Algorytmy i struktury danych (studia stacjonarne)

Dr Anna Muranova

Semestr zimowy 2025/2026, UWM w Olsztynie

Ćwiczenie 2 (Rekurencja) 15.10.2025

Wszystkie zadanie wykonać metodą rekurencyjną oraz bez niej. Jak jest optymalnej? W razie wątpliwości użyj globalną zmienną dla obliczenia ilości operacje w każdym przypadku.

Zadanie 2.1. Zaimplementować funkcję `numbers(n: int)`, która wypisze liczby od n do 0.

Zadanie 2.2. Zaimplementować funkcję `reverse(txt: str) -> str`, która zwróci odwrócony ciąg znaków przekazany w parametrze txt.

Zadanie 2.3. Zaimplementować wyszukiwanie binarne `search(n: int, m: int, p: int, numbers: Array)-> Tuple[bool, int]` metodą rekurencyjną.

Zadanie 2.4. Zaimplementować funkcję `n_sum(n: int, m: int) -> list[int]`, która zwróci wszystkie n-cyfrowe liczby z sumą cyfr $m > 0$.

Zadanie 2.5. * Zaimplementować funkcję `n_sums(m: int, p: int, np: int) -> list[int]`, która zwróci wszystkie n-cyfrowe liczby z sumą cyfr p na indeksach parzystych i np na nieparzystych.

Zadanie 2.6. Zaimplementować funkcję `power(number: int, n: int) -> int`, której zadaniem jest zwrócenie wyniku działania $number^n$

- wprost,
- przez algorytm szybkiego potęgowania

  `https://pl.wikipedia.org/wiki/Algorytm_szybkiego_pot%C4%99gowania`

NIE UŻYWAĆ OPERATORA **

Zadanie 2.7. Zaimplementować funkcję `calculator(t: tuple) -> int`, która oblicze wyraz t podany w postaci trójelementowej krotki, w której skrajne elementy są wyrazami lub liczbami, a środkowe znakiem "+łub *-*. Przykład

```
calculator (((5,'+',-1),'+',3), '-', 2) -> 5
```