KLUCZ ODPOWIEDZI · DLA NAUCZYCIELA

Sprawdzian z teorii informatyki

Grupa A

1. (1 pkt) **B**

2. (1 pkt) **D**

3. (3 pkt) **1. P; 2. F; 3. P** — po 1 pkt za zdanie

4. (3 pkt) **1. P; 2. F; 3. F** — po 1 pkt za zdanie

5. (4 pkt) **1–A, 2–D, 3–C, 4–B** — po 1 pkt za parę

6. (2 pkt) **(1) DNS; (2) zapora sieciowa** — po 1 pkt za lukę

7. (2 pkt) Przykładowa odpowiedź: Proces to samodzielnie wykonywany program posiadający własną, odizolowaną przestrzeń adresową pamięci, natomiast wątek to jednostka wykonania działająca wewnątrz procesu — jeden proces może mieć wiele wątków, które współdzielą tę samą pamięć i zasoby procesu. Przełączanie kontekstu między wątkami tego samego procesu jest szybsze niż między procesami, ponieważ system operacyjny nie musi zmieniać mapowania pamięci (przestrzeni adresowej) — wątki już współdzielą te same dane i kod, więc trzeba przełączyć jedynie stan rejestrów procesora, a nie całą tablicę stron pamięci.
Punktacja: Poprawne rozróżnienie procesu (samodzielny program z własną przestrzenią adresową) od wątku (jednostka wykonania wewnątrz procesu) (1 pkt) + wyjaśnienie, że wątki tego samego procesu współdzielą pamięć, więc przełączanie między nimi nie wymaga zmiany kontekstu pamięci, w przeciwieństwie do procesów (1 pkt). 2 pkt.

8. (3 pkt) Przykładowa odpowiedź: Halting problem: czy istnieje algorytm HALT(P,W) który odpowie, czy P zatrzyma się na W? Turing (1936) udowodnił nierozstrzygalność przez sprzeczność: gdyby HALT istniało, można zbudować program D który się zapętla gdy HALT(D,D)=tak i zatrzymuje gdy HALT(D,D)=nie - sprzeczność. Wniosek: niektórych problemów nie można rozwiązać algorytmicznie.
Punktacja: 1. Opis: czy program P zatrzyma się dla wejścia W? (1 pkt). 2. Dowód: sprzeczność przez auto-referencję (1 pkt). 3. Konsekwencje: istnieją problemy poza MT (1 pkt).

9. (3 pkt) Przykładowa odpowiedź: CFG to czwórka (V, Σ, R, S): V — skończony zbiór nieterminali (np. E, T, F), Σ — alfabet terminali (np. +, ×, id, (,)), R — zbiór reguł produkcji postaci $A \to \alpha$, S — symbol startowy. Reguły dla wyrażeń: $E \to E + T \mid T$; $T \to T \times F \mid F$; $F \to \mathrm{id} \mid (E)$. Zastosowanie: parsery i kompilatory używają CFG do opisu gramatyki języków programowania.
Punktacja: Formalna definicja CFG jako czwórki (V, Σ, R, S): nieterminale V, terminale Σ, reguły R, symbol startowy S — 1 pkt. Poprawne reguły produkcji (np. $E \to E + T \mid T$, $T \to T \times F \mid F$, $F \to \mathrm{id} \mid (E)$) lub uproszczone poprawne reguły — 1 pkt. Zastosowanie (parsery, kompilatory, opis składni języków programowania) — 1 pkt.

10. (3 pkt) Przykładowa odpowiedź: Problem NP-zupełny to problem, który należy do klasy NP (rozwiązanie można zweryfikować w czasie wielomianowym) i jest NP-trudny (każdy problem z NP można do niego wielomianowo zredukować). Przykład: problem spełnialności formuł logicznych (SAT) — czy istnieje przypisanie wartości logicznych spełniające formułę. Dowód P=NP oznaczyłby, że wszystkie problemy NP-zupełne mają wydajne algorytmy: można by lamic szyfry RSA i AES w czasie wielomianowym, co załamałoby bezpieczeństwo internetu, ale rownoczesnie umożliwiłoby efektywne leczenie optimalizacji trasy dostaw, foldowania białek i tysięcy innych zadań.
Punktacja: Kryterium 1 (1pkt): Poprawna definicja NP-zupełności — problem należy do NP (weryfikowalny wielomianowo) i każdy problem z NP można do niego wielomianowo zredukować. Kryterium 2 (1pkt): Poprawny przykład: SAT, problem komiwojażera (TSP), kolorowanie grafu, problem plecakowy, pokrycie wierzchołkowe. Kryterium 3 (1pkt): Implikacje P=NP — efektywne łamanie szyfrów RSA/AES, rozwiązanie problemów optymalizacyjnych, rewolucja w bioinformatyce i logistyce. 3 pkt.

Grupa B

1. (1 pkt) **D**

2. (1 pkt) **C**

3. (3 pkt) **1. P; 2. F; 3. P** — po 1 pkt za zdanie

4. (3 pkt) **1. F; 2. P; 3. F** — po 1 pkt za zdanie

5. (4 pkt) **1–B, 2–D, 3–A, 4–C** — po 1 pkt za parę

6. (2 pkt) **(1) P; (2) kompilator** — po 1 pkt za lukę

7. (2 pkt) Przykładowa odpowiedź: Wirtualizacja polega na tworzeniu programowej (wirtualnej) wersji zasobów sprzętowych, np. całego komputera z własnym systemem operacyjnym, działającej w izolowanym środowisku na fizycznym serwerze, nadzorowanej przez specjalne oprogramowanie zwane hipernadzorcą (hypervisorem). Uruchamianie wielu maszyn wirtualnych na jednym serwerze pozwala efektywniej wykorzystać jego zasoby (procesor, pamięć), zamiast utrzymywać osobny fizyczny serwer dla każdej aplikacji, a przy tym izoluje środowiska od siebie, dzięki czemu awaria lub błąd w jednej maszynie wirtualnej nie wpływa na pozostałe.
Punktacja: Poprawne wyjaśnienie pojęcia: technika tworzenia wirtualnych (symulowanych programowo) zasobów sprzętowych, np. wirtualnych komputerów, działających niezależnie od siebie na jednym fizycznym urządzeniu, nadzorowanych przez hipernadzorcę (hypervisor) (1 pkt). Poprawne wskazanie korzyści, np. lepsze wykorzystanie zasobów sprzętowych jednego serwera, izolacja aplikacji/środowisk, oszczędność kosztów sprzętu (1 pkt). Łącznie 2 pkt.

8. (3 pkt) Przykładowa odpowiedź: Kompilacja polega na przetłumaczeniu całego kodu źródłowego na kod maszynowy przed uruchomieniem programu — powstaje samodzielny plik wykonywalny (np. .exe), który można uruchamiać wielokrotnie bez ponownego tłumaczenia. Przykładem języka kompilowanego jest C++. Interpretacja polega na odczytywaniu i wykonywaniu kodu źródłowego instrukcja po instrukcji w trakcie działania programu, bez tworzenia osobnego pliku wykonywalnego — wymaga obecności interpretera przy każdym uruchomieniu. Przykładem języka interpretowanego jest Python.
Punktacja: 1 pkt — poprawny opis kompilacji (cały kod źródłowy jest tłumaczony na kod maszynowy przed uruchomieniem, powstaje plik wykonywalny). 1 pkt — poprawny opis interpretacji (kod jest odczytywany i wykonywany instrukcja po instrukcji przez interpreter, bez wcześniejszego tworzenia pliku wykonywalnego). 1 pkt — poprawne przykłady języków (np. C/C++ jako kompilowany, Python jako interpretowany). Łącznie 3 pkt.

9. (3 pkt) Przykładowa odpowiedź: Złożoność pesymistyczna (worst-case) określa maksymalną liczbę operacji, jaką algorytm może wykonać dla najgorszego możliwego zestawu danych wejściowych o danym rozmiarze. Złożoność średnia (average-case) opisuje oczekiwaną liczbę operacji przy założeniu typowego, losowego rozkładu danych wejściowych. Przykładem jest algorytm quicksort — jego złożoność pesymistyczna wynosi $O(n^2)$ i występuje np. dla danych już posortowanych przy naiwnym wyborze pivotu, natomiast złożoność średnia wynosi O(n log n) i dotyczy typowych, losowo ułożonych danych.
Punktacja: Poprawne wyjaśnienie złożoności pesymistycznej jako maksymalnej liczby operacji dla najgorszych możliwych danych wejściowych (1 pkt) + poprawne wyjaśnienie złożoności średniej jako oczekiwanej liczby operacji przy losowym rozkładzie danych (1 pkt) + trafny przykład, np. quicksort: pesymistyczna $O(n^2)$, średnia O(n log n) (1 pkt). Razem 3 pkt.

10. (3 pkt) Przykładowa odpowiedź: DFA to krotka (Q, Sigma, delta, q0, F) gdzie Q - stany, Sigma - alfabet, delta - funkcja przejścia, q0 - stan początkowy, F - stany akceptujące. Automat z jednym stanem q0 (akceptującym) i przejściem delta(q0, '1') = q0 akceptuje język 1* (dowolna liczba jedynek, łącznie z pustym słowem).
Punktacja: 1. Definicja DFA: 5-elementowa krotka (Q, Sigma, delta, q0, F) (1 pkt). 2. Opis elementów: stany, alfabet, funkcja przejścia, stan początkowy, stany akceptujące (1 pkt). 3. Przykład: automat akceptujący ciągi złożone wyłącznie z '1' (ciągi ze zbioru 1*) (1 pkt).

Skala ocen (orientacyjna, 25 pkt w każdej grupie)

bardzo dobry (5)	23–25 pkt
dobry (4)	19–22 pkt
dostateczny (3)	13–18 pkt
dopuszczający (2)	8–12 pkt
niedostateczny (1)	0–7 pkt

Progi: 90%, 75%, 50%, 30%. Progi w Twojej szkole mogą być inne. W zadaniach otwartych uznaj każdą odpowiedź poprawną merytorycznie, także sformułowaną inaczej niż przykład.

Ten sprawdzian online, z wyjaśnieniem każdej odpowiedzi: **matury-online.pl/sprawdzian/informatyka/teoria-informatyki**. Nauczyciele i korepetytorzy ułożą własną kartę z banku ponad 20 tys. zadań: **matury-online.pl/dla-korepetytorow**.