Jakub "morgul" Adamczyk

Egzaminy wejściowe - opracowanie

Pytania pochodzą zarówno z oficjalnej listy na rok 2020/2021, jak i z poprzednich lat z informatyk WIEiT oraz WEAIiB. W roku 2021/2022 pytania mają zostać zmienione, ale może komuś się przyda.

WDI

1. **Najbardziej prymitywnym systemem liczbowym jest:**
   a. **dwójkowy system liczbowy**
   b. **jedynkowy system liczbowy (X)**
   c. **szesnastkowy system liczbowy**

W systemie jedynkowym piszemy po prostu tyle jedynek, jak duża jest liczba, np. 3 w systemie dziesiętnym to 111 w jedynkowym.

Obliczanie wartości w systemie binarnym

Liczymy osobno część całkowitą i ułamkową (po przecinku).

Dla części całkowitej: dzielimy przez 2, dopisujemy resztę z dzielenia (0 lub 1) z lewej strony budowanej liczby, powtarzamy aż dojdziemy do 0.
Przykładowo:

```
17 | 2 = 8, r. 1    liczba: 1
 8 | 2 = 4, r. 0    liczba: 01
 4 | 2 = 2, r. 0    liczba: 001
 2 | 2 = 1, r. 0    liczba: 0001
 1 | 2 = 0, r. 1    liczba: 10001
 0
```

Dla części ułamkowej: mnożymy przez 2, dopisujemy część całkowitą (0 lub 1) z prawej strony budowanej liczby, jako kolejną liczbę bierzemy ułamek z uzyskanej liczby (w razie czego trzeba będzie odciąć 1), powtarzamy aż dojdziemy do 0.
Przykładowo:

```
0.375 * 2 = 0.75          liczba: 0.0
0.75  * 2 = 1.5 -> 0.5    liczba: 0.01
0.5   * 2 = 1   -> 0      liczba: 0.011
```

Jeżeli w ułamku zauważymy, że coś zaczyna się powtarzać (wpadamy w pętlę z wartościami), to prawdopodobnie mamy ułamek okresowy.

2. **Wartość 5/16 ma postać w systemie binarnym:**
   a. **0.0101 (X)**
   b. **0.1011**
   c. **0.1010**

```
5/16  * 2 = 10/16            liczba: 0.0
10/16 * 2 = 20/16 -> 4/16    liczba: 0.01
0.25  * 2 = 0.5              liczba: 0.010
0.5   * 2 = 1 -> 0           liczba: 0.0101
```

3. **Wartość 1/10 ma postać w systemie binarnym:**
   a. **0.1010**
   b. **0.0(0011) (X)**
   c. **0.(1010)**
   d. **0.0(1010)**

```
0.1 * 2 = 0.2    liczba: 0.0
0.2 * 2 = 0.4    liczba: 0.00
0.4 * 2 = 0.8    liczba: 0.000
0.8 * 2 = 1.6 -> 0.6    liczba: 0.0001
0.6 * 2 = 1.2 -> 0.2    liczba: 0.00011
```

Zauważamy, że wróciliśmy do początku 2 linijki, czyli od wtedy jest okres:
`0.0010... = 0.0(0011)`

<u>Obliczanie ilości informacji</u>

Wykorzystuje się tutaj wzór na entropię informacji Shannona:

$$H(X) = -\sum_{i=1}^{N} p_i \bullet \log_2(p_i)$$

p_i ($p_i>0$, $\Sigma p_i=1$) jest prawdopodobieństwem wystąpienia elementu x_i

Żeby obliczyć na X-znakowego słowa: dla każdego znaku (np. 0/1, a/b/c…) liczymy powyższy wzór, mnożymy przez liczbę miejsc X (zakładając, że dla każdego miejsca jest takie samo prawdopodobieństwo) i potem sumujemy dla poszczególnych znaków.
Przykładowo:
Ile informacji zawiera 8-znakowe słowo, którego każdy znak jest jedną z liter A, B, C? Prawdopodobieństwo wystąpienia litery A wynosi 0.5, natomiast liter B i C po 0.25.

A: $0.5 * \log_2(0.5) = -0.5$
B: $0.25 * \log_2(0.25) = -0.5$
C: $0.25 * \log_2(0.25) = -0.5$

$H(X) = -8 * (-0.5 - 0.5 - 0.5) = -8 * (-1.5) = 12$

4. **Ile informacji zawiera 8-znakowe słowo, którego każdy znak jest jedną z liter a, b? Prawdopodobieństwo pojawienia się (na każdej pozycji) samogłoski jest dwukrotnie większe od prawdopodobieństwa pojawienia się spółgłoski.**
   a. **więcej niż 8 bitów**
   b. **mniej niż 8 bitów (X)**
   c. **dokładnie 8 bitów**

Tutaj ciężko jest zacząć tak jak powyżej, łatwiej jest wrzucić wszystko do wzoru i skorzystać z przekształceń dla logarytmów (zakładając, że nie mamy kalkulatora).

$$H(X) = -8 * [2/3 * \log_2(2/3) + 1/3 * \log_2(1/3)] =$$
$$-8 * [2/3 * \log_2(2) + 2/3 * \log_2(1/3) + 1/3 * \log_2(1/3)] =$$
$$-8 * [2/3 + \log_2(1/3)] = -8 * [2/3 - \log\_2(3)] =$$
$$-8 * [2/3 - \log_2(3)]$$

Ważne jest dla nas tylko przyrównanie tego do 8 (jednostką jest bit). Kluczowe jest, czy wyrażenie w nawiasie ma wartość bezwzględną większą od 1, równą czy mniejszą. Po przybliżeniu otrzymujemy mniej niż 8 bitów.

$$\log\_2(3) \approx 1.5 \rightarrow H(X) \approx -8 * (2/3 - 1.5) = 8 * 5/6 < 8$$

5. **Algorytm to:**
   a. **uporządkowany zbiór operacji (X)**
   b. **specyfikacja ciągu elementarnych operacji, które przekształcają dane wejściowe na wynik (X)**

6. **Język formalny jest:**
   a. **sposobem zapisu algorytmów opartym na blokach operacyjnych**
   b. **sposobem zapisu algorytmów opartym na pseudokodzie (X)**

7. **Syntaktyka języka programowania opisuje:**
   a. **znaczenie instrukcji w języku**
   b. **formalnie poprawne programy**
   c. **budowę instrukcji w języku (X)**
   d. **działanie poprawnego programu**

b. - dyskusyjne, ale uznaję, że fałsz; dowodzenie formalnej poprawności programu polega na przeprowadzeniu typowego matematycznego dowodu, że program robi to, co powinien, czyli jego semantyka zgadza się z tym, czego my jako ludzie żądamy; jeżeli by było "formalnie poprawnie zbudowane", to już byłaby prawda

8. **Semantyka języka programowania opisuje:**
   a. **znaczenie instrukcji w języku (X)**
   b. **formalnie poprawne programy**
   c. **budowę instrukcji w języku**
   d. **działanie poprawnego programu**

Notacja EBNF

Sposób zapisu gramatyki bezkontekstowej. W szczególności jest używana do opisywania składni (syntaktyki) języków programowania. Ważne elementy notacji:

- litery, liczby, ciągi znaków - symbole terminalne (końcowe)
- <> - symbole nieterminalne (pomocnicze)
- ::= - produkcja
- | - alternatywa
- [] - wystąpienie 0 lub 1 raz
- {} - wystąpienie 0 lub więcej razy

9. **Nawiasy <> w notacji EBNF oznaczają:**
   a. **opcjonalne wystąpienie elementu**
   b. **symbol terminalny**
   c. **symbol nieterminalny (X)**
   d. **wielokrotne występowanie elementu**

10. **Semantyka denotacyjna to:**
   a. **opis w postaci funkcji przekształcającej dane wejściowe w dane wyjściowe (X)**
   b. **opis budowy poprawnych semantycznie programów**
   c. **stan maszyny przed i po wykonaniu instrukcji**

Standardy kodowania znaków

ASCII:
- najstarszy format kodowania znaków
- standardowo tylko znaki z języka angielskiego, zapisywane na 7 bitach
- 1 znak ASCII zajmuje 1 bajt, czyli 8 bitów; ostatni bit zgodnie ze standardem nie jest używany (są różne rozszerzenia)
- realnie są to liczby z zakresu 0-127, zapisywane binarnie

Unicode:
- znaki z praktycznie wszystkich języków
- przyporządkowuje każdemu znakowi liczbę - to, jak ta liczba będzie zakodowana, to już inna sprawa
- wstecznie kompatybilne z ASCII - pierwsze 128 znaków to właśnie ASCII
- zmienna długość: 1 bajt dla znaków ASCII, do 4 bajtów dla innych języków
- standardowo używa kodowania UTF-8

UTF-8:
- 8-bit Unicode Transformation Format
- koduje znaki standardu Unicode
- znaki ASCII zapisuje na 8 bitach, późniejsze zestawy na 16, 24 i 32 bitach (czyli 1-4 bajty)
- rozróżnia zestaw znaków po pierwszych bajtach, np. znaki ASCII to 0xxxxxxx, a znaki z drugiej serii to 110xxxxx 10xxxxxx etc.
- jeden znak zgodnie z powyższą zasadą można by zapisać na kilka sposobów, dlatego poprawny jest tylko najkrótszy możliwy zapis danego znaku

11. **Kodem ASCII możemy zakodować:**
   a. **dowolny znak z zakresu 0-128**
   b. **dowolny znak z zakresu 0-255**
   c. **dowolny znak z zakresu 32-255**
   d. **dowolny znak z zakresu 0-127 (X)**

12. **Kodowanie znaków metodą UTF-8 cechuje:**
    a. **pozwala zakodować dowolne znaki Unicode (X)**
    b. **zmienna długość kodu (X)**
    c. **pozwala zakodować dowolne znaki ASCII za pomocą 1 bajta (X)**
    d. **pozwala zakodować dowolne znaki ASCII za pomocą 8 bajtów**
    e. **pozwala zakodować dowolne znaki Unicode za pomocą nie więcej niż 6 bajtów (X)**
    f. **kod znaku ASCII może być częścią dłuższego kodu**

Kodowanie liczb

Ważne są 2 kodowania: liczb stałopozycyjnych (kodowanie U2) i liczb zmiennoprzecinkowych (kodowanie IEEE 754).

Kodowanie U2 dla liczb dodatnich to zwykły zapis binarny, tylko z 0 (bit znaku) na początku. Kiedy chcemy zamienić taką liczbę na ujemną, to zamieniamy znaki wszystkich bitów i dodajemy 1. Jest to wydajne, bo dodawanie i odejmowanie działa identycznie jak dla liczb bez znaku (mniej potrzebnych rozkazów procesora).
W praktyce jest to operacja odjęcia liczby od 2-krotnej wartości najstarszego bitu ($2 * 2^n$), stąd nazwa "kod uzupełnień do 2". Łatwo zauważyć przy tej konstrukcji, kiedy następuje overflow - zmienia się bit znaku.
Ma tylko jedno 0, reprezentowane samymi bitami 0. Jest asymetryczne - jest jedna liczba ujemna więcej, niż jest liczb dodatnich, bo 1 liczba dodatnia jest "zużywana" na 0, bo wymaga ono 0 na bicie znaku. Dla n bitów mamy liczby z zakresu $[-2^{n-1}, 2^{n-1} - 1]$.

Liczby zmiennoprzecinkowe przechowuje się korzystając z notacji Z * 1.M * 2^C, gdzie Z to znak, M to mantysa (część ułamkowa), a C to cecha (część całkowita).
Mantysa typowo jest znormalizowana, czyli z zakresu [1, 2), więc w praktyce przechowuje się tylko część po przecinku. Wyznacza ona, jak precyzyjnie potrafimy przechowywać liczby.
Cecha wyznacza, jak duże / małe potrafimy przechowywać liczby. W szczególności wyznacza też, jak blisko 0 jesteśmy w stanie dojść z naszymi wartościami.
Precyzja reprezentacji zależy od liczby bitów mantysy. 1 miejsce po przecinku wymaga około $\log_2(10)$ bitów, czyli ok. 3.3 bitu. Mając N bitów na mantysę, mamy precyzję do N / 3.3 miejsc po przecinku.

Standard IEEE 754 wyznacza kodowanie floatów (pojedyncza precyzja). Zajmują one 32 bity: 1 na znak, 8 bitów na cechę i 23 bity na mantysę. Znak to 0 dla liczb dodatnich i 1 dla liczb ujemnych.
Cecha jest z zakresu [-127, 128], ale jest przechowywana z nadmiarem (przesunięciem, bias), żeby nie tracić bitu na znak, czyli realnie przechowywany jest zakres [0, 255].
Mantysa zapewnia precyzję około 7-8 miejsc po przecinku, bo 23 / 3.3 ≈ 7.
Niektóre wartości są specjalne i zarezerwowane. Są to np. plus nieskończoność (cecha 128, mantysa 0), NaN (Not a Number, cecha 128, mantysa różna od 0).
W wersji double mamy 11 bitów na cechę, 52 na mantysę i precyzję do ≈16 miejsc.

13. **Od czego zależy dokładność liczb zmiennopozycyjnych w komputerze?**
    a. **od długości cechy**
    b. **od długości mantysy (X)**
    c. **od długości cechy i mantysy**

14. **Cechami kodu uzupełnień do dwóch są:**
    a. **podwójna reprezentacja 0**
    b. **pojedyncza reprezentacja 0 (X)**
    c. **symetryczny zakres liczb**
    d. **asymetryczny zakres liczb (X)**

15. **Liczby stałopozycyjne w komputerze są reprezentowane w kodzie uzupełnień do 2. Dla jakich wartości funkcja abs(x) będzie obliczona prawidłowo?**
    a. **tylko dla liczb ujemnych**
    b. **tylko dla liczb nieujemnych**
    c. **dla wszystkich liczb typu integer**
    d. **wszystkich poza najmniejszą liczbą w reprezentacji (X)**

16. **Dana jest następująca reprezentacja liczb zmiennopozycyjnych: mantysa zajmuje 20 bitów, wykładnik zajmuje 8 bitów. Wykładnik i mantysa zapisywane są w kodzie U2. Przecinek leży na lewo od mantysy. Z dokładnością do ilu cyfr dziesiętnych można pamiętać liczby w tej reprezentacji?**
    a. **około 10**
    b. **około 6 (X)**
    c. **około 3**
    d. **14**
    e. **8**

Ważna jest tylko informacja o mantysie. Mamy 20 bitów, czyli:
$\text{precision} = 20 / \log_2(10) \approx 20 / 3.3 \approx 6$

17. **Dana jest następująca reprezentacja liczb zmiennopozycyjnych: mantysa zajmuje 22 bity, w tym bit znaku, cecha zajmuje 10 bitów, w tym bit znaku. Cecha i mantysa zapisywane są w kodzie U2. Przecinek leży na lewo od mantysy (mantysa jest ułamkiem [1/2...1)). Jaka jest największa możliwa liczba w tym systemie?**
    a. **10^10**
    b. **(1 - 2^(-21)) * 2^(2^9 - 1) (X)**

Cecha po odjęciu bitu na znak ma 9 bitów, a w U2 liczb dodatnich jest o 1 mniej niż ujemnych, czyli największa możliwa wartość cechy to 2^9 - 1. Cała potęga to 2^(2^9 - 1).
Jako że mantysa jest z zakresu [0.5, 1), to największa mantysa będzie minimalnie mniejsza od 1. Mantysa też jest w kodzie U2, czyli mamy do dyspozycji 21 bitów. Najmniejsza możliwa liczba to w związku z tym 2^(-21) - taką mamy "rozdzielczość" liczb, to jest to "minimalnie" mniejsze od 1. W związku z tym największa możliwa mantysa ma wartość 1- 2^(-21).
Sumarycznie mamy zatem (1 - 2^(-21)) * 2^(2^9 - 1).

