PIERWSZA ŁAWKA · KURS INFORMATYKA · INFORMATYKA · MATURA PR · ROZSZERZONY

Rekurencja i metoda dziel i zwyciężaj

Funkcje rekurencyjne i warunek zakończenia, analiza wywołań, rekurencyjne wersje algorytmu Euklidesa i szybkiego potęgowania, metoda zstępująca i wstępująca (spamiętywanie), dziel i zwyciężaj: minimum i maksimum, sortowanie przez scalanie, fraktale rekurencyjne. Sześć kart pracy zgodnych z podstawą 2024 i maturą 2027 – z zadaniem przy komputerze na plikach danych (archiwum ZIP), kluczem, zasadami oceniania i numerami wymagań.

Karta	Temat	Zadania
1	**Rekurencja i metoda dziel i zwyciężaj – Teoria krok po kroku** Wszystko, co trzeba wiedzieć, z przykładami	2 zad. 4 pkt
2	**Rekurencja i metoda dziel i zwyciężaj – Rozgrzewka** Proste zastosowania – sprawdź, czy rozumiesz	2 zad. 6 pkt
3	**Rekurencja i metoda dziel i zwyciężaj – Zadania maturalne, część 1** Analiza algorytmów, tabele i zapis algorytmu – jak w arkuszu	3 zad. 11 pkt
4	**Rekurencja i metoda dziel i zwyciężaj – Zadania maturalne, część 2** Trudniejsze zadania i uzasadnienia	4 zad. 17 pkt
5	**Rekurencja i metoda dziel i zwyciężaj – Zadanie z danymi – jak na egzaminie** Praca z plikami danych przy komputerze – wyniki w pliku wyniki1.txt	1 zad. 10 pkt
6	**Rekurencja i metoda dziel i zwyciężaj – Sprawdź się** 30 minut · 13 pkt	5 zad. 13 pkt

JAK KORZYSTAĆ Z ZESTAWU

- Karta 1 – teoria z przykładami. Karty 2–4 – zadania w formie maturalnej: analiza algorytmów, tabele, zapis algorytmu ze specyfikacją (Dane, Wynik) i ograniczeniami z uwagi pod treścią.
- Karta 5 – zadanie przy komputerze: w archiwum ZIP pełne dane i pliki *_przyklad.txt z odpowiedziami kontrolnymi; wyniki w pliku wyniki1.txt z numerami zadań, kod w osobnych plikach. Karta 6 – zestaw kontrolny na czas.
- Ramki „Uwaga na maturze” – błędy, przez które zdający najczęściej tracą punkty (sprawozdania CKE 2023–2026).
- Klucz: rozwiązania wzorcowe (Python; C++ i Java są równoprawne), zasady oceniania i wymagania podstawy (Dz.U. 2024 poz. 1019): I.n, II.n, I+II.n – zakres rozszerzony, P.… – podstawowy. Wyniki dla plików danych policzono programem.

Materiał oryginalny zgodny z podstawą programową informatyki w brzmieniu z 2024 r. (Dz.U. 2024 poz. 1019) i z formułą egzaminu maturalnego z informatyki w 2027 r. Nie zawiera zadań z arkuszy CKE. Do użytku własnego i pracy z własnymi uczniami.

Rekurencja i metoda dziel i zwyciężaj – Teoria krok po kroku

Wszystko, co trzeba wiedzieć, z przykładami

KARTA 1/6

Imię i nazwisko: Data: Wynik: ..... / 4 pkt

PRZYPOMNIJ SOBIE

- **Funkcja rekurencyjna** wywołuje samą siebie dla mniejszych danych. Musi mieć **warunek zakończenia** (przypadek bazowy), a każde wywołanie musi zbliżać dane do tego przypadku – inaczej program przerwie się przepełnieniem stosu.
- **Stos wywołań**: każde wywołanie przechowuje własne zmienne i adres powrotu, aż się zakończy. Pamięć rośnie z *głębokością* rekurencji (w Pythonie domyślny limit to ok. 1000 poziomów – sys.setrecursionlimit go zwiększa).
- **Analiza wywołań** (typowe zadanie maturalne bez komputera): rysujemy drzewo wywołań, zapisujemy argumenty i wyniki. Liczba wywołań spełnia zależność podobną do samej funkcji, np. dla funkcji z dwoma wywołaniami dla $n - 1$ i $n - 2$ rośnie wykładniczo.
- **Metoda zstępująca** (od całego problemu do mniejszych – rekurencja) i **wstępująca** (od najmniejszych przypadków do całego – pętla wypełniająca tablicę). Gdy te same podproblemy powtarzają się, zapamiętujemy ich wyniki (spamiętywanie) albo liczymy wstępująco – złożoność spada z wykładniczej do wielomianowej. W podstawie programowej ciąg Fibonacciego liczymy iteracyjnie.
- **Rekurencyjny algorytm Euklidesa**: $\text{nwd}(a, b) = a$ dla $b = 0$, w przeciwnym razie $\text{nwd}(b, a \bmod b)$. **Rekurencyjne szybkie potęgowanie**: $a^0 = 1$, $a^n = p \cdot p$ ($\cdot a$ dla n nieparzystego), gdzie $p = a^{n \text{ div } 2}$ liczone *raz* – około $\log_2 n$ wywołań.
- **Dziel i zwyciężaj**: 1) podziel problem na mniejsze tego samego rodzaju, 2) rozwiąż je rekurencyjnie, 3) połącz wyniki. Przykłady: sortowanie przez scalanie ($T(n) = 2T(n/2) + n \rightarrow O(n \log n)$), sortowanie szybkie (średnio $O(n \log n)$, pesymistycznie $O(n^2)$), jednoczesne minimum i maksimum ($T(n) = 2T(n/2) + 2 \rightarrow 3n/2 - 2$ porównań), wyszukiwanie binarne.
- **Fraktale rekurencyjne**: obraz stopnia n składa się z kilku pomniejszonych kopii obrazu stopnia $n - 1$. Trójkąt Sierpińskiego stopnia n ma 3^n zamalowanych trójkątów o boku $a/2^n$; krzywa Kocha stopnia n – 4^n odcinków o długości $a/3^n$ (łączna długość rośnie jak $(4/3)^n$).
- **Rekurencja czy iteracja**: rekurencja jest naturalna dla struktur rekurencyjnych (drzewa katalogów, fraktale, dziel i zwyciężaj); iteracja nie zużywa stosu. Każdą rekurencję da się zapisać iteracyjnie (czasem z własnym stosem).
- **Jak zapisać algorytm, żeby dostać punkty**: pseudokod albo wybrany język (Python, C++, Java) – oba zapisy są równoprawne. Trzymaj się nazw ze *Specyfikacji* (Dane/Wynik), na końcu jawnie ustal wynik (zwróć go albo wypisz), przestrzegaj ograniczeń z *Uwagi* pod treścią i sprawdź algorytm „na sucho” na przykładzie z treści oraz na przypadku brzegowym (np. $n = 1$, liczba 0).

Podgląd • Pierwsza Ławka

PRZYKŁADY Z WYJAŚNIENIEM

Przykład 1. Zapisz silnię rekurencyjnie i wskaż warunek zakończenia.

silnia(n) = 1 dla $n \leq 1$ (warunek zakończenia), w przeciwnym razie $n \cdot$ silnia$(n - 1)$. Dla $n = 4$ są cztery wywołania: silnia(4), silnia(3), silnia(2), silnia(1); wynik $4 \cdot 3 \cdot 2 \cdot 1 = 24$.

Przykład 2. Ile ruchów potrzeba w wieżach Hanoi dla n krążków?

$R(1) = 1$, $R(n) = 2 \cdot R(n - 1) + 1$ (przenieś $n - 1$, największy, znów $n - 1$), stąd $R(n) = 2^n - 1$: dla 3 krążków 7, dla 10 – 1023.

Przykład 3. Ile wywołań wykona rekurencyjne szybkie potęgowanie dla n = 16?

$n = 16, 8, 4, 2, 1, 0$ – sześć wywołań ($\log_2 16 + 2$). Wywołanie potega(a, n div 2) dwa razy zamiast raz dałoby aż 63 wywołania (dla n będącego potęgą dwójki $4n - 1$) – ich liczba rosłaby liniowo z n zamiast logarytmicznie.

Przykład 4. Dlaczego wyszukiwanie binarne to „dziel i zwyciężaj”?

Dzieli przedział na dwie połowy i rozwiązuje problem tylko w jednej z nich – stąd około $\log_2 n$ kroków.

ZADANIA

Zadanie 1. Zapisz rekurencyjną funkcję obliczającą sumę cyfr liczby naturalnej n. ___ / 3 pkt

Uwaga: Twój algorytm może używać tylko zmiennych przechowujących pojedyncze liczby całkowite, operacji arytmetycznych (dodawanie, odejmowanie, mnożenie, dzielenie całkowite div i reszta z dzielenia mod), porównań, operatorów logicznych, instrukcji warunkowych, pętli i przypisań oraz samodzielnie napisanych funkcji. Nie wolno używać funkcji wbudowanych, tablic ani list, ani zamieniać liczby na napis (i odwrotnie).

Specyfikacja

Dane: n – liczba całkowita, $n \geq 0$

Wynik: suma cyfr zapisu dziesiętnego n

Zadanie 2. Co się stanie, gdy funkcja rekurencyjna nie ma osiągalnego warunku zakończenia?

___ / 1 pkt

A. Kompilator zawsze wykryje błąd.

B. Zwróci 0.

C. Program przerwie się z powodu przepełnienia stosu wywołań.

D. Zapętli się bez zużywania pamięci.

Podgląd • Pierwsza Ławka

Rekurencja i metoda dziel i zwyciężaj – Zadanie z danymi – jak na egzaminie

Praca z plikami danych przy komputerze – wyniki w pliku wyniki1.txt

KARTA 5/6

Imię i nazwisko: ..... Data: ..... Wynik: ..... / 10 pkt

Zadanie 1. Plik katalogi.txt opisuje drzewo katalogów na dysku. W każdym z 300 wierszy zapisano: nazwę katalogu, nazwę jego katalogu nadrzędnego (dla katalogu głównego root jest to znak „-") oraz łączny rozmiar plików leżących bezpośrednio w tym katalogu (w KB). Nazwy katalogów są unikatowe; wiersze są w przypadkowej kolejności. *Rozmiarem całkowitym* katalogu nazywamy sumę rozmiaru jego plików i rozmiarów całkowitych wszystkich jego podkatalogów. Napisz program (lub kilka programów), który da odpowiedzi do poniższych zadań. Odpowiedzi zapisz w pliku wyniki1.txt, poprzedzając każdą numerem zadania. ___ / 10 pkt

Pierwsze wiersze pliku katalogi.txt

```
zopoka lozole 51
lejasa zozoja 0
lote mida 0
sarupo giru 2051
hobe root 0
…
```

Do dyspozycji masz także **katalogi_przyklad.txt** (25 wierszy) – odpowiedzi dla nich podano w treściach zadań. Twój program musi ostatecznie działać na pełnych danych (**katalogi.txt**).

Do oceny oddajesz: plik **wyniki1.txt** z odpowiedziami do zadań 1.1–1.4 (każda poprzedzona numerem zadania) oraz pliki z kodem źródłowym, np. **zad1_1.py**, **zad1_2.py**… (lub .cpp, .java). Brak plików z kodem oznacza brak rozwiązania.

Dane przygotowane na potrzeby kursu (archiwum ZIP dołączone do opracowania, katalog DANE)

UWAGA NA MATURZE Rekurencja o głębokości kilkudziesięciu poziomów jest bezpieczna, ale dla bardzo głębokich drzew w Pythonie zwiększ limit (sys.setrecursionlimit) albo użyj własnego stosu. Plik przykładowy pozwala sprawdzić, czy liczysz rozmiary przez wszystkie poziomy.

1.1. Głębokość katalogu root wynosi 1, a głębokość każdego innego katalogu jest o 1 większa niż głębokość jego katalogu nadrzędnego. Podaj największą głębokość katalogu oraz liczbę katalogów o tej głębokości. Odpowiedź dla danych przykładowych: 10 1 ___ / 2 pkt

PIERWSZA ŁAWKA

Rekurencja i metoda dziel i zwyciężaj – Sprawdź się

30 minut · 13 pkt

KARTA 6/6

Imię i nazwisko: ..... Data: ..... Wynik: ..... / 13 pkt

Zadanie 1. Zapisz iteracyjną wersję funkcji obliczającej sumę cyfr liczby naturalnej n. ___ / 2 pkt

Uwaga: Twój algorytm może używać tylko zmiennych przechowujących pojedyncze liczby całkowite, operacji arytmetycznych (dodawanie, odejmowanie, mnożenie, dzielenie całkowite div i reszta z dzielenia mod), porównań, operatorów logicznych, instrukcji warunkowych, pętli i przypisań oraz samodzielnie napisanych funkcji. Nie wolno używać funkcji wbudowanych, tablic ani list, ani zamieniać liczby na napis (i odwrotnie).

Specyfikacja
Dane: n – liczba całkowita, $n \geq 0$
Wynik: suma cyfr zapisu dziesiętnego n

Zadanie 2. Ile razy zostanie wywołana rekurencyjna funkcja potega(a, n) z karty 3 dla n = 1000 (razem z pierwszym wywołaniem)? ___ / 2 pkt

Zadanie 3. Oceń prawdziwość zdań. ___ / 4 pkt

1. Każdą funkcję rekurencyjną można zapisać iteracyjnie.	**P**	**F**
2. Pesymistyczna liczba porównań w sortowaniu przez scalanie rośnie jak n^2 (a nie jak n log n).	**P**	**F**
3. Krzywa Kocha stopnia n składa się z 4^n odcinków.	**P**	**F**
4. Spamiętywanie wyników podproblemów może zmniejszyć złożoność z wykładniczej do wielomianowej.	**P**	**F**

Zadanie 4. Na czym polega metoda dziel i zwyciężaj? Podaj jej trzy etapy na przykładzie sortowania przez scalanie. ___ / 3 pkt

Zadanie 5. Ile zamalowanych trójkątów ma trójkąt Sierpińskiego stopnia 4, a ile odcinków ma krzywa Kocha stopnia 3? ___ / 2 pkt

CO JESZCZE JEST W ŚRODKU

Rekurencja i metoda dziel i zwyciężaj – opracowanie maturalne z informatyki (6 kart pracy z kluczem)

Podgląd pokazał 5 z 16 stron. Pełna wersja: 16 stron A4, PDF do druku i DOCX do edycji.

- Karta 1: Teoria krok po kroku – 2 zadania, 4 pkt
- Karta 2: Rozgrzewka – 2 zadania, 6 pkt
- Karta 3: Zadania maturalne, część 1 – 3 zadania, 11 pkt
- Karta 4: Zadania maturalne, część 2 – 4 zadania, 17 pkt
- Karta 5: Zadanie z danymi – jak na egzaminie – 1 zadanie, 10 pkt
- Karta 6: Sprawdź się – 5 zadań, 13 pkt Razem 16 stron A4 w PDF do druku oraz ten sam materiał w DOCX do edycji w Wordzie, a do tego archiwum ZIP z plikami danych do zadania z karty

Kup pełną wersję – plik od razu na e-mail

pierwszalawka.pl/material/matura-informatyka-rekurencja-i-metoda-dziel-i-zwyciezaj-opracowanie