdot 2^{n-1} = 3 \cdot 2^{n-1}$
A protože $3 < 4$, platí:
$3 \cdot 2^{n-1} < 4 \cdot 2^{n-1}$
$4 \cdot 2^{n-1} = 2^2 \cdot 2^{n-1} = 2^{2+n-1} = 2^{n+1}$
Z toho plyne, že:
$F_{n+1} < 2^n + 2^{n-1} < 3 \cdot 2^{n-1} < 4 \cdot 2^{n-1} = 2^{n+1}$
Tedy:
$F_{n+1} < 2^{n+1}$

Závěr:
Na základě principu matematické indukce jsme dokázali, že každý n-tý člen Fibonacciho posloupnosti je menší než 2^n pro všechna přirozená čísla $n \geq 1$.

Ukaž mi grafické porovnání růstu.

Grafické porovnání růstu F_n a 2^n

Zde je grafické znázornění růstu Fibonacciho posloupnosti (F_n) ve srovnání s 2^n. Z grafu je patrné, že 2^n roste mnohem rychleji než F_n, což vizuálně potvrzuje naše tvrzení, že $F\_n < 2^n$.

```
import matplotlib.pyplot as plt
import numpy as np

# Funkce pro výpočet Fibonacciho posloupnosti
def fibonacci(n):
    if n <= 0:
        return 0
    elif n == 1:
        return 1
    else:
        a, b = 0, 1
        for _ in range(2, n + 1):
            a, b = b, a + b
        return b

# Generování dat
n_values = np.arange(0, 15) # Od 0 do 14 pro lepší vizualizaci počáteční
ch členů
```

```
ch clenu
fib_values = [fibonacci(n) for n in n_values]
power_of_2_values = [2**n for n in n_values]

# Vytvoření grafu
plt.figure(figsize=(10, 6))
plt.plot(n_values, fib_values, marker='o', linestyle='-', color='blue',
label='$F_n$ (Fibonacciho posloupnost)')
plt.plot(n_values, power_of_2_values, marker='x', linestyle='--', color
='red', label='$2^n$')

# Nastavení popisků a názvů
plt.title('Porovnání růstu Fibonacciho posloupnosti a 2^n')
plt.xlabel('n')
plt.ylabel('Hodnota')
plt.xticks(n_values)
plt.grid(True, linestyle='--', alpha=0.6)
plt.legend()
plt.yscale('linear') # Můžete zkusit i 'log' pro jiný pohled na rané hod
noty
plt.show()
```

Vysvětlení grafu:

- **Modrá plná čára s kroužky** reprezentuje hodnoty **Fibonacciho posloupnosti (** F_n**)**. Můžete vidět, jak pozvolna začíná a pak se její růst zrychluje.
- **Červená přerušovaná čára s křížky** reprezentuje hodnoty 2^n. Je zjevné, že tato funkce roste mnohem agresivněji než Fibonacciho posloupnost, a rychle se od ní vzdaluje.

Tento graf názorně ukazuje, že pro všechna n
$ge1$ je hodnota F_n vždy pod hodnotou 2^n.