18. **Jaka jest minimalna liczba bitów, aby reprezentować liczby zmiennopozycyjne z zakresu -106 ... 106 z dokładnością do 2 miejsc znaczących?**
    a. **8 bitów**
    b. **12 bitów (X)**
    c. **4 bity**
    d. **14 bitów**

Trzeba 1 bitu na znak. W części całkowitej (cecha) trzeba zakodować zakres do 256, czyli 4 bity, bo $2 * 2^7 = 256$ (kodowanie jak w IEEE 754).
Co do części ułamkowej wiemy, że precision ≈ N / 3.3, więc N ≈ precision * 3.3, czyli na ułamek potrzebujemy około 2 * 3.3 ≈ 6.6 ≈ 7 bitów.
Mamy zatem minimalnie 12 bitów.

19. **Dana jest następująca reprezentacja liczb zmiennopozycyjnych: mantysa zajmuje 16 bitów, wykładnik zajmuje 8 bitów. Wykładnik i mantysa zapisywane są w kodzie U2. Przecinek leży na lewo od mantysy. Z dokładnością do ilu cyfr dziesiętnych można pamiętać liczby w tej reprezentacji?**
    a. **7-8**
    b. **15-16**
    c. **4-5 (X)**
    d. **2-3**

Ważna jest tylko mantysa. Liczymy ze wzoru dla 16 bitów:
$\text{precision} = 16 / \log_2(10) \approx 16 / 3.3 \approx 4,8$

20. **Liczba 10010 reprezentowana w systemie U2 ma wartość:**
    a. **2**
    b. **-14 (X)**
    c. **14**
    d. **6**

Mamy 1 na pierwszym bicie, czyli jest to liczba ujemna. Żeby dostać faktyczną wartość liczby (wartość dodatnią), zamieniamy wartości bitów i dodajemy 1:
`10010 -> 01101 + 1 -> 01110 = 14`

21. **W standardzie IEEE 754 liczby zmiennopozycyjne podwójnej precyzji:**
    a. **mantysa zawiera 52 bity łącznie z bitem znaku**
    b. **mantysa zawiera 53 bity łącznie z bitem znaku (X)**
    c. **wykładnik zawiera 11 bitów (X)**
    d. **wykładnik zawiera 12 bitów**
    e. **gwarantują 7-8 dziesiętnych miejsc znaczących (X)**

e. - gwarantują nawet więcej, ok. 15, ale zachodzi tutaj logiczna implikacja: jak gwarantuje 15, to gwarantuje też 7-8

22. **Bramki wykonują operacje na:**
    a. **słowach**
    b. **wartościach logicznych (X)**

23. **Bramka NAND wykonuje:**
    a. **najpierw operację iloczynu logicznego, potem negację (X)**
    b. **najpierw operację negacji, potem iloczynu logicznego**
    c. **operację różnicy symetrycznej**

24. **Zbiór przerzutników służących do przechowywania informacji cyfrowej to:**
    a. **procesor**
    b. **rejestr (X)**

Jeszcze: pamięć, bufory.

25. **Do czego służy stos systemowy?**
    a. **do przechowywania wszystkich zmiennych w programie**
    b. **do przechowywania zmiennych alokowanych procedurą new(p)**
    c. **dla przechowywania zmiennych lokalnych procedur i funkcji (X)**
    d. **do przechowywania adresów powrotu z funkcji (X)**
    e. **do przechowywania zmiennych alokowanych dynamicznie**

a., c. - tylko zmiennych lokalnych
b., e. - alokacja dynamiczna za pomocą new odkłada na stercie (heap)

26. **Zmienna typu wskaźnik zajmuje 4 bajty. Ile pamięci można zaadresować takim wskaźnikiem?**
    a. **64 kilobajty**
    b. **4 gigabajty (X)**
    c. **2 megabajty**
    d. **6 gigabajtów**
    e. **8 kilobajtów**

4 B = 32 bity, czyli 2^32 adresów. 1 adres = 1 bajt, czyli mamy 4 GB.

27. **Jaki paradygmat programowania jest realizowany w języku C?**
    a. **aplikatywny**
    b. **imperatywny (X)**
    c. **deklaratywny**
    d. **strukturalny (X)**
    e. **żaden z wymienionych**

28. **Jeżeli w programie następuje odwołanie poza obszar tablicy:**
    a. **zawsze sygnalizowany jest błąd wykonania**
    b. **nie jest sygnalizowany błąd, jeżeli pamięć jest zaalokowana (X)**

Będąc precyzyjnym, w C/C++ jest to undefined behaviour, patrz np. ta dyskusja na SO.

Odwrotna Notacja Polska

Jest to sposób beznawiasowego zapisywania obliczeń, w którym potrzebujemy tylko stosu i 2 rejestrów na operatory, na których wykonujemy operację.
Sposób ewaluacji wyrażeń: widząc liczbę odkładamy na stos, widząc operator ściągamy wartości ze stosu (1 dla unarnego, 2 dla binarnego), aplikujemy i odkładamy wynik na stos. Jeżeli coś się nie będzie zgadzać (np. wyciągniemy operator), to był błąd w wyrażeniu. Kończy się, gdy stos jest pusty - ostatnia zdjęta wartość to wynik.

29. **Których nawiasów trzeba użyć w Odwrotnej Notacji Polskiej do zmiany kolejności wykonywania działań?**
    a. **{ }**
    b. **o kolejności działań nie decydują nawiasy (X)**
    c. **( )**
    d. **[ ]**

30. **Jaki wynik da następujące wyrażenie zapisane w ONP : 2 3 4 5 + * +?**
    a. **25**
    b. **29 (X)**

```
Stos:          Wyrażenie: 2 3 4 5 + * +
Stos: 2        Wyrażenie: 3 4 5 + * +
...
Stos: 2 3 4 5  Wyrażenie: + * +     Działanie: 4 + 5  = 9
Stos: 2 3 9    Wyrażenie: * +       Działanie: 3 * 9  = 27
Stos: 2 27     Wyrażenie: +         Działanie: 2 + 27 = 29

Wynik: 29
```

31. **Co oznacza, że algorytm sortowania tablicy posiada złożoność $O(n^2)$?**
    a. **wykonywana liczba porównań wynosi n^2**
    b. **wykonywana liczba porównań jest rzędu n^2 (X)**
    c. **wykonywana liczba przestawień elementów tablicy w algorytmie wynosi n^2**
    d. **dwukrotne zwiększenie rozmiaru tablicy spowoduje czterokrotne zwiększenie czasu sortowania (X)**

32. **Co oznacza, że algorytm sortowania tablicy posiada złożoność $O(n^2)$?**
    a. **wykonywana liczba porównań i wykonywana liczba przestawień elementów tablicy w algorytmie wynosi n^2**
    b. **wykonywana liczba porównań lub wykonywana liczba przestawień elementów tablicy w algorytmie wynosi n^2**
    c. **żadne z powyższych (X)**

33. **Translacja analizuje symbole:**
    a. **od lewej do prawej strony (X)**

Unix

Prawa dostępu do plików

W systemach Unixowych mamy 3 prawa dostępu do każdego pliku, działające różnie dla zwykłych plików i katalogów:

Uprawnienie	Pliki	Katalogi
r - read	Odczyt zawartości	Odczyt zawartości (wypisanie plików z katalogu)
w - write	Modyfikacja (w tym usunięcie) zawartości	Tworzenie / zmiana nazwy / usuwanie plików z katalogu, modyfikacja atrybutów katalogu
x - execute	Uruchamianie jako programu	Przeszukiwanie, wejście do katalogu (użycie jako katalogu bieżącego), dostęp do plików i katalogów wewnątrz

Uprawnienia są przyznawane z osobna 3 grupom użytkowników, typowo kolejno z coraz mniejszymi uprawnieniami:
- owner - użytkownik, właściciel pliku
- group - grupa-właściciel pliku
- other - wszyscy pozostali

Uprawnienia zapisuje się na 2 sposoby: ciągiem znaków albo 3 liczbami. Ciągiem znaków jest prosto, np. `rw-r-x---`. Dla liczb definiuje się wartości porządkujące każdą trójkę uprawnień r/w/x jako liczbę binarną:
- 4 - read
- 2 - write
- 1 - execute

Liczby te sumuje się dla każdej trójki, a trójka sum (użytkownik, grupa, inni) reprezentuje uprawnienia. Uprawnienia `rw-r-x---` to inaczej 650.

Do powyższych można jeszcze dorzucić 3 rozszerzenia: sticky bit, SUID i SGID. Są one zapisywane binarnie jako dodatkowa trójka liczb, lub zmieniając niektóre litery w ciągu znaków. Jeżeli z nich korzystamy, to mamy dodatkową liczbę w sumach, np. 4750.

Sticky bit ma 2 wersje:
- pliki: ignorowane przez większość dystrybucji
- katalogi: pliki wewnątrz mogą być usuwane / mieć zmienianą nazwę tylko przez właściciela pliku, właściciela katalogu lub roota (normalnie może każdy, kto ma prawo zapisu do katalogu); używa się do zabezpieczenia katalogów wspólnych, np. /tmp

Wyświetla się jako T (nie ustawiony) lub t (ustawiony) na ostatnim miejscu, np. `rw-r-x--t`. Bit jedności w zapisie binarnym uprawnień specjalnych, np. 001. Przykład całej sumy: 1777 (pełne uprawnienia + sticky bit).

SUID (Set User ID upon execution) - program uruchamia się z uprawnieniami jego właściciela, a nie użytkownika uruchamiającego. W praktyce UID procesu jest takie samo, jak UID właściciela. Używane np. przy passwd, ping, traceroute, bo wymagają one uprawnień roota do działania.
Dla katalogów albo jest ignorowane, albo tworzone pliki i katalogi wewnątrz tego katalogu dziedziczą właściciela.
Wyświetla się jako S (nie ustawiony) lub s (ustawiony) na ostatnim miejscu pierwszej trójki, np. `rwsr-x---`. Bit setek w zapisie binarnym uprawnień specjalnych, np. 100. Przykład całej sumy: 4777 (pełne uprawnienia + SUID).

SGID (Set Group ID upon execution) - dla plików jak SUID ale dla właściciela grupowego, dla katalogów pliki będą dziedziczyć właściciela grupowego po katalogu, w którym się znajdują.
Dla katalogów tworzone pliki i katalogi wewnątrz tego katalogu dziedziczą właściciela grupowego.
Wyświetla się jako S (nie ustawiony) lub s (ustawiony) na ostatnim miejscu drugiej trójki, np. `rwxr-s---`. Bit setek w zapisie binarnym uprawnień specjalnych, np. 010. Przykład całej sumy: 2777 (pełne uprawnienia + SGID).

1. **Prawo dostępu do pliku 453 pozwala:**
   a. **właścicielowi czytać plik (X)**
   b. **wszystkim czytać plik**
   c. **właścicielowi na odczyt, grupie na odczyt i uruchomienie, pozostałym na zapis i uruchomienie**
   d. **grupie na odczyt i zapis, pozostałym tylko na odczyt**

`453 = r--r-x-wx`

2. **Przy konfiguracji obsługi sieci w Unixie:**
   a. **plik /etc/hosts przechowuje listę znanych hostów i interfejsów sieciowych (X)**
   b. **plik /etc/hosts zawiera tylko adresy IP lokalnych interfejsów**
   c. **plik /etc/networks zawiera adresy IP znanych sieci IP (X)**
   d. **plik /etc/networks przechowuje listę lokalnych interfejsów**

3. **W systemie operacyjnym UNIX prawdziwe są następujące stwierdzenia dotyczące użytkownika i jego sesji:**
   a. **praca programu logującego (login) jest sterowana m. in. zawartością plików /etc/nologin lub /etc/motd (X)**
   b. **o możliwościach użytkownika w systemie decyduje przynależność do odpowiednich grup użytkowników (X)**
   c. **w zależności od konfiguracji systemu mogą być odnotowywane podłączenia do systemu poprzez zmianę kontekstu użytkownika (komenda su) (X)**
   d. **grupy użytkowników wprowadzono, aby ułatwić zarządzanie użytkownikami oraz podnieść bezpieczeństwo systemu (X)**

a. - /etc/motd to plik "message of the day" wyświetlany przy logowaniu
c. - mogą być też przy okazji blokowane (XKCD)

4. **W systemie operacyjnym UNIX prawdziwe są następujące stwierdzenia dotyczące procesu logowania się i uprawnień użytkowników:**
   a. **o postaci hasła decyduje administrator systemu zapisując ograniczenia w różnych wersjach systemu w różnych plikach konfiguracyjnych (X)**
   b. **wszystkie grupy użytkowników dają im jednakowe uprawnienia**
   c. **cała informacja dotycząca konfiguracji użytkownika w systemie jest zapisana jedynie w pliku /etc/passwd, co zapewnia spójność informacji**
   d. **do systemu użytkownik może podłączać się jedynie z konsoli systemu oraz zdalnie poprzez sieć**
   e. **nazwa użytkownika w systemie UNIX musi być unikalna, zaś numer identyfikacyjny może się powtarzać (X)**
   f. **technika shadow umożliwia podglądnięcie postaci zaszyfrowanej hasła każdego użytkownika w systemie przez dowolnego użytkownika**
   g. **system PAM umożliwia m.in. odnotowywanie nieudanych prób podłączania się do systemu (X)**

a. - np. w katalogu /etc/pam.d/ są pliki dla Debiana
c. - chociażby informacja o grupie użytkownika jest w /etc/group
d. - mamy GUI
f. - tylko przez administratora (sudo)

5. **Zaznacz prawdziwe stwierdzenia na temat procesów:**
   a. **proces jest wykonaniem programu i składa się ze zbiorowości bajtów, w których wyróżnia się instrukcje maszynowe (tzw. tekst), dane i stos (X)**
   b. **w kategoriach praktycznych proces w systemie UNIX jest jednostką utworzoną za pomocą funkcji systemowej fork (z wyjątkiem procesu o numerze 0) (X)**
   c. **każdy uruchomiony w systemie UNIX proces otrzymuje od procesu potomnego dostęp do jego (procesu macierzystego) zmiennych środowiskowych**
   d. **liczba możliwych do uruchomienia w systemie UNIX procesów jest ograniczona jedynie rozmiarem pamięci o dostępie swobodnym danego systemu komputerowego**
   e. **wartość parametru NICE dla nowo utworzonego procesu jest wyliczania na podstawie średniego obciążenia systemu**
   f. **każdy uruchomiony w systemie UNIX proces otrzymuje od systemu trzy otwarte pliki o numerach 0, 1 i 2 (X)**
   g. **rekord w tablicy procesów znajdującej się w jądrze systemu operacyjnego zawiera podstawowe informacje o procesie, w tym jego właścicieli, listę segmentów oraz wskaźniki do U-obszaru (X)**
   h. **zmiana kontekstu z trybu użytkownika na tryb jądra następuje między innymi wówczas, gdy system przydziela procesor kolejnemu procesowi z kolejki gotowych do wykonania procesów**

a. - jakby ktoś chciał być pedantyczny, to jeszcze można wyróżnić segment BSS
c. - na odwrót, potomny otrzymuje od macierzystego, i tylko zmiennych eksportowanych
d. - jest ograniczona liczbą dostępnych PID, można to zmienić w odpowiednim pliku
e. - jest stała i dokładnie zależy od systemu, np. 0 dla BSD
f. - STDIN, STDOUT, STDERR

6. **Interpretery poleceń systemu UNIX:**
   a. **w niektórych interpreterach poleceń można uniemożliwić odłączanie się od systemu poprzez wykorzystanie sekwencji Ctrl-D dzięki ustawieniu wartości odpowiedniej zmiennej środowiskowej (X)**
   b. **wszystkie polecenia shella zwracają wartość 0 (zero), kiedy wykonanie ich zakończy się niepowodzeniem lub wartość różną od 0 (zazwyczaj 1) w przypadku przeciwnym**
   c. **każdy interpreter wykorzystuje trzy podstawowe typy zmiennych: specjalne, środowiska oraz programowe (X)**
   d. **błędne wykonanie komendy w linii komend można rozpoznać m. in. po zapisach do urządzenia /dev/null**

a. - wystarczy ustawić IGNOREEOF na wartość większą niż 1 i wtedy Ctrl-D nie wystarczy (trzeba będzie klepać więcej razy)
b. - na odwrót, 0 gdy jest ok i różna od 0 dla niepowodzenia
d. - /dev/null to strumień systemowy, do którego można zrzucać błędy i wtedy nie da się ich wykryć, dla rozpoznawania ich sprawdza się STDERR

7. **W systemie operacyjnym UNIX:**
   a. **nie ma możliwości odnotowywania nieudanych prób podłączania się do systemu**
   b. **każda komenda posiada inną składnię (X)**
   c. **istnieje możliwość zabronienia podłączania się do systemu wybranemu użytkownikowi z wykorzystaniem protokołu SSH (X)**
   d. **każdy użytkownik posiada unikalną nazwę oraz unikalny numer**

a. - wystarczy skonfigurować PAM
b. - po co komu identyczne komendy?
c. - inaczej byłoby to dość słabe zabezpieczenie
d. - tylko unikalną nazwę

8. **W systemie plików systemu UNIX:**
   a. **system plików składa się z ciągu bloków logicznych zawierających 512, 1024, 2048 lub dowolną inną wielokrotność 512 bajtów (X)**
   b. **system plików przed użyciem należy zamontować, gdyż inaczej zawarte w nim dane nie będą dostępne (X)**
   c. **prawo SUID obowiązuje dla pliku i dla katalogu: w przypadku pliku oznacza, iż program uruchamiany jest z UID właściciela pliku, zaś w przypadku katalogu, że użytkownik pracuje w nim (katalogu) z UID właściciela**
   d. **cofnięcie prawa zapisu do pliku dla jego właściciela uniemożliwia usunięcie go**

b. - po odmontowaniu są niedostępne, przy ponownym zamontowaniu znowu dostępne
c. - dla plików prawda, dla katalogów SUID dotyczy tworzenia nowych plików
d. - żeby usunąć plik, wystarczy mieć prawo modyfikacji katalogu, jest to niezależne od prawa modyfikacji samego pliku

9. **W systemie plików systemu UNIX:**
   a. **stosowany w systemach plików systemu UNIX system kontyngentów (quota) jest zorientowany na użytkownika i obejmuje wszystkie jego pliki w całym drzewie katalogów od katalogu głównego począwszy (X)**
   b. **każdy plik opisuje dokładnie jeden i-węzeł (X)**
   c. **prawo SUID nadane plikowi przechowującemu kod wykonywalny np. komendy passwd umożliwia modyfikację zawartości plików, których właścicielem jest właściciel pliku /bin/passwd (X)**
   d. **i-węzły są przechowywane w systemach plików na dysku oraz kopiowane do tablic jądra systemu w chwili montowania systemu plików**

a. - definiowany dla użytkownika i/lub grupy, obejmuje bloki danych oraz i-węzły, więc siłą rzeczy wszystko
b. - nawet dla pliku występującego pod różnymi nazwami mamy 1 i-węzeł
c. - przy SUID wykonujemy program jak jego właściciel, a właściciel może zawsze modyfikować własne pliki
d. - są tak przechowywane, ale jądro czyta je w chwili operacji na plikach, nie podczas montowania

10. **Szyfrowanie kluczem publicznym w szyfrowaniu asymetrycznym:**
    a. **pozwala jedynie właściwemu odbiorcy odkodować komunikat**
    b. **tylko posiadacz klucza prywatnego może odkodować komunikat (X)**
    c. **wystarczy użyć tego samego klucza do deszyfrowania wiadomości**
    d. **wiadomość zaszyfrowana za pomocą klucza publicznego może być odszyfrowana tylko za pomocą klucza prywatnego (X)**
    e. **tworzy funkcję skrótu w podpisie elektronicznym (X)**
    f. **pozwala każdemu odkodować komunikat**

a., b. - dyskusyjne, czy "właściwy odbiorca" to zawsze posiadacz klucza prywatnego, ale moim zdaniem nie, poza tym są 2 odpowiedzi. "Właściwy" odbiorca to pewna ludzka semantyka, sens który my nadajemy. Z perspektywy komputera (szyfrowania) nie ma czegoś takiego, po prostu kto ma klucz prywatny, ten sobie może odszyfrować.

ASD

1. **Jeżeli elementy w tablicy są posortowane możliwe jest wyszukiwanie:**
   a. **sekwencyjne (X)**
   b. **binarne (X)**
   c. **interpolacyjne (X)**
   d. **żadne z wymienionych**

a. - zawsze można przejrzeć po kolei
b. - dzięki posortowaniu
c. - ulepszone wyszukiwanie binarne, w którym nie skaczemy do połowy przedziału, tylko robimy prostą interpolację liniową dla większej dokładności

2. **Przy rozmieszczaniu algorytmicznym kolizja może wystąpić, gdy:**
   a. **klucze w dwóch rozmieszczanych rekordach są identyczne (X)**
   b. **klucze w dwóch rozmieszczanych rekordach są różne (X)**
   c. **znumeryzowane klucze dwóch rekordów są różne (X)**
   d. **znumeryzowane klucze dwóch rekordów są identyczne (X)**

Konflikt może być zawsze, funkcje haszujące nie są idealne. Poza idealnymi funkcjami haszującymi rzecz jasna, ale one wymagają znajomości z góry możliwych kluczy i odpowiednio dużej ilości pamięci.

3. **Najszybsza metoda sortowania oparta o porównywanie kluczy posiada złożoność:**
   a. **liniową**
   b. **liniowo-logarytmiczną (X)**
   c. **stałą**
   d. **kwadratową**

Patrz twierdzenie w Cormenie, które dowodzi, że dla sortowań porównujących nie da się szybciej. Da się lepiej tylko dla sortowań bez porównywania kluczy między sobą, które coś dodatkowo zakładają, np. kubełkowe z niewielkim zakresem kluczy.

4. **Ekstensywna metoda sortowania to:**
   a. **taka, w której czas sortowania wzrasta wraz ze wzrostem tablicy**
   b. **taka, która wymaga dodatkowej pamięci porównywalnej z rozmiarem sortowanej struktury (X)**
   c. **taka, która wymaga dodatkowej pamięci porównywalnej z rozmiarem pojedynczego rekordu**
   d. **każda szybka metoda sortowania**

Algorytmy w miejscu - niewielka, stała ilość pamięci. Algorytmy ekstensywne - potrzebują dużo dodatkowej pamięci, często za cenę lepszej szybkości.

5. **Stabilną metodą sortowania jest sortowanie:**
   a. **bąbelkowe (X)**
   b. **grzebieniowe**
   c. **kubełkowe (X)**
   d. **pozycyjne (X)**
   e. **przez kopcowanie**
   f. **przez scalanie (X)**
   g. **przez wstawianie (X)**
   h. **przez wybieranie**
   i. **przez zliczanie (X)**
   j. **szybkie**

Formalnie pozostałe też można zaimplementować w wersji stabilnej, ale typowo zwiększając wymagania pamięciowe, więc nie jest to standardem. Najłatwiej zapamiętać te niestabilne:

- grzebieniowe - jest dziwne i poza Garkiem nikt go nie lubi
- kopcowanie - wrzucając rzeczy do kopca, który sam się w środku organizuje, zapominamy o ich względnej kolejności
- przez wybieranie - dokonujemy wymiany wybranego minimalnego elementu z pierwszym (najbardziej po lewej) nieposortowanym, bez uwzględniania kolejności
- szybkie - po prostu przerzucamy mniejsze od pivota na lewo i większe na prawo, bez przejmowania się względną kolejnością (a nawet ją zwykle zmieniamy)

6. **Które metody sortowania mają złożoność O(n)?**
   a. **bąbelkowe**
   b. **grzebieniowe**
   c. **kubełkowe (X)**
   d. **pozycyjne (X)**
   e. **przez kopcowanie**
   f. **przez scalanie**
   g. **przez wstawianie**
   h. **przez wybieranie**
   i. **przez zliczanie (X)**
   j. **szybkie**

7. **Które metody sortowania mają złożoność $O(n^2)$?**
   a. **bąbelkowe (X)**
   b. **grzebieniowe**
   c. **kubełkowe**
   d. **pozycyjne**
   e. **przez kopcowanie**
   f. **przez scalanie**
   g. **przez wstawianie (X)**
   h. **przez wybieranie (X)**
   i. **przez zliczanie**
   j. **szybkie**

8. **Drzewo BST jest drzewem AVL, gdy:**
   a. **dla każdego wierzchołka rozmiary jego poddrzew różnią się co najwyżej o 1**
   b. **dla każdego wierzchołka wysokości jego poddrzew różnią się co najwyżej o 1 (X)**
   c. **dla każdego wierzchołka wysokości jego poddrzew różnią się dokładnie o 1**
   d. **dla każdego wierzchołka rozmiary jego poddrzew różnią się dokładnie o 1**

9. **Drzewo binarne pełne o N poziomach posiada:**
   a. **N węzłów**
   b. **2 * N węzłów**
   c. **2^N węzłów**
   d. **(2^N) - 1 węzłów (X)**

10. **Złożoność wyszukiwania w drzewie AVL jest:**
   a. **liniowa ze względu na liczbę poziomów (X)**
   b. **kwadratowa ze względu na liczbę poziomów**
   c. **liniowa ze względu na liczbę węzłów**
   d. **logarytmiczna ze względu na liczbę węzłów (X)**

11. **Drzewo rozpinające graf zawiera:**
   a. **niektóre wierzchołki grafu**
   b. **wszystkie wierzchołki grafu (X)**
   c. **żadne z wymienionych**

12. **Ile cykli Hamiltona posiada graf pełny o N wierzchołkach?**
   a. **N**
   b. **1**
   c. **0**
   d. **N^2**
   e. **N!**
   f. **(N-1)! / 2 (X)**

Mamy graf pełny, czyli zawsze taki cykl z każdego wierzchołka istnieje. Mamy (n-1) kandydatów na pierwszą krawędź, (n-2) na drugą (bo nie możemy wybrać tej, co poprzednio), (n-3) na trzecią i tak dalej, czyli mamy (n-1)! możliwości. Do tego graf jest nieskierowany, bo nie mamy o tym informacji, więc dzielimy przez 2 dla uwzględnienia symetrii.

13. **Zadanie o rozmiarze *n*, realizowane pewnym algorytmem o złożoności *f(n)*, zostało sprowadzone do dwóch podzadań o rozmiarze *n/2* każde oraz do *n* działań o stałym czasie wykonania, zapewniających rozbicie i scalenie zadania. Złożoność *f(n)* wynosi:**
    a. f(n) = O(log(n))
    *b. f(n) = O(n*log(n)) (X)*
    c. f(n) = O(n + log(n)
    d. f(n) = O(n)

Jest to po prostu merge sort, czyli oczywiście O(n*log(n)).

14. **Dla problemu komiwojażera algorytm pozwalający wyznaczyć rozwiązanie optymalne:**
    a. **istnieje i ma złożoność wielomianową**
    b. **istnieje i ma złożoność wykładniczą (X)**
    c. **nie istnieje**

15. **Dana jest procedura: Proc(n){ if(warunek(x)) then { A(x); Proc(f(n)); B(x) } else C(x) }. Przyjmijmy konwencję, że np. zapis AAABCC oznacza trzykrotne wykonanie instrukcji A, po czym następuje wykonanie instrukcji B, a następnie dwukrotne wykonanie instrukcji C. Następujące sekwencje instrukcji mogą być wynikiem wywołania powyższej procedury:**
    a. **AACBB (X)**
    b. **ACBB**
    c. **AACCBB**
    d. **ACCB**
    e. **C (X)**
    f. **AAACCCBBB**
    g. **AABBC**

Przepisując procedurę na normalny format:

```
Proc(n)
{
  if(warunek(x)) then
  {
    A(x);
    Proc(f(n));
    B(x);
  }
  else
    C(x);
}
```

Czyli mamy A Proc B, albo C. Wynika z tego, że jest to równoznaczne z gramatyką postaci S -> ASB | C. Musimy mieć więc C w środku, a po lewej stronie tyle samo liter A, co po prawej liter B.

Języki programowania

1. **Dana w C, C++, czy Javie umieszczana jest w spójnym obszarze pamięci operacyjnej:**
   a. **tylko dla danych typów predefiniowanych**
   b. **tylko dla danych skalarnych**
   c. **tylko dla danych złożonych**
   d. **i dla danych skalarnych, i dla danych złożonych (X)**

Dane złożone o pamięci spójnej to np. tablica, struct, vector. Jeżeli za "dane złożone" uznaje się też struktury danych, to już tak nie musi być, np. lista w Pythonie (zbiór wskaźników).

2. **W językach imperatywnych (również obiektowych) przy wyborze reprezentacji dla danych rzeczywistych (zmiennoprzecinkowych) pojawiają się problemy:**
   a. **wystąpienia nadmiaru (X)**
   b. **wystąpienia niedomiaru (X)**
   c. **dokładności (X)**

3. **Wskazania (pointers) w C, C++ używane są do reprezentowania (wskazywania):**
   a. **obszarów pamięci operacyjnej (X)**
   b. **zmiennych złożonych (X)**
   c. **zmiennych skalarnych (X)**
   d. **funkcji i metod (X)**

4. **W jaki sposób można obliczyć długość tekstu przekazanego jako argument w poniższej funkcji w języku C?**
   ```
   void foo(const char* txt) { ... }
   ```
   a. `sizeof(txt)`
   b. `strlen(txt)` **(X)**
   c. `txt.length()`
   d. **zliczając, ile znaków występuje w tekście od znaku na który wskazuje wskaźnik do znaku końca łańcuch znaków ('\0') (X)**

a. - fałsz, bo wskaże tylko rozmiar wskaźnika

5. **Co możesz powiedzieć o poniższej deklaracji w języku C?**
   ```
   int t[10] = {1, 2, [4]=1};
   ```
   a. **zgodnie ze standardem C99 spowoduje ona utworzenie tablicy zawierającej 10 elementów, z których 7 ma wartość 0 (X)**
   b. **da ona taki sam efekt, jak deklaracja:** `int t[] = {1, 2, 0, 0, 1}`

Stworzy tablicę z wartościami: `1200100000`.

6. **W jaki sposób obliczyć długość tablicy w funkcji foo() w języku C?**
   `void foo(double t[]) { // dlugosc tablicy t? }`
   a. **nie da się obliczyć (X)**
   b. **po wykonaniu poniższej instrukcji długość tablicy będzie umieszczona w zmiennej len: `int len; for(len=0; t[len]; len++);`**

Nie da się obliczyć, bo chociaż dla samej zmiennej z tablicą używającą [] się da, to przy przekazaniu do funkcji następuje decaying into pointer, a dla wskaźnika już nie mamy żadnej wiedzy.

7. **Która z implementacji funkcji zwracającej tablicę w języku C jest poprawna?**

a. `int[] getTable() { int tab[10]; return tab; }`

b.
```
int * getTable(int n)
   { return (int*) malloc(n * sizeof(int)); } (X)
```

c.
```
int * getTable(int n)
   { return (int*) calloc(n, sizeof(n)); } (X)
```

a. - fałsz, bo tab to zmienna lokalna i po powrocie pamięć zostanie zwolniona.
b., c. - prawda, bo przy ręcznej alokacji pamięci zadziała

Można by jeszcze ewentualnie użyć tablicy statycznej, alokując pamięć w segmencie danych, tablica istnieje wtedy do końca wykonywania programu:

```
int* getTable()
{
  static int tab[10];
  return tab;
}
```

8. **Przeanalizuj poniższą deklarację w języku C:**
   `int (*x)(int, int);`
   a. **zmienna x jest dwuwymiarową tablicą wskaźników typu int* o zmiennym rozmiarze**
   b. **deklaracja jest niezgodna ze składnią języka**
   c. **zmienna x jest wskaźnikiem na funkcję przyjmującą dwa argumenty typu int, zwracającą wartość typu int (X)**

a. - chcąc zrobić tablicę 2D wskaźników `int*`, typem byłoby `int**`, np.:

```
int **arr = malloc(H * sizeof(int*));
```

c. - składnia wskaźników na funkcje:

```
fun_return_type (*fun_name)(arg1_type, arg2_type, ...);
```

9. **Które stwierdzenia dotyczące operatorów w języku C/C++ są poprawne:**
   a. **operatory addytywne mają mniejszy priorytet niż multiplikatywne (X)**
   b. **wyrażenie `z == ++z` jest zawsze fałszywe**
   c. **żadne z pozostałych**

10. **Dzięki konwencji wywołania funkcji w języku C/C++ znanej jako __cdecl możliwa jest implementacja funkcji o zmiennej liczbie argumentów, jak printf(). Które stwierdzenia charakteryzujące funkcje typu __cdecl są prawdziwe?**
    a. **w wygenerowanym kodzie wywołania funkcji argumenty umieszczane są na stosie od końca, dzięki temu na szczycie stosu jest jej pierwszy argument i analizując jego zawartość można określić spodziewaną liczbę argumentów wywołania (X)**
    b. **w języku C kompilator może utworzyć kod wywołania funkcji typu __cdecl nie mając żadnych informacji o typach jej parametrów**

__cdecl to rozszerzenie kompilatorów Microsoftu do C i C++. Oznacza konwencję wewnętrznego umieszczania argumentów w rejestrach i na stosie jak w tradycyjnym C (w kompilatorach Microsoftu to domyślna konwencja), czyli cdecl. Ważne elementy:

- argumenty są umieszczane na stosie od prawej do lewej (od ostatniego)
- to wywołujący usuwa argumenty ze stosu
- powyższe pozwala na realizowanie funkcji ze zmienną liczbą argumentów
- underscore _ jest dodawany na początek nazw argumentów

a. - prawda, bo wywołujący odkłada argumenty na stos i potem je ściąga, więc z góry (w czasie kompilacji) wie, z iloma argumentami została wywołana funkcja
b. - kompilator zawsze zna typy parametrów

11. **W jaki sposób przekazywany jest parametr będący tablicą do funkcji w języku C, np.:**
    `int main(int argc, char* argv[]){ ... }`
    a. **cała zawartość tablicy kopiowana jest na stos i funkcja działa na kopii tablicy**
    b. **na stos przekazywany jest adres pierwszego elementu tablicy (X)**
    c. **tablice są zawsze przekazywane do funkcji jako wskaźnik (X)**
    d. **tablice są zawsze przekazywane do funkcji przez referencję**
    e. **na stosie umieszczany jest pierwszy element tablicy**
    f. **funkcja działa na kopii tablicy, dla której pamięć przydzielona jest na stercie; do funkcji trafia adres tablicy**

W C nie da się przekazywać tablicy przez wartość, i tak jest zamieniana w momencie przekazania jako argument na wskaźnik (decaying into pointer), więc po prostu przekazywany jest wskaźnik, czyli adres pierwszego elementu.

12. **Aby sprawdzić, czy dwa obiekty typu `String` w języku Java mają taką samą zawartość, można:**
    a. **użyć metody `equals()` (X)**
    b. **==**

Inną możliwością jest wykorzystać metodę `compareTo()` i sprawdzić, czy zwraca 0 (identyczna zawartość).
b. - fałsz, bo sprawdzi to referencje, a nie zawartość

13. **Który z poniższych fragmentów kodu w języku Java sprawdza, czy obiekt wskazywany przez referencję xyz należy do klasy XYZ?**
    a. `if (xyz instanceof XYZ)` (X)
    b. `if (xyz.dynamicCastTo(XYZ.class) != null)`
    c. `XYZ.class.isInstance(xyz)` (X)
    d. `XYZ.class.isAssignableFrom(xyz.getClass())` (X)

b. - fałsz, bo to C++, a nie Java

Inne możliwości:
- `if (xyz.getClass() == XYZ.class)`
- `try XYZ c = (XYZ) xyz; catch(ClassCastException e)`
- `try XYZ c = XYZ.class.cast(xyz); catch(ClassCastException e)`

14. **Tablica w języku Java jest zadeklarowana jako:**
    `int tab[] = new int[]{3, 2, 1, 0};`
    Który z fragmentów kodu poprawnie wypisze jej elementy?
    a. `for (int i: tab) System.out.println(tab[i] + " ");`
    b. `for (int i = 0; i < tab.length; i++)`
       `System.out.println(tab[i] + " ");` (X)
    c. `for (int i: tab) System.out.println(i + " ");` (X)

15. **Które zdanie opisujące własności klas w języku Java jest prawdziwe?**
    a. **aby zaznaczyć, że klasa dziedziczy po kilku klasach, należy podać ich listę po słowie kluczowym extends, np.** `class D extends A, B, C {}`
    b. **dla każdej klasy w języku Java możliwe jest zdefiniowanie klasy potomnej**
    c. **klasa może implementować wiele interfejsów (X)**
    d. **w języku Java zawsze bezpośrednio możemy dziedziczyć po jednej klasie (X)**

b. - nie dla klas final

16. **W jaki sposób usuwane są obiekty w języku Java?**
    a. **usuwa się je przez przekazanie referencji do usuwanego obiektu do metody** `System.gc()`
    b. **nie są programowo usuwane, to środowisko wykonawcze podejmuje decyzje czy i kiedy je usunąć (X)**

a. - nie, bo po pierwsze nie przekazuje się referencji przy wywoływaniu garbage collectora (czyli `System.gc()`), po drugie to wywołanie to tylko sugestia dla JVMa i może zostać zignorowane, po trzecie nikt tak nie robi
Obiekty są usuwane automatycznie przez garbage collector po utracie wszystkich referencji na niego, np. po wyjściu z zakresu widoczności (powrót z funkcji) albo nadpisaniu referencji.

17. **Które z poniższych stwierdzeń odnoszące się do klas wewnętrznych i zagnieżdżonych w języku Java są prawdziwe?**
    a. **w metodach klasy zagnieżdżonej (zadeklarowanej z modyfikatorem `static`) jest dostępna referencja `Outer.this`, gdzie `Outer` jest nazwą klasy zewnętrznej**
    b. **obiekt klasy wewnętrznej ma swój stan niezależny od innych obiektów powiązanych z obiektem klasy zewnętrznej (X)**
    c. **klasa wewnętrzna (zadeklarowana bez modyfikatora `static`) nie ma dostępu do prywatnych zmiennych klasy zewnętrznej**
    d. **klasy wewnętrzne muszą dziedziczyć po otaczających je klasach zewnętrznych**

W Javie klasy w klasach dzielą się na 2 rodzaje:
- static - static nested classes, jest to klasa przynależąca do klasy zewnętrznej
- non-static - inner classes, jest to klasa przynależąca do obiektu klasy zewnętrznej

a. - fałsz, bo co prawda samo odwołanie `Outer.this` jest poprawne, ale wskazuje ono na instancję (obiekt) klasy zewnętrznej; w związku z tym klasa zagnieżdżona nie może być static
c. - ma dostęp, jest to wyjątek co do dostępu do prywatnych atrybutów

18. **Jaki typ w Haskellu będzie miało następujące wyrażenie: `r x = x:r x`?**
    a. **`r :: [a]`**
    b. **`r :: Integer a => a -> [a]`**
    c. **`r :: a -> [a]` (X)**

`r x` to definicja funkcji, po prostu `r(x)`. Dwukropek `:` to operator infixowy (w środku wyrażenia, jak np. `+`) oznaczający dopisanie nowego elementu do listy z lewej strony. Powyższa definicja to rekurencyjne dopisanie x do początku listy i wywołanie, z nieskończoną rekurencją. Tworzy to zatem nieskończoną listę jednolitego typu, czyli ma typ:
`r :: a -> [a]`

19. **Jak wygląda poprawna wartość w Haskellu dla typu:**
    `data Tree a = L a | N (Tree a) a (Tree a)`
    a. **`Tree 5 Nil Nil`**
    b. **`N (L 4) 5 (L '4')`**
    c. **`N (L 4) 5 (L 4)` (X)**
    d. **`N (L '4') '5' (L '4')` (X)**

`Tree` to tutaj algebraiczny typ danych (z alternatywą). Drzewo jest homogeniczne (przyjmuje pewien typ `a`) i składa się albo z liścia (`L a` to wartość liścia, a de facto funkcja tworząca liść, składający się z wartości `a`), albo z węzła z wartością i 2 dzieci.
a. - fałsz, bo `Tree` to tylko typ, nie jest częścią wartości. Poza tym nie można mieć jednocześnie 5 (`Int`) i `Nil`, bo mamy tylko jeden typ `a` dla drzewa.
b. - fałsz, bo mamy mieszane typy `4` i `'4'`

20. **Haskell jest językiem opartym o paradygmat:**
    a. **funkcyjny (X)**
    b. **strukturalny**
    c. **imperatywny**
    d. **obiektowy**

21. **Zaznacz prawdziwe zdania odnoszące się do programowania funkcyjnego:**
    a. **w programowaniu funkcyjnym koncepcja funkcji jest taka, jak w algebrze (X)**
    b. **programowanie funkcyjne opiera się na rachunku lambda (X)**
    c. **dobrym nawykiem w programowaniu funkcyjnym jest, aby zmienne były niemutowalne (X)**
    d. **w programowaniu funkcyjnym możemy korzystać jedynie z wbudowanych typów danych**
    e. **funkcyjny styl programowania można uprawiać w ograniczonym zakresie w językach imperatywnych jak C albo JavaScript (X)**
    f. **funkcyjnymi językami programowania są: Erlang, Haskell, C#, Perl**
    g. **można używać funkcyjnego stylu programowania w języku JavaScript (X)**
    h. **niektóre języki imperatywne zostały wyposażone w konstrukcje z języków funkcyjnych (X)**

e., g. - "styl programowania" to np. funkcje czyste (bez efektów ubocznych) albo immutability, które jak najbardziej da się stosować w nie-funkcyjnych językach
f. - C# i Perl zdecydowanie nie
h. - np. list comprehension w Pythonie

22. **Funkcje wyższego rzędu w programowaniu funkcyjnym to:**
    a. **funkcje zwracające inne funkcje jako rezultat obliczeń (X)**
    b. **funkcje, które przyjmują inne funkcje jako parametry (X)**
    c. **potoczne określenie funkcji trudnych w implementacji**

23. **Jaki mechanizm w językach funkcyjnych pozwala na wykonanie operacji na zbiorze danych?**
    a. **iteracja**
    b. **rekurencja (X)**
    c. **funkcje specyficzne dla języka (X)**
    d. **warunkowy skok do etykiety**
    e. **pętla**

Bazy danych

1. **Wskaż, w których przypadkach klauzule instrukcji select są ułożone we właściwej kolejności:**
   a. **from, group by, where, having, order by**
   b. **from, where, group by, having, order by (X)**
   c. **from, group by, having, where, order by**

Kolejność:
- SELECT
- DISTINCT
- FROM
- JOIN, ON
- WHERE
- GROUP BY
- HAVING
- ORDER BY
- TOP / LIMIT / OFFSET

Normalizacja baz danych

Normalizujemy bazy danych, żeby pozbyć się anomalii, zmniejszyć redundancję danych i je zoptymalizować pod względem przechowywania. Wady to trudniejsze pisanie kwerend oraz potencjalnie mniejsza szybkość, bo normalizacja zwiększa liczbę tabel, a więc JOINów.

Ważne pojęcia:
- klucz potencjalny (candidate key) - minimalny zestaw atrybutów (kolumn) relacji, jednoznacznie identyfikujący każdą krotkę (wiersz)
- klucz główny - wybrany klucz potencjalny, którego używamy w praktyce
- atrybut podstawowy - należy do jakiegokolwiek klucza potencjalnego
- atrybut wtórny - nie należy do żadnego klucza
- zależność funkcyjna - relacja X -> Y będąca funkcją, tzn. na podstawie zbioru atrybutów X możemy jednoznacznie określić atrybuty Y; przykładowo, dla tabeli Produkty możemy mieć {NazwiskoDostawcy, Towar} -> Cena
- pełna zależność funkcyjna - taka, że mamy zależność funkcyjną X -> Y tylko dla pełnego X, tzn. to jest minimalny zbiór atrybutów
- częściowa zależność funkcyjna - zbiór atrybutów Y jest częściowo funkcyjnie zależny od X, gdy istnieje zależność funkcyjna X' -> Y, gdzie X' to pewien podzbiór atrybutów X
- przechodnia zależność funkcyjna - taka zależność funkcyjna X -> Y, że istnieje podzbiór atrybutów Z taki, że zachodzi X -> Z i Z -> Y, a nie zachodzi Z -> X ani Y -> Z

Najłatwiej zapamiętać: "The key, the whole key, and nothing but the key, so help me Codd."

Relacja jest w 1NF, jeśli:
- wartości atrybutów są atomowe (niepodzielne), np. nie ma atrybutów-list
- jest zdefiniowany klucz relacji ("the key")
- wszystkie atrybuty niekluczowe są w zależności funkcyjnej od klucza

Relacja jest w 2NF, jeśli:
- żaden atrybut wtórny nie jest częściowo funkcyjnie zależny od żadnego klucza
- alternatywnie: atrybuty wtórne są tylko w pełnej zależności funkcyjnej od kluczy ("the whole key")

Przy okazji z tego wynika, że jeżeli mamy tylko jeden klucz i jest on 1-atrybutowy, to zawsze będziemy mieli 2NF, bo nie ma podzbiorów.

Relacja jest w 3NF, jeśli:
- żaden atrybut niekluczowy nie jest zależny funkcyjnie od innych atrybutów niekluczowych ("nothing but the key")
- innymi słowy: żaden atrybut niekluczowy nie jest przejściowo funkcyjnie zależny od klucza głównego

Mając 3NF, wiersze zależą od klucza (1NF), całego klucza (2NF) i tylko od niego (3NF).

Relacja jest w BCNF, jeśli:
- wszystkie zależności funkcyjne X -> A spełniają warunek, że jeżeli A nie jest zawarte w X, to X jest kluczem lub zawiera klucz
- innymi słowy: dla każdej relacji X -> A, X nie może być atrybutem spoza klucza głównego, jeżeli A jest częścią klucza głównego
- każda relacja, która jest w 3NF i nie ma co najmniej 2 nachodzących na siebie (mających wspólne atrybuty) kluczy kandydujących, jest w BCNF

2. **Dana jest relacja R o schemacie H = {A, B, C, D, E} oraz zbiór zależności funkcyjnych F = {{B, C} -> {D, E}, {C, D} -> {B, E}, {D} -> {C}, {E} -> {B}}. W jakiej maksymalnie postaci normalnej jest relacja R? Zakładamy, że jest w 1NF.**
   a. **1NF**
   b. **2NF**
   c. **3NF (X)**
   d. **BCNF**
   e. **4NF**

Najpierw wyznaczamy klucze kandydujące. Zauważamy, że żadna relacja nie da nam atrybutu A, czyli trzeba go ręcznie dodać. Będą 3:
- jeżeli weźmiemy D, to mamy {D} -> {C}, a mamy wtedy też B i E, bo {C, D} -> {B, E}, czyli klucz to {A, D}
- jeżeli weźmiemy E, to mamy {E} -> {B}, dodajmy jeszcze C i z {B, C} -> {D, E} mamy całość, czyli klucz to {A, C, E}
- jeżeli weźmiemy B i C, to mamy {B, C} -> {D, E}, więc wszystkie atrybuty, czyli klucz to {A, B, C}

Mamy więc klucze kandydujące {A, B, C}, {A, D} oraz {A, C, E}.

Z powyższego wynika, że wszystkie atrybuty są kluczowe, bo każdy należy do jakiegoś klucza potencjalnego. Jako że tylko atrybuty niekluczowe mogą złamać zasady 2NF i 3NF, to mamy 2NF i 3NF.

Nie mamy BCNF, bo dla {B, C} -> {D, E} relacja jest nietrywialna (prawa strona nie jest podzbiorem lewej) i lewa strona nie jest superkluczem.

Ta strona ma automatyczną sprawdzarkę, przydaje się przy takich zadaniach.

3. **Wskaż wszystkie prawdziwe stwierdzenia dotyczące trzeciej postaci normalnej:**
   a. **jeżeli wszystkie atrybuty ze schematu relacji są atrybutami kluczowymi, to relacja jest w 3NF (X)**
   b. **dowolną relację można sprowadzić do 3NF stosując dekompozycję bezstratną i zachowującą zależności funkcyjne (X)**
   c. **3NF oznacza, że każdy atrybut niekluczowy (informacyjny) zależy wyłącznie od klucza; innymi słowy, atrybuty informacyjne są wzajemnie niezależne (X)**
   d. **BCNF jest nieco bardziej restrykcyjną wersją 3NF - w BCNF wszystkie atrybuty (również kluczowe) muszą spełniać warunek zależności wyłącznie od klucza; ten dodatkowy wymóg ma znaczenie, gdy relacja zawiera wiele kluczy (X)**
   e. **jeżeli relacja jest w BCNF, to jest również w 3NF (X)**
   f. **jeżeli relacja jest w 3NF, to możliwe jest występowanie pewnych redundancji (X)**
   g. **relacja jest w 3NF, jeżeli jest w 2NF i nie zawiera zależności funkcyjnych**
   h. **jeżeli relacja jest w 3NF, to jest również w BCNF**

a. - prawda, bo tylko atrybuty niekluczowe mogą spowodować brak 2NF albo 3NF
g. - dyskusyjne; co prawda nie można powiedzieć, że 3NF ⇔ 2NF i brak zależności funkcyjnych, chociaż 2NF i brak zależności funkcyjnych => 3NF

4. **Wskaż wszystkie prawdziwe stwierdzenia dotyczące postaci normalnej Boyce'a-Codda:**
   a. **dowolną relację R o schemacie H można sprowadzić do BCNF stosując dekompozycję bezstratną, ale niekoniecznie zachowującą zależności funkcyjne (X)**
   b. **dowolna relacja dwuatrybutowa jest w BCNF (X)**
   c. **jeżeli schemat relacji znajduje się w postaci normalnej Boyce'a-Codda, to nie ma w nim redundancji (X)**
   d. **BCNF oznacza, że lewa strona każdej nietrywialnej zależności funkcyjnej zawiera klucz (X)**
   e. **atrybut z prawej strony zależności może być podstawowy**
   f. **schematy relacji zawsze należy doprowadzać do postaci BCNF**

b. - prawda, bo dla 2 atrybutów (A1 i A2) mamy możliwości:
- A1 -> A2, czyli A1 jest kluczem, A2 nie jest zawarte w kluczu, więc BCNF jest spełnione
- A2 -> A1, jak wyżej
- brak zależności funkcyjnych - BCNF trywialnie spełnione
- A1 -> A2 oraz A2 -> A1, czyli mamy 2 klucze, A1 i A2 nie są w sobie zawarte, ale że są

kluczami, to BCNF jest spełnione

f. - tylko wtedy, gdy godzimy się z potencjalną utratą pewnych zależności funkcyjnych (i wiemy jakich)

5. **Wskaż wszystkie prawdziwe stwierdzenia dotyczące kluczy obcych w relacyjnym modelu danych:**
   a. **wartości klucza obcego są unikatowe**
   b. **klucze obce są sposobem łączenia przechowywanych danych w różnych tabelach (X)**
   c. **klucz obcy jest kolumną lub grupą kolumn tabeli, która czerpie swoje wartości z tej samej dziedziny, co klucz główny powiązanej z nią tabeli (X)**
   d. **klucz obcy musi odnosić się do istniejącej krotki lub przyjmować wartość NULL, aby jawnie stwierdzić, że nie ma związku z reprezentowanymi obiektami w bazie danych albo że ten związek jest nieznany (X)**
   e. **klucz obcy nie musi być unikatowy w obrębie tabeli (X)**
   f. **klucz obcy może pochodzić z tej samej tabeli, gdy chcemy utworzyć związek rekurencyjny (X)**
   g. **klucz obcy i klucz do którego się on odwołuje muszą mieć tyle samo atrybutów (X)**
   h. **muszą się nazywać tak samo jak klucz tabeli głównej**
   i. **muszą zawierać nazwę tabeli głównej w swojej nazwie**
   j. **powinny wskazywać na klucz główny w tabeli głównej (X)**

a. - tylko klucz główny musi być unikatowy
d., e. - może być NULL albo duplikatem

Własności ACID

ACID to zbiór własności transakcji dla (typowo) relacyjnych baz danych:
- Atomicity (atomowość, niepodzielność) - transakcja to atomowa całość, "wszystko albo nic" - następuje cała albo w ogóle nie jest realizowana.
- Consistency (spójność) - transakcja nie może naruszać zasad integralności systemu, po jej wykonaniu baza danych ma spełniać wszystkie swoje warunki integralności (przekształca stan spójny na stan spójny).
- Isolation (izolacja) - jeśli dwie transakcje są wykonywane współbieżnie, to nie widzą wprowadzanych przez siebie zmian (działają w izolacji).
- Durability (trwałość) - po wykonaniu transakcji system musi prawidłowo uruchamiać się i udostępniać spójne, nienaruszone, aktualne dane zapisane po zatwierdzonych transakcjach, także w wypadku krytycznego błędu (np. nagłej utraty zasilania).

6. **Wskaż wszystkie prawdziwe stwierdzenia dotyczące transakcji:**
   a. **wykonanie instrukcji ROLLBACK powoduje anulowanie transakcji i wycofanie wprowadzonych zmian (X)**
   b. **transakcja jest ciągiem operacji w bazie danych, które należy wykonać wszystkie lub nie wykonywać żadnej z nich (X)**
   c. **równolegle wykonywane transakcje mają ten sam poziom izolacji (X)**
   d. **dane zmodyfikowane przez transakcję, która nie została jeszcze zakończona, nigdy nie są dostępne dla innych równolegle realizowanych transakcji (X)**

a. - standardowe polecenie SQL
b. - Atomicity
c., d. - Isolation

7. **Wskaż, które ograniczenia można definiować na poziomie kolumny (w instrukcji create table).**
   a. **wartość domyślna atrybutu (X)**
   b. **krótsze ścieżki na płycie głównej**
   c. **większy bufor**
   d. **unikalne wartości atrybutu (X)**
   e. **proste klucze główne (X)**
   f. **NULL / NOT NULL (X)**
   g. **wyrażenia regularne**
   h. **funkcje walidujące**
   i. **wyzwalacze typu "zdarzenie, warunek, akcja"**

Ogólnie możliwości:
- NOT NULL
- UNIQUE
- PRIMARY KEY
- DEFAULT
- CHECK
- FOREIGN KEY

8. **Wskaż wszystkie prawdziwe stwierdzenia dotyczące wartości NULL:**
   a. **w modelu relacyjnym wartość NULL jest traktowana jako trzecia, różna od *false* i *true*, wartość logiczna (X)**
   b. **dwie wartości NULL są traktowane jako równe**
   c. **wartości NULL są różne od spacji, zera czy też pustego łańcucha znaków (X)**
   d. **jeżeli wystąpi w wartości atrybutu NULL, to suma po tym atrybucie też jest NULL (X)**
   e. **klucz obcy musi zawierać przynajmniej jedną wartość NULL**
   f. **wartość NULL oznacza zero**

9. **Które wyrażenie SQL zwraca wszystkie wiersze tabeli "Osoby" posortowane malejąco według atrybutu "Imie"?**
   a. **SELECT * FROM Osoby SORT BY 'Imie' DESC**
   b. **SELECT * FROM Osoby ORDER BY Imie DESC (X)**
   c. **SELECT * FROM Osoby SORT 'Imie' DESC**
   d. **SELECT * FROM Osoby ORDER Imie DESC**

10. **Jak możesz zmienić wartość "Kowalski" na "Nowak" w atrybucie "Nazwisko" w tabeli "Osoby"?**
   a. **MODIFY Osoby SET Nazwisko ='Nowak' WHERE Nazwisko ='Kowalski'**
   b. **MODIFY Osoby SET Nazwisko ='Kowalski' INTO Nazwisko ='Nowak'**
   c. **UPDATE Osoby SET Nazwisko ='Kowalski' INTO Nazwisko ='Nowak'**
   d. **UPDATE Osoby SET Nazwisko ='Nowak' WHERE Nazwisko ='Kowalski' (X)**

11. **Jak można przy pomocy SQL uzyskać informację o liczbie wierszy w tabeli "Osoby"?**
   a. **SELECT COUNT(*) FROM Osoby (X)**
   b. **SELECT COUNT() FROM Osoby**
   c. **SELECT COLUMNS() FROM Osoby**
   d. **SELECT COLUMNS(*) FROM Osoby**

12. **Jak można przy pomocy SQL wybrać wszystkie wiersze z tabeli "Osoby", w których wartość atrybutu "Imie" zaczyna się od "a"?**
   a. **SELECT * FROM Osoby WHERE Imie LIKE '%a'**
   b. **SELECT * FROM Osoby WHERE Imie ='%a%'**
   c. **SELECT * FROM Osoby WHERE Imie LIKE 'a%' (X)**
   d. **SELECT * FROM Osoby WHERE Imie ='a'**

13. **Którą z poniższych cech musi posiadać klucz właściwy?**
   a. **jego wartość jednoznacznie wyznacza wiersz w danej tabeli (X)**
   b. **jest automatycznie generowany przez System Zarządzania Bazą Danych**
   c. **nie może być generowany przez SZBD**
   d. **jego wartość składa się tylko z jednego atrybutu**
   e. **nie może być pomniejszony o żaden atrybut (X)**
   f. **żaden podzbiór jego atrybutów nie może być kluczem (X)**

b., c. - może być generowany, ale nie musi
d. - może być złożony
e., f. - bo klucz właściwy musi być minimalny (nie może być superkluczem)

14. **Algebra relacji jest podstawą dla:**
   a. **realizacji przez relacyjny SZBD operacji języka manipulacji danymi (X)**
   b. **budowy diagramów ERD**
   c. **budowy diagramów DFD**
   d. **algorytmicznych języków relacyjnych baz danych (X)**

15. **Które z poniższych cech posiadają widoki (view)?**
    a. **mogą realizować politykę ograniczania dostępu do danych (X)**
    b. **ułatwiają konstrukcję zapytań SQL do bazy danych (X)**
    c. **zabezpieczają dane przed ich utratą w trakcie realizacji złożonych transakcji**
    d. **powodują problemy z konstrukcją zapytań SQL do bazy danych**
    e. **dostarczają prostego mechanizmu uaktualniania danych**

16. **Z jakim problemem trzeba się uporać przy odwzorowaniu w schemacie relacyjnej bazy danych relacji typu n:m (wiele do wiele)?**
    a. **zdefiniować podwójne indeksowanie plików odpowiadających każdej z tabel**
    b. **trzeba zaprojektować dodatkową tabelę (X)**
    c. **dobrać odpowiedni klucz obcy w drugiej z tabel**
    d. **trzeba użyć dodatkowo widoku (view)**

17. **Które z poniższych stwierdzeń są prawdziwe, jeśli dotyczą systemów baz danych realizowanych w architekturze klient-serwer?**
    a. **mocno obciążają sieć w stosunku do systemów scentralizowanych**
    b. **pozwalają klientom pracować w różnych systemach operacyjnych (X)**
    c. **uniezależniają sposób realizacji aplikacji od serwera (X)**
    d. **wymagają co najmniej dwóch komputerów**

a. - kwestia dyskusyjna, bo systemy scentralizowane nie obciążają w ogóle (serwer i klient są tam fizycznie połączone, typowo na 1 maszynie), ale moim zdaniem architektura klient-serwer nie obciąża "mocno" sieci, bo ilość przesyłanych danych jest optymalizowana, np. niektóre elementy SQLa wykonują się już u klienta, na serwerze tylko te pobierające dane i filtrujące (żeby właśnie zmniejszyć obciążenie sieci)
d. - można odpalić na jednym komputerze w 2 procesach

OLTP i OLAP

OLTP - OnLine Transaction Processing, typowe systemy relacyjnych baz danych przechowujące informacje biznesowe. Mają przede wszystkim być szybkie.

OLAP - OnLine Analytical Processing, systemy bazodanowe zbierające informacje z innych baz / źródeł danych i przetwarzające je do formy wygodnej do analizowania. Używane w hurtowniach danych. Mają przede wszystkim dobrze organizować dane do łatwego przetwarzania.

Cecha	OLTP	OLAP
Zadanie	Wspomaganie operacji biznesowych w czasie rzeczywistym	Planowanie, wspomaganie decyzji, analityka, odkrywanie wiedzy
Typowe zapytania	Dużo małych i prostych	Złożone zapytania przetwarzające dużo danych

Czas odpowiedzi	Milisekundy	Sekundy, minuty, godziny - zależy od wolumenu danych
Projekt	Zależny od potrzeb biznesowych	Zależy od domeny i projektu analityki (dla wydajności)
Źródła danych	Transakcje	Zebrane dane z transakcji, baz danych, innych źródeł
Wymagania sprzętowe	Niewielkie, typowo małe bazy (+ archiwizacja starych danych)	Duże, typowo data warehouse
Schemat	Znormalizowane bazy o stałym schemacie	Zdenormalizowane bazy o schemacie dobranym do zadania (np. płatek śniegu)

18. Systemy OLTP i OLAP różnią się przede wszystkim:
- **a. czasem reakcji na zapytanie (X)**
- **b. stopniem szczegółowości przechowywanych danych (X)**
- **c. zakresem realizowanych funkcji użytkowych (X)**

19. Rolą hurtowni danych jest:
- **a. przygotowanie danych do szybkiej analizy (X)**
- **b. kontrola poprawności danych realizowana podczas wykonywania transakcji**
- **c. minimalizacja plików fizycznych przez ujednolicenie sposobu przechowywania danych**
- **d. przechowywanie danych w sposób scentralizowany (X)**
- **e. wydajne odpowiadanie na zapytania o charakterze analitycznym (X)**
- **f. zwiększenie bezpieczeństwa**
- **g. obsługa dużej ilości równoległych transakcji modyfikujących dane**

20. Modele danych w postaci gwiazdy i płatka śniegu:
- **a. ich implementacja przyspiesza analizę danych (X)**
- **b. pozwalają oddać sens złożonej struktury danych (X)**
- **c. zapewniają brak redundancji danych**
- **d. pomagają utrzymać spójność danych**

Systemy operacyjne

1. **System operacyjny jest:**
   a. **zbiorem składników sprzętowych (hardware routines)**
   b. **zbiorem driverów obsługujących urządzenia wejścia-wyjścia (input-output devices)**
   c. **zbiorem składników programowych (software routines) (X)**

Są 3 podstawowe definicje systemu operacyjnego:
- rozszerzona maszyna - ukrywa złożoność architektury (abstrakcja od sprzętu) i dostarcza maszynę rozszerzoną / wirtualną łatwiejszą w programowaniu
- zarządca zasobów - dysponuje zasobami sprzętowymi i programowymi, udostępnia je użytkownikowi i kontroluje dostęp do nich
- program sterujący - nadzoruje działania programów, chroni przed błędami, szereguje zadania do wykonania

2. **Proces transferowania danych, które mają być docelowo wyprowadzone na urządzenie peryferyjne, do przestrzeni pamięci pomocniczej i transferowanie ich na to urządzenie w dogodniejszym czasie nosi nazwę:**
   a. **spooling (X)**
   b. **virtualization**
   c. **caching**
   d. **throttling**

Spooling (simultaneous peripheral operations on-line) - dane dla urządzenia peryferyjnego wrzucamy do bufora, dalej ogarnia je sobie urządzenie peryferyjne, a CPU może się zająć swoją pracą.

3. **Centralny Procesor, po otrzymaniu informacji o przerwaniu z urządzenia wejścia/wyjścia:**
   a. **zatrzymuje się na określony okres czasu**
   b. **przekazuje sterowanie do systemu obsługi przerwań po zakończeniu wykonywania bieżącej instrukcji (X)**

Po otrzymaniu przerwania (dla których system operacyjny musi mieć zaprogramowane wsparcie) używa on zdefiniowanej procedury obsługi dla danego typu przerwania. Wykorzystuje się tutaj wektor przerwań, definiujący akcje dla poszczególnych rodzajów przerwań.

4. **Buforowanie plików realizowane jest w celu:**
   a. **zwiększenia wydajności dostępu do pamięci pomocniczej**
   b. **wspomagania obsługi przerwań**
   c. **zwiększenia wydajności procesora**
   d. **wyrównania różnic prędkości przesyłania danych między różnymi urządzeniami (X)**

Zarządzanie pamięcią

W pamięci pojawia się w praktyce niekorzystne zjawisko fragmentacji. Ma ono 2 rodzaje:
- wewnętrzna - kiedy dzielimy pamięć na kawałki, to nie wykorzystujemy całego, np. chcąc zapisać 1 bajt musimy wziąć cały blok i reszta się marnuje
- zewnętrzna - procesy szatkują pamięć na małe kawałki, alokując ją i zwalniając, przez co mamy tylko małe wolne fragmenty, które ciężko przydzielać, bo na zmienne trzeba pamięci ciągłej

Stronicowanie to właśnie idea podziału pamięci na kawałki, bloki rozmiaru kilka-kilkanaście KB. Jest to podział stały, statyczny. Bloki pamięci fizycznej (to, co siedzi na dysku) to ramki (frames), natomiast fragmenty pamięci procesu to strony (pages).
Adresy fizyczny i logiczny w stronicowaniu są różne, potrzebna jest tablica stron, aby wzajemnie tłumaczyć adresy stron i ramek.
Stronicowanie rozwiązuje problem fragmentacji zewnętrznej, bo zawsze mamy pewne kawałki wolnej pamięci, wielkości co najmniej 1 ramki, a procesy mogą dostawać niezależne kawałki (rozrzucone po pamięci). Następuje to kosztem niewielkiej fragmentacji wewnętrznej, bo musimy za każdym razem zażądać przynajmniej 1 strony.

Segmentacja, w przeciwieństwie do stronicowania, jest oparta na podziale dynamicznym pamięci. Obraz procesu jest dzielony na logiczne części: kod, dane, BSS, stos (ew. wiele stosów). Dla każdej sekcji tworzymy segment. Na potrzeby segmentu przydzielane są dynamiczne partycje w pamięci. Segmenty konkretnego procesu mogą być dowolnie rozmieszczone w pamięci, ale każdy segment zajmuje ciągły obszar pamięci.
Segmenty są dość zróżnicowane rozmiarem czy zawartością, mają natomiast podobne metadane. Do śledzenia segmentów jest wykorzystywana tablica segmentów, za pomocą której można tłumaczyć adresy logiczne w fizyczne. Metadane segmentów to np.:
- adres bazowy (adres początku segmentu w pamięci)
- rozmiar
- atrybuty określające rodzaj zawartości i dostępność, np. kod programu, dane tylko do odczytu, stos etc.
- ID - identyfikuje segment w tablicy segmentów

Kiedy korzystamy ze stronicowania, to możemy w RAMie przechowywać skończoną liczbę stron, czyli trzeba zadecydować, które trzymać. Podstawowe algorytmy:
- FIFO - zastępujemy najwcześniej wczytaną stronę, przy czym nie obchodzi nas to, jak często była ona używana po wczytaniu; rezultatem jest anomalia Belady'ego, czyli zjawisko, w którym po wczytaniu kolejnych stron potrzebnych procesowi brakuje mu ich jeszcze więcej, niż wcześniej (bo wyrzuciliśmy te, z których korzystał)
- optymalny - zastępuje stronę, która najdłużej nie będzie używana; nieużywany, bo jest teoretycznie najlepszy, ale wymagałby wiedzy o przyszłych odniesieniach
- LRU - wymieniamy najdłużej nieużywaną stronę, jest często stosowane; ciężko zaimplementować bezpośrednio, więc używa się przybliżeń, np. bitu odniesienia (1 - było odniesienie do strony, 0 - nie było) lub dodatkowych bitów odniesienia (mamy wektor bitów z historią odniesień)

Pamięć wirtualna to technika łączona ze stronicowaniem. Tworzy ona abstrakcję posiadania przez proces przestrzeni ciągłej, tak logicznie widzi ją programista. Fizycznie jest pofragmentowana i porozrzucana po różnych ramkach w pamięci. Wykorzystanie jej pozwala używać programów o zapotrzebowaniu na pamięć większym, niż dostępna pamięć RAM, bo wykorzystuje dodatkowo obszar wymiany (swap) na dysku jako zapasową pamięć. Odczyt/zapis ze swapa są wolne, dlatego wykorzystuje się technikę leniwej wymiany stron w pamięci, dopiero wtedy, kiedy jest to potrzebne.

5. **Zaznacz prawdziwe stwierdzenia na temat stronicowania:**
   a. **stronicowanie rozwiązuje problem zewnętrznej fragmentacji pamięci (X)**
   b. **tablica stron jest stosowana do translacji adresu logicznego na adres fizyczny (X)**
   c. **podczas stronicowania przyjmuje się, że pamięć fizyczna jest podzielona na strony, a pamięć logiczna na ramki, oraz że rozmiary stron i ramek są jednakowe**
   d. **stronicowanie rozwiązuje problem wewnętrznej fragmentacji pamięci**
   e. **podczas stronicowania przyjmuje się, że pamięć fizyczna jest podzielona na ramki, a pamięć logiczna na strony, i że rozmiary stron i ramek są jednakowe (X)**

6. **Zaznacz prawdziwe stwierdzenia na temat segmentacji:**
   a. **segmentacja ułatwia nadanie częściom procesu odpowiednich atrybutów ochrony: dopuszczalny odczyt, dopuszczalny zapis, dopuszczalne wykonanie (X)**
   b. **zastosowanie segmentacji wyklucza użycie stronicowania**
   c. **mogą istnieć następujące przykładowe rodzaje segmentów: kodu, danych, stosu (X)**
   d. **elementy tablicy segmentów zawierają adresy/numery ramek, w których znajdują się segmenty oraz rozmiary poszczególnych segmentów**

b. - te mechanizmy nie wykluczają się, po prostu segmenty mogą się składać ze stron
d. - zawierają początek segmentu i jego rozmiar, nie wszystkie adresy ramek, bo można korzystać z segmentacji bez stronicowania, wtedy żadne ramki nie istnieją

7. **Zaznacz prawdziwe stwierdzenia na temat algorytmów wywłaszczania stron:**
   a. **zaletą algorytmu zastępowania stron FIFO jest to, że nie zachodzi w nim efekt zwany anomalią Belady'ego**
   b. **w algorytmie zastępowania stron LRU zastępowana jest strona, która najdłużej nie była używana (X)**
   c. **w algorytmie zastępowania stron zwanym algorytmem drugiej szansy algorytm wykorzystuje bit odniesienia, który określa, czy w pewnym przedziale czasu nastąpiło odwołanie do strony (X)**
   d. **zasadą działania algorytmu optymalnego zastępowania stron jest to, że zastąpiona zostaje strona, która najdłużej nie będzie używana (X)**
   e. **w algorytmie zastępowania stron LRU zachodzi efekt zwany anomalią Belady'ego**

f. **optymalny algorytm wywłaszczania stron zapewnia minimalną ilość wywłaszczeń stron przy ustalonej liczbie ramek (X)**
g. **algorytm FIFO wywłaszcza stronę, do której czas dostępu był najdawniejszy**

8. **Zaznacz prawdziwe stwierdzenia dotyczące pamięci wirtualnej:**
   a. **stronicowanie na żądanie jest jednym ze sposobów realizacji pamięci wirtualnej (X)**
   b. **procedura leniwej wymiany (ang. lazy swapper) polega na tym, że nie wykonuje się wymiany stron w pamięci, jeśli nie zachodzi taka potrzeba (X)**
   c. **pamięć wirtualna umożliwia wykonywanie procesów, które nie są w całości przechowywane w pamięci operacyjnej (X)**
   d. **przydział ramek oparty na globalnym algorytmie zastępowania może ograniczyć szamotanie (w porównaniu do algorytmu lokalnego zastępowania), gdyż szamoczący się proces nie doprowadza do szamotania innych procesów**

d. - na odwrót, lokalny algorytm zastępowania (każdy proces wymienia swoje strony osobno) zapewnia taką "izolację" szamotania

9. **Który z problemów rozwiązuje zaproponowany przez Dijkstrę algorytm bankiera:**
   a. **wzajemnego wykluczania (mutual exclusion)**
   b. **wykluczania zakleszczenia (deadlock exclusion)**
   c. **unikania zakleszczenia (deadlock avoidance) (X)**
   d. **unikania wykluczania (exclusion avoidance)**

Algorytm bankiera rozwiązuje problem unikania zakleszczeń przy alokacji zasobów (resource allocation and deadlock avoidance).
a. - obecność w sekcji krytycznej tylko 1 procesu, rozwiązuje go wykorzystanie mutexów, algorytmu piekarni lub algorytm Dekkera
b., d. - w ogóle nie ma czegoś takiego

Można jeszcze wyróżnić zapobieganie zakleszczeniom (deadlock prevention), które rozwiązuje algorytm grafu przydziału zasobów.

10. **Dla uniknięcia błędów uwarunkowanych czasowo, maksymalna liczba procesów, które mogą znajdować się wewnątrz sekcji krytycznej, wynosi:**
    a. **8**
    b. **1 (X)**
    c. **0**
    d. **16**

11. **Inicjalna wartość semafora uogólnionego implementującego sekcję krytyczną wynosi:**
    a. **0**
    b. **1 (X)**
    c. **-1**
    d. **dowolna liczba dodatnia**

Zawsze chcemy co najwyżej 1 wątek w sekcji krytycznej, dlatego też zwykle używa się mutexa / semafora binarnego zamiast uogólnionego.

12. **Zaznacz prawdziwe zdania na temat semaforów:**
    a. **zaletą aktywnego czekania w trybie użytkownika jest brak konieczności kosztownego przejścia do trybu uprzywilejowanego (systemowego) (X)**
    b. **semafor może być użyty do synchronizacji dostępu do sekcji krytycznej (X)**
    c. **aktywne czekanie oznacza ciągłe testowanie wartości wyrażenia do momentu, gdy przyjmie ono wartość, dla której czekanie może być zakończone (X)**
    d. **semafor nie nadaje się do zdeterminowania kolejności wykonywanych operacji w grupie współbieżnych procesów**
    e. **semafor może być użyty do synchronizacji dostępu do sekcji krytycznej, poprzez sygnalizowanie (V) przed wejściem i czekanie (P) po zakończeniu**

a. - zasadniczo jedyna zaleta semaforów wirujących (z aktywnym czekaniem)
b. - główne zastosowanie
c. - przez co marnuje cykle procesora, ale nie zmienia kontekstu
d. - nadaje się, można go np. połączyć z jakąś listą
e. - na odwrót, czekanie (P) przed wejściem i sygnalizowanie (V) po zakończeniu

13. **Semafory:**
    a. **posiadają operacje czekaj (P) i sygnalizuj (V) (X)**
    b. **przed wejściem do sekcji krytycznej wykonywana jest operacja sygnalizuj (V)**
    c. **aktywne czekanie polega na uśpieniu procesu i aktywnym oczekiwaniu na sygnał wybudzający**
    d. **można stworzyć semafor zliczający z 2 semaforów binarnych i zmiennej zliczającej wykorzystującej te semafory (X)**

a. - po holendersku czekanie to "post", sygnalizowanie to "verhoren"
b. - przed P, po wyjściu V
c. - aktywne czekanie to ciągłe sprawdzanie w pętli, opis z odpowiedzi to wersja ulepszona ze zmienną warunkową

Metody numeryczne

1. **Rola metod komputerowych w trójkącie "teoria eksperyment symulacja" to:**
   a. **opracowywanie nowych praw o charakterze podstawowym dla danej dziedziny**
   b. **realizacja symulacji w oparciu o prawa podstawowe, sugerowanie nowych eksperymentów, sugerowanie teorii (X)**
   c. **zastępowanie kosztownych eksperymentów i weryfikowanie teorii**
   d. **realizacja symulacji w oparciu o prawa podstawowe, sugerowanie nowych eksperymentów, sterowanie eksperymentami, sugerowanie teorii**

2. **Dokładność reprezentacji zmiennoprzecinkowej jest określona przez:**
   a. **liczbę bitów mantysy (X)**
   b. **liczbę bitów mantysy i wykładnika**
   c. **zakres wykładnika**
   d. **liczbę bitów mantysy i zakres wykładnika**

3. **Jeśli niewielkie względne zaburzenia danych wejściowych powodują niewielkie względne zmiany wyników, to wówczas:**
   a. **współczynnik uwarunkowania osiąga wysoką wartość**
   b. **współczynnik uwarunkowania osiąga niską wartość (X)**

4. **Wybierz poprawną odpowiedź:**
   a. **uwarunkowanie to cecha wynikająca tylko z dokładności operacji zmiennoprzecinkowych**
   b. **wskaźniki uwarunkowania określają poprawność numeryczną**
   c. **uwarunkowanie i stabilność numeryczna są cechami zadania**
   d. **algorytmy numerycznie poprawne dają rozwiązanie będące nieco zaburzonym dokładnym rozwiązaniem zadania o nieco zaburzonych danych (X)**

a., b., c. - wskaźnik uwarunkowania to cecha danych, poprawność numeryczna to cecha algorytmu

5. **Błędy związane z ograniczeniem nieskończonego ciągu wymaganych obliczeń do skończonej liczby działań nazywamy:**
   a. **błędami zaokrągleń (ang. rounding errors)**
   b. **błędami obcięcia (ang. truncation errors) (X)**

6. **Macierz Hilberta osiąga wysokie wartości współczynnika uwarunkowania (ang. condition number). Na tej podstawie możemy stwierdzić, że:**
   a. **macierz Hilberta jest dobrze uwarunkowana**
   b. **macierz Hilberta jest źle uwarunkowana (X)**
   c. **macierz Hilberta jest zawsze diagonalnie dominująca**

Dobrze uwarunkowany = niski współczynnik uwarunkowania.

Efekt Rungego

Efekt Rungego - pogorszenie jakości interpolacji wielomianowej dla węzłów równoodległych, mimo zwiększenia liczby węzłów N. Na początku wraz ze wzrostem N przybliżenie poprawia się, ale potem zaczyna "rozjeżdżać" się, szczególnie na końcach przedziałów.
Przeciwdziałanie:

- inna funkcja interpolująca, np. interpolacja splajnami (funkcjami sklejanymi)
- wyrzucenie warunku równoodległości: gęstsze upakowanie na końcach przedziału lub wybór na węzły miejsc zerowych wielomianu Czebyszewa N-tego stopnia

7. **Efekt Rungego w interpolacji wielomianowej to:**
   a. **wynik wyboru złej funkcji interpolującej**
   b. **wynik braku jednoznaczności rozwiązania zadania interpolacji**
   c. **konsekwencja arytmetyki zmiennoprzecinkowej**
   d. **wynik błędu metody (X)**

Efekt Rungego wynika z wyboru węzłów równoodległych dla interpolacji wielomianowej. Rozwiązanie jak najbardziej może mieć wielomian określonego stopnia, jest on jednoznaczny, arytmetyka zmiennoprzecinkowa działa, natomiast błędem jest metoda doboru węzłów.

8. **Aby wyeliminować lub znacząco ograniczyć efekt Rungego przy zadaniu interpolacji można:**
   a. **zastosować interpolację funkcjami sklejanymi zamiast metody Lagrange'a (X)**
   b. **zastosować interpolację z węzłami gęściej upakowanymi na końcach przedziału (X)**

9. **Efekt Rungego jest charakterystyczny dla następujących metod interpolacji:**
   a. **interpolacji funkcjami sklejanymi 1 stopnia dla węzłów równoodległych**
   b. **interpolacji funkcjami sklejanymi 3 stopnia dla węzłów równoodległych**
   c. **interpolacji metodą Lagrange'a (wielomianowej) dla węzłów równoodległych (X)**
   d. **interpolacji metodą Lagrange'a dla węzłów będących zerami wielomianów Czebyszewa**

10. **Wskaż zdania prawdziwe dotyczące zagadnienia interpolacji wielomianowej z wykorzystaniem jednomianów (tzw. bazy naturalnej):**
    a. **jest to zadanie dobrze uwarunkowane**
    b. **ma zdecydowanie lepsze właściwości obliczeniowe niż metoda Lagrange'a**
    c. **jest to zadanie źle uwarunkowane (X)**

Interpolacja wielomianowa za pomocą jednomianów (baza naturalną) wykorzystuje macierz Vandermonde'a i rozwiązanie układu równań. Macierz ta jest źle uwarunkowana dla większych n, a jednomiany są do siebie zbyt podobne, oscylują i są niestabilne.

b. - metoda Lagrange'a jest lepsza

11. **Wybierz poprawną odpowiedź:**
    a. **interpolacja z węzłami będącymi zerami wielomianu Czebyszewa odpowiedniego stopnia eliminuje efekt Rungego (X)**
    b. **przybliżenia Pade to technika aproksymacji średniokwadratowej**
    c. **funkcje sklejane nie mogą być wykorzystane do aproksymacji średniokwadratowej ze względu na postać minimalizowanej normy**
    d. **aproksymacja średniokwadratowa polega na minimalizacji normy Czebyszewa**

b. - aproksymacja Pade to technika aproksymacji jednostajnej
c. - mogą być
d. - aproksymacja jednostajna na tym polega

Funkcje sklejane (splajny)

Splajny to sposób definiowania funkcji do zastosowań numerycznych, w których funkcja jest kawałkami wielomianem (piecewise polynomial) i są "sklejone" w węzłach interpolacji. Nakłada się na nie warunki dotyczące tych punktów, tak, aby można było traktować takie "sklejenie" faktycznie jako jedną funkcję. Splajn ma stopień N, definiowany jako największy ze stopni wielomianów, które go tworzą.

Warunki, które muszą spełniać:
- przechodzenie przez węzły interpolacji
- ciągłość (w szczególności w węzłach interpolacji)
- ciągłość pochodnych do N-1 włącznie

Szczególnie ważne i najczęściej wykorzystywane są splajny 3 stopnia (cubic spline), bo dzięki ciągłości 2 pochodnej są gładkie i dobrze interpolują funkcje.
Mają one szczególnie ważną odmianę, natural cubic spline, w której do powyższych warunków dodajemy jeszcze zerowanie się 2 pochodnej na końcowych węzłach (funkcja jest liniowa poza przedziałem [a, b]), tzn. $S''(a) = S''(b) = 0$. Są one najgładszą funkcją interpolującą, a ponadto bardzo efektywnie się je oblicza, bo sprowadza się to do trójprzekątniowego układu równań liniowych.

12. **Funkcje sklejane stopnia m na przedziale [a, b]:**
    a. **są ciągłe wraz z (m-1) pochodnymi na [a, b] (X)**
    b. **są ciągłe wraz z (m+1) pochodnymi na [a, b]**
    c. **są używane tylko do interpolacji przedziałowej**
    d. **nie są przydatne do interpolowania funkcji periodycznych**
    e. **są jednoznacznie określone przez podanie warunków brzegowych**
    f. **nadają się tylko do interpolowania przedziałowego funkcji z nieciągłymi osobliwościami**
    g. **są przydatne nie tylko do interpolacji, ale też do aproksymacji (X)**

c. - przedział [a, b] może być nieskończony, jeśli tylko mamy wzór na obliczanie węzłów interpolacji na całej osi liczb rzeczywistych
e. - zależy od typu interpolacji, np. dla wielomianowej interpolacja dla jednoznaczności wystarczy, że punkty pomiarowe są parami różne

13. **Wielomiany sklejane (ang. spline) trzeciego stopnia muszą spełniać następujące warunki w punktach sklejeń:**
    a. **przechodzenie funkcji interpolującej przez węzły interpolacji (X)**
    b. **ciągłość drugiej pochodnej funkcji interpolującej (X)**
    c. **ciągłość funkcji interpolującej (X)**
    d. **ciągłość pochodnej funkcji interpolującej (X)**

14. **Które z poniżej wymienionych zagadnień numerycznych wykorzystują właściwości przybliżania funkcji wielomianem interpolującym:**
    a. **metoda siecznych, metoda stycznych szukania miejsc zerowych funkcji**
    b. **obliczanie całki oznaczonej funkcji za pomocą kwadratur Newtona-Cotesa (X) (X)**
    c. **równania różniczkowe zwyczajne (X)**
    d. **rozwiązywanie układów równań metodami iteracyjnymi**

b. - najpierw interpolujemy funkcję wielomianowo, a potem całki z wielomianów się bardzo prosto liczy

15. **Dla n+1 wartości zmiennej niezależnej x_i, i = 0,1,...,n, x_(i-1) < x_i, i = 1,2,...,n wykonano pomiary i otrzymano n+1 wartości y_i. Zależność wielkości mierzonej od *x* aproksymowano wielomianem podanym poniżej. Zaznacz prawdziwe implikacje.**

$$W_m(x) = \sum_{j=0}^{m} a_{j,m} x^j$$

    a. **m = n => $E_m = 0$ (X)**
    b. **m > n => $E_m < 0$**
    c. **$E_m > 0$ => n > m (X)**

a., b. - dla m = n wielomian aproksymacyjny to taki sam jak interpolacyjny. W związku z tym przechodzi on przez zadane węzły, czyli błąd to 0
c. - jeżeli mamy więcej punktów niż wynosi stopień wielomianu, to nie może on przejść przez wszystkie punkty i trzeba aproksymować z pewnym błędem (patrz np. regresja liniowa)

16. **Dla n+1 wartości zmiennej niezależnej x_i, i = 0,1,...,n wykonano pomiary i otrzymano n+1 wartości y_i. Zależność wielkości mierzonej od x aproksymowano wielomianem podanym poniżej.**

$$W_m(x) = \sum_{j=0}^{m} a_{j,m} x^j$$

    Rozważamy 3 sposoby obliczania błędu aproksymacji E_m:

1. $E_m = \min_{a_{0,m},a_{1,m},\ldots,a_{m,m}} \sum_{i=0}^{n} |y_i - W_m(x_i)|,$
2. $E_m = \min_{a_{0,m},a_{1,m},\ldots,a_{m,m}} \sum_{i=0}^{n} (y_i - W_m(x_i))^2,$
3. $E_m = \min_{a_{0,m},a_{1,m},\ldots,a_{m,m}} \max_{i=0,\ldots,n} |y_i - W_m(x_i)|$

    Obliczanie współczynników a_i można sprowadzić do zagadnienia liniowego:
    a. **w przypadku 1 (X)**

b. **w przypadku 2 (X)**
c. **w przypadku 3**
d. **w żadnym z tych przypadków**

Zakładając, że "zagadnienie liniowe" to programowanie liniowe, to:
1. Regresja LAD (Least Absolute Deviations), programowanie liniowe to najbardziej typowe rozwiązanie
2. Regresja liniowa (problem najmniejszych kwadratów), da się programowaniem liniowym, chociaż normalnie bardziej opłacalnie jest zwykłym układem równań

17. **Dla tych samych danych eksperymentalnych podanych poniżej wyznaczono 3 funkcje aproksymujące.**

i	0	1	2
x_i	2	4	6
y_i	1	2	1

W każdym przypadku k = 1, 2, 3 funkcja aproksymująca miała taką samą postać f_k(x) = a_k * x + b_k, ale użyto innego kryterium jakości aproksymacji:

1. Dla $k = 1$: $\min_{a_1,b_1} \sum_{i=0}^{2} |y_i - f_1(x_i)|$,
2. Dla $k = 2$: $\min_{a_2,b_2} \sum_{i=0}^{2} (y_i - f_2(x_i))^2$,
3. Dla $k = 3$: $\min_{a_3,b_3} \max_{i=0,1,2} |y_i - f_3(x_i)|$

Zaznacz prawidłowe odpowiedzi:
a. **a_1 = a_2 = a_3, b_1 = b_2 = b_3**
b. **a_1 = a_2 = a_3, b_1 != b_2 != b_3 (X)**

Jest to prosta regresja liniowa, z 3 różnymi funkcjami kosztu. Rozwiązania:
1. f_1(x) = 0 * x + 1 - obliczone z użyciem Python + Statsmodels (LAD regression)
2. f_2(x) = 0 * x + 1.33333... - zwykła regresja liniowa
3. f_3(x) = 0 * x + 1.5 - linia musi przebiegać między punktami

18. **Czy obliczanie parametrów (współczynników) funkcji aproksymującej można sprowadzić do rozwiązania układu równań?**
    a. **tak, ale wtedy i tylko wtedy, gdy funkcja aproksymująca jest funkcją liniową względem zmiennej niezależnej**
    b. **tak, ale wtedy i tylko wtedy, gdy funkcja aproksymująca jest wielomianem (zmiennej niezależnej) (X)**
    c. **nie można**

Obliczanie takie zachodzi dla aproksymacji średniokwadratowej, która ma postać:
$F(x) = a_0 * \varphi_0(x) + a_1 * \varphi_1(x) + \ldots + a_m * \varphi_m(x)$
Baza aproksymacji φ może być różna (wielomiany, sin/cos etc.). Daje to aproksymację liniową, w której dla aproksymacji średniokwadratowej to mamy układ równań.
Jest to tzw. wielomian uogólniony, czyli odpowiedź b. Gdyby była odpowiedź w stylu "kombinacja liniowa", to byłoby bardziej precyzyjne i poprawne.

19. **Do aproksymacji zbioru punktów P = {(x_i, y_i) | i = 0,1,...,n} używamy funkcji $f^{(k)}(x; a_{k,j}|j = 0, 1, \ldots, m)$. a_(k, j) to parametry funkcji. Stosując 3 różne kryteria jakości aproksymacji (miary błędu aproksymacji) podane poniżej otrzymujemy trzy różne funkcje aproksymujące f_k(x), k = 1,2,3 dla tej samej wartości m, a różniące się między sobą wartościami parametrów a_(k, j).**

$$1.\ k = 1: \quad \min_{a_{1,0},\ldots,a_{1,m}} \sum_{i=0}^{n} |y_i - f^{(1)}(x_i)|,$$

$$2.\ k = 2: \quad \min_{a_{2,0},\ldots,a_{2,m}} \sum_{i=0}^{n} \left(y_i - f^{(2)}(x_i)\right)^2,$$

$$3.\ k = 3: \quad \min_{a_{3,0},\ldots,a_{3,m}} \max_{i=0,1,\ldots,n} |y_i - f^{(3)}(x_i)|$$

Niech $\Delta_{\max}^{(k)}$ oznacza odległość (w sensie metryki maksimum) k-tej funkcji aproksymującej f_k od najbardziej oddalonego punktu ze zbioru P, tzn. $\Delta_{\max}^{(k)} = \max_{i=0,\ldots,n} |y_i - f^{(k)}(x_i)|$. Zaznacz prawdziwe relacje:

a. $\Delta_{\max}^{(1)} \geq \Delta_{\max}^{(2)}$ **(X)**
b. $\Delta_{\max}^{(1)} \leq \Delta_{\max}^{(2)}$
c. $\Delta_{\max}^{(3)} \leq \Delta_{\max}^{(2)}$ **(X)**
d. $\Delta_{\max}^{(3)} \leq \Delta_{\max}^{(1)}$ **(X)**

Aproksymacja jednostajna (przykład 3) minimalizuje największą odległość, czyli zadane kryterium, czyli Δ_3 jest najmniejsza.
Kwadratowa funkcja błędu (przykład 2) zwraca większą uwagę na outliery (punkty leżące daleko, o dużym błędzie aproksymacji) niż funkcja bezwzględna (przykład 1), więc mocniej się odchyli w ich stronę, zmniejszając maksymalną odległość, więc Δ_2 <= Δ_1.
Finalnie: Δ_3 <= Δ_2 <= Δ_1.

20. **Kwadratury Gaussa stosowane są:**
    a. **ze względu na łatwiejszy wybór węzłów całkowania**
    b. **dlatego, że kwadratury Newtona-Cotesa nie dają możliwości usuwania całkowalnych osobliwości**
    c. **ze względu na mniejszą złożoność obliczeniową niż kwadratur Newtona-Cotesa**
    d. **ponieważ umożliwiają uzyskanie prawie dwa razy większego stopnia dokładności niż kwadratury Newtona-Cotesa dla tej samej liczby węzłów (X)**

a. - jest cięższy
b. - dają, to kwadratury Gaussa mają z nimi problem na końcach przedziału

21. **Do całkowania numerycznego używa się m. in. kwadratur Newtona-Cotesa. Do prostych kwadratur Newtona-Cotesa należą:**
    a. **metoda Eulera**
    b. **metoda Romberga**
    c. **metoda prostokątów (X)**
    d. **metoda trapezów (X)**
    e. **metoda Simpsona (X)**
    f. **reguła 3/8 (X)**
    g. **metoda Boole'a (X)**

Proste kwadratury (węzły całkowania równoodległe): prostokątów, trapezów, Simpsona, reguła 3/8, Boole'a.
Metoda Romberga jest dyskusyjna, ale uznaję ją za coś innego niż "prosta kwadratura Newtona-Cotesa", bo chociaż jest oparta na metodzie Newtona-Cotesa (węzły równoodległe), to wykorzystuje ekstrapolację Richardsona i jest zbyt inna od pozostałych metod.

22. **Wybierz właściwe uporządkowanie metod całkowania Monte Carlo według narastającej ich jakości:**
    a. **podstawowa, warstwowa, średniej ważonej, orzeł-reszka**
    b. **podstawowa, orzeł-reszka, warstwowa, średniej ważonej**
    c. **podstawowa, średniej ważonej, orzeł-reszka, warstwowa**
    d. **orzeł-reszka, podstawowa, warstwowa, średniej ważonej (X)**

Metody rozwiązywania równań nieliniowych

Metoda bisekcji:
- najstarsza i najprostsza
- zbieżność liniowa
- oparte na twierdzeniu Darboux
- tnie przedział na pół i idzie w stronę mniejszych wartości
- wymagania:
    - funkcja ciągła na przedziale [a, b]
    - różne znaki na końcach przedziału

Metoda stycznych:
- inne nazwy: metoda Newtona, metoda Newtona-Raphsona
- zbieżność kwadratowa
- wykorzystuje pochodną funkcji
- aktualizacja iteracyjna: `x = x - f(x) / f'(x)`
- wymagania:
    - dokładnie 1 pierwiastek w przedziale [a, b]
    - różne znaki na końcach przedziału
    - 1 i 2 pochodna mają stały znak na przedziale [a, b]

Metoda siecznych:

- inna nazwa: metoda Eulera
- zbieżność nadliniowa - ok. 1,6
- wykorzystuje przybliżenie pochodnej różnicami skończonymi:
  $f'(x_{n-1}) \approx f(x_{n-1}) - f(x_{n-2}) / (x_{n-1} - x_{n-2})$
- nie zawsze jest zbieżna
- wymagania:
  - funkcja ciągła na przedziale [a, b]

Wszystkie te metody mogą zostać uogólnione na układy równań nieliniowych.

23. **Warunkami wystarczającymi, gwarantującymi zbieżność poszukiwania miejsc zerowych funkcji f(x) metodą bisekcji są:**
    a. **funkcja f(x) jest ciągła w przedziale domkniętym [a, b] (X)**
    b. **pierwsza i druga pochodna f(x) istnieją i są ciągłe w przedziale domkniętym [a, b]**
    c. **pierwsza i druga pochodna mają stały znak w całym przedziale**
    d. **na końcach przedziału [a, b] wartości funkcji f(x) przyjmują przeciwne znaki, czyli zachodzi f(a) * f(b) < 0 (X)**

Warunek wystarczający metody bisekcji składa się z 2 części, zakładam, że trzeba w tym pytaniu zaznaczyć wszystkie odpowiedzi dające w sumie ten warunek.

24. **Stosując algorytm stycznych poszukiwania jednokrotnego miejsca zerowego funkcji f(x) w przedziale domkniętym [a, b] w dostatecznej bliskości pierwiastka uzyskujemy zbieżność:**
    a. **kwadratową (X)**
    b. **wykładniczą**

25. **Metoda Newtona-Raphsona rozwiązywania równań nieliniowych:**
    a. **wykorzystuje niejawnie przyspieszenie Aitkena**
    b. **jest przykładem metody iteracyjnej o stałym punkcie (X)**
    c. **jest przykładem metody interpolacyjnej**
    d. **ma rząd zbieżności równy 1**
    e. **ma rząd zbieżności równy 2 (X)**
    f. **jest szybciej zbieżna niż metoda siecznych (X)**
    g. **jest szybciej zbieżna niż metoda bisekcji (X)**
    h. **nie może być uogólniona dla układów równań nieliniowych**
    i. **wymaga znajomości pochodnej funkcji (X)**

26. **Metody blokowe rozwiązywania układów równań nieliniowych:**
    a. **nie wymagają wyznaczania elementu wiodącego**
    b. **są jedynymi metodami dostępnymi na architekturach równoległych**
    c. **redukują złożoność obliczeniową metodą "dziel i rządź" (X)**
    d. **zawsze automatycznie konwertują macierz do postaci trójkątnej dolnej, co upraszcza dalsze operacje**

Układy równań liniowych

Rozwiązywanie układów równań liniowych można podzielić na 3 grupy metod: dokładne i iteracyjne.

Metody dokładne:
- metoda Gaussa, dekompozycja LU
- złożoność obliczeniowa $O(n^3)$, n - liczba równań
- dobrze nadają się do macierzy gęstych
- daje się je zawsze zastosować
- dają zawsze tak samo dokładne rozwiązanie, przy czym metoda LU jest dokładniejsza numerycznie od metody Gaussa
- wykorzystują pivoting, czyli zamianę wierszy, aby element wiodący był jak największy
- metoda Gaussa przekształca macierz do trójkątnej górnej i podstawieniami rozwiązuje równania
- metoda LU dekomponuje macierz A na iloczyn macierzy dolnej trójkątnej L i górnej trójkątnej U
- przy wielokrotnym rozwiązywaniu układu Ax=b dla stałego A metoda LU jest szybsza, bo drugie i kolejne równanie rozwiązuje się w czasie $O(n^2)$

Metody iteracyjne:
- Jacobiego, Gaussa-Seidla, SR, SOR, Czebyszewa
- iteracyjnie ulepszamy przybliżenie rozwiązania
- złożoność zależy od liczby iteracji
- dokładność zależy od metody, kolejność powyżej to rosnąca jakość rozwiązania
- dzielą się na stacjonarne (powyżej, szukają stałego wektora x) oraz niestacjonarne
- wymagają twierdzenia o zbieżności dla danego rodzaju macierzy, najogólniejsze to: proces iteracyjny jest zbieżny, gdy promień spektralny (największa co do modułu wartość własna) macierzy iteracji A jest < 1

27. Metoda eliminacji Gaussa rozwiązywania układów równań liniowych:
   a. wymaga wyszukiwania elementu wiodącego i polega na przekształceniu macierzy do postaci trójkątnej górnej (X)
   b. polega na doprowadzeniu macierzy do postaci diagonalnej
   c. ma złożoność $O(n^3)$, gdzie n - liczba równań (X)
   d. jest przykładem metody iteracyjnej niestacjonarnej
   e. ma złożoność $O(n^2)$, gdzie n - liczba równań

28. Które zdania dotyczące metody eliminacji Gaussa rozwiązywania układów równań są prawdziwe:
   a. jest to metoda iteracyjna
   b. jest to metoda dokładna (X)
   c. przekształca macierz do postaci macierzy schodkowej (X)
   d. nie wymaga przekształcenia układu równań do postaci z macierzą trójkątną górną

d. - po konsultacji z prof. Paszyńskim wniosek jest taki, że klasyczna eliminacja Gaussa, o którą prawdopodobnie chodzi autorowi zadania, wymaga przejścia do macierzy trójkątnej górnej, po czym wykonuje się backward substitution

29. **Metody dekompozycji LU:**
    a. **mają znacznie lepsze własności numeryczne niż metoda eliminacji Gaussa (X)**
    b. **należą do rodziny metod iteracyjnych niestacjonarnych**
    c. **nie wymagają wyszukiwania elementu wiodącego**
    d. **są przydatne tylko do rozwiązywania układów równań z macierzą symetryczną dodatnio określoną**
    e. **mają lepszą złożoność obliczeniową niż metoda Gaussa, gdyż nie wymagają wyszukiwania elementu wiodącego**
    f. **są metodami iteracyjnymi stacjonarnymi**

c. - wymagają, można stosować też pivoting
d. - niektóre rodzaje metod iteracyjnych (np. steepest descent) tego wymagają, ale LU nie

30. **Do metod nazywanych metodami dokładnymi rozwiązywania układów równań liniowych zalicza się:**
    a. **metoda Jacobiego**
    b. **metoda rozkładu LU (X)**
    c. **eliminacja Gaussa (X)**
    d. **eliminacja Jordana (X)**
    e. **eliminacja Gaussa z wyborem elementu głównego (X)**
    f. **metoda Cramera (X)**
    g. **metoda SOR (Successive Over-Relaxation)**

31. **Warunkiem koniecznym i wystarczającym zbieżności metod iteracyjnych prostych (takich jak metoda Jacobiego czy metoda Gaussa-Seidla) rozwiązywania układów równań liniowych jest:**
    a. **promień spektralny macierzy iterowanej w danej metodzie jest zawsze mniejszy od 1 (X)**
    b. **promień spektralny macierzy iterowanej w danej metodzie jest zawsze większy od 1**

32. **Wybierz właściwe uporządkowanie metod iteracyjnego rozwiązywania układów równań liniowych według narastającej ich jakości:**
    a. **Jacobiego, Czebyszewa, nadrelaksacji, Gaussa-Seidla**
    b. **SOR, Gaussa-Seidla, Czebyszewa, Jacobiego**
    c. **Jacobiego, nadrelaksacji, Czebyszewa, SR**
    d. **Jacobiego, SR, SOR, Czebyszewa (X)**

33. **Szybka transformata Fouriera:**
    a. **jest możliwa tylko wtedy, gdy liczba punktów jest całkowitą potęgą 2**
    b. **ma złożoność obliczeniową O(n), gdzie n - liczba węzłów**
    c. **jest realizacją wzoru całkowego Fouriera na architekturach równoległych**
    d. **polega na zamianie pojedynczej transformaty Fouriera na sumę transformat Fouriera (X)**

a. - wtedy jest najefektywniej, ale da się dla innych sytuacji
b. - złożoność O(n*log(n))

34. **Algorytmy optymalizacji statycznej:**
    a. **metoda simplexu Neldera-Meada jest metodą bezgradientową (X)**
    b. **metoda Newtona wymaga obliczania w każdym kroku gradientu i hesjanu (X)**
    c. **metody z funkcją kary (penalty methods) stosuje się w przypadkach optymalizacji bez ograniczeń**
    d. **metoda najszybszego spadku wymaga obliczania hesjanu**

35. **Wybierz poprawne zdania opisujące metody minimalizacji:**
    a. **wzór Davidona-Fletchera-Powella jest podstawą metody zmiennej metryki (X)**
    b. **metoda sprzężonych kierunków jest przykładem ogólnej metody poszukiwania minimum globalnego**
    c. **metoda największego spadku jest przykładem metody poszukiwania minimum globalnego**
    d. **metoda simpleksów należy do metod gradientowych**

b., c. - minimum lokalnego

36. **Metoda symulowanego wyżarzania:**
    a. **to rodzina heurystycznych technik optymalizacji opartych na analogii z fizyką statystyczną układów losowych (X)**
    b. **to jedna z technik optymalizacji kombinatorycznej wykorzystująca metodę Newtona**
    c. **to specjalne sformułowanie problemu komiwojażera**
    d. **to specjalna odmiana algorytmu Metropolisa**

b. - może być optymalizacją kombinatoryczną, ale nie wykorzystuje metody Newtona
c. - może być rozwiązaniem problemu komiwojażera
d. - nie ma z nim nic wspólnego, algorytm Metropolisa-Hastingsa służy do próbkowania metodami Markov Chain Monte Carlo (MCMC)

37. Rozważmy funkcję kwadratową n zmiennych, f: R^n -> R:

$$f(\mathbf{x}) = \mathbf{x}^\mathsf{T} A\mathbf{x} + \mathbf{b}^\mathsf{T}\mathbf{x} + c$$

A jest macierzą n × n, b wektorem n × 1 o stałych współczynnikach, c jest skalarem. Załóżmy, że macierz A jest dodatnio określona. Funkcja f ma minimum w punkcie x_{min}.
Rozważmy tylko 3 metody szukania minimum tej funkcji: simpleksu Neldera-Meada, najszybszego spadku (steepest descent) oraz Newtona. Startujemy z dowolnego punktu x_0, x_0 != x_{min}.

a. metoda Newtona gwarantuje znalezienie minimum funkcji f w pierwszym kroku (X)
b. metoda najszybszego spadku gwarantuje znalezienie minimum funkcji f w pierwszym kroku
c. metoda simpleksu Neldera-Meada gwarantuje znalezienie minimum funkcji f w pierwszym kroku
d. żadne z pozostałych

Ważne jest, że x^TAx to macierz dodatnio określona, a całe wyrażenie to po prostu wielowymiarowa funkcja kwadratowa, czyli funkcja jest wypukła i ma unikalne minimum globalne. W związku z tym każda z tych metod w tym konkretnym przypadku w pewnej liczbie kroków osiągnie minimum globalne.
Metoda Newtona aproksymuje funkcję jako kwadratową, zatem ona osiągnie minimum już w pierwszym kroku.

38. Podejście wariacyjne do rozwiązywania równań różniczkowych:
a. nie nadaje się do obliczeń numerycznych
b. polega na tym, że szukamy funkcji minimalizującej odpowiedni funkcjonał (X)
c. polega na wprowadzeniu specjalnej siatki całkowania
d. polega na uzmiennieniu położeń części punktów siatki

39. Wymagania stawiane schematom rozwiązywania numerycznego zagadnienia początkowego to:
a. jawne uwzględnienie warunków brzegowych
b. wprowadzenie siatki przestrzennej zgodnej z siatką czasowa
c. zgodność aproksymacji różnicowej, dokładność, stabilność, efektywność czasowa i pamięciowa (X)
d. algorytm powinien mieć względnie małe wskaźniki uwarunkowania
e. współczynnik propagacji błędu < 1, zgodność aproksymacji różnicowej (X)
f. wskaźniki uwarunkowania powinny mieć takie same wartości w każdej iteracji algorytmu

40. Metoda Cranka-Nicolsona to:
a. technika poszukiwania minimum lokalnego z uwzględnieniem drugich pochodnych
b. numeryczny sposób rozwiązywania równań różniczkowych parabolicznych (X)
c. technika wyszukiwania miejsc zerowych wielomianów
d. numeryczny sposób rozwiązywania równań różniczkowych eliptycznych

Teoria automatów i języków formalnych

1. **Wybierz, które z poniższych stwierdzeń są prawdziwe:**
   a. **autorem klasyfikacji (hierarchii) języków formalnych jest Noam Chomsky (X)**
   b. **zbiór wszystkich palindromów nad danym alfabetem jest językiem regularnym**
   c. **istnieją języki, które nie należą do hierarchii Chomsky'ego, więc nie są generowane przez żadną gramatykę ani akceptowane przez żaden automat (X)**
   d. **językiem formalnym nazywamy dowolny podzbiór zbioru wszystkich możliwych słów zbudowanych z symboli należących do skończonego alfabetu (X)**
   e. **język oznaczany jako {ε} jest językiem pustym**
   f. **najszerszą klasą języków w hierarchii Chomsky'ego jest klasa języków rekurencyjnie przeliczalnych generowanych przez gramatyki bez ograniczeń (X)**
   g. **jeżeli L jest językiem bezkontekstowym, to istnieje automat ze stosem akceptujący ten język i jest on mu równoważny (X)**

b. - jest językiem bezkontekstowym
c. - są to języki naturalne, a hierarchia Chomsky'ego obejmuje wszystkie języki formalne
e. - nie, bo to język składający się z jednego słowa pustego, pusty byłby ∅

2. **Niech N będzie niedeterministycznym automatem skończonym posiadającym n stanów, oraz niech M będzie minimalnym deterministycznym automatem skończonym rozpoznającym ten sam język, posiadającym m stanów. Wtedy:**
   a. **m <= n**
   b. **m <= 2^n (X)**
   c. **n <= m**
   d. **M posiada dokładnie jeden stan akceptujący**
   e. **każdy deterministyczny automat skończony rozpoznający ten sam język musi posiadać co najmniej 2^n stanów**
   f. **każdy automat deterministyczny akceptujący ten sam język co N musi mieć co najwyżej 2^n stanów**

f. - formalnie można dodać dowolną liczbę bezsensownych, a nawet nieosiągalnych stