KARTA PRACY – ZŁOŻONOŚĆ OBLICZENIOWA ALGORYTMÓW

Imię i nazwisko: ..... **Klasa:** ..... **Data:** .....

INSTRUKCJE

Poniższa karta pracy opiera się na artykule "Jak ocenić złożoność obliczeniową algorytmu?". Zanim zaczniesz rozwiązywać zadania, przeczytaj artykuł.

CZĘŚĆ 1: PYTANIA SPRAWDZAJĄCE

Odpowiedz na poniższe pytania na podstawie artykułu. Odpowiadaj pełnymi zdaniami.

1. Wyjaśnij, jakie są dwa główne sposoby wyszukiwania numeru telefonu w książce telefonicznej. Który z nich jest bardziej efektywny i dlaczego?

_____

_____

_____

_____

2. Co to jest algorytm? Podaj przykład algorytmu z życia codziennego (nie koniecznie z programowania).

_____

_____

_____

_____

3. Dlaczego "jak rośnie" czas lub pamięć jest ważniejsze niż konkretna liczba operacji? Przytocz przykład z tekstu.

_____

_____

_____

_____

4. Co oznacza litera 'n' w kontekście złożoności obliczeniowej? Podaj dwa przykłady, co mogłoby być 'n'.

_____

_____

_____

CZĘŚĆ 2: NOTACJA DUŻEGO O

Zadanie: Dopasuj typ złożoności do opisanego scenariusza.

Wybierz z: O(1), O(log n), O(n), O(n^2), O(2^n)

SCENARIUSZ	ZŁOŻONOŚĆ
Szukasz swojego imienia w alfabetycznej liście wszystkich uczniów, czytając od góry do dołu	
Znasz dokładny numer szafki – otwierasz ją od razu	
Każdy uczeń musi przywitać się z każdym innym uczeniem	
Szukasz w dzienniku – otwierasz w połowie i zawężasz zakres	

CZĘŚĆ 3: ANALIZA KODU

Dla każdego fragmentu kodu określ złożoność obliczeniową. Wyjaśnij swój wybór.

1. Poniższy kod sprawdza, czy liczba 7 jest na liście:

```
for i in range(n):
    if lista[i] == 7:
        return i
```

Złożoność: _____

Wyjaśnienie: _____

_____

2. Poniższy kod porównuje każdy element z każdym:

```
for i in range(n):
    for j in range(n):
        if lista[i] == lista[j]: print("Match!")
```

Złożoność: _____

Wyjaśnienie: _____

CZĘŚĆ 4: ZADANIA PRAKTYCZNE

1. PORÓWNANIE ALGORYTMÓW

Wyobraź sobie, że szukasz numeru w książce telefonicznej z 10 000 numerów.

a) Wyszukiwanie liniowe (od początku do końca):

Ile maksymalnie kroków? _____

b) Wyszukiwanie binarne (dzielenie na pół):

Ile maksymalnie kroków? _____

c) Jaka jest różnica? Ile razy szybsze jest wyszukiwanie binarne?

_____

2. SKALOWANIE ALGORYTMÓW

Algorytm A ma złożoność O(n), a algorytm B ma złożoność $O(n^2)$.

a) Jeśli n = 100, ile operacji wykonuje każdy algorytm?

_____

b) Jeśli n wzrośnie do 1000, ile operacji będzie wykonywać każdy algorytm?

_____

c) Który algorytm "skaluje się" lepiej? Dlaczego?

_____

CZĘŚĆ 5: WYZWANIA DODATKOWE

Dla tych, którzy chcą pójść dalej:

1. Wyobraź sobie funkcję, która sumuje wszystkie elementy listy (przegląda każdy element raz). Jaka jest jej złożoność? Czy mogłaby mieć lepszą złożoność? Dlaczego lub dlaczego nie?

_____

_____

2. Artykuł wspomina "złożoność pamięciową". Wyjaśnij, jaka jest różnica między czasową i pamięciową. Czy zawsze szybszy algorytm zajmuje mniej pamięci?

_____

_____

3. Funkcja fibonacci może być napisana na wiele sposobów. Proste podejście rekurencyjne ma złożoność O(2^n). Dlaczego?

   (Wskazówka: ile razy funkcja wywołuje siebie samą?)

_____

_____

REFLEKSJA

Napisz krótką refleksję (3-5 zdań) na temat tego, czego nauczyłeś się dzisiaj o złożoności obliczeniowej:

_____

_____

_____

_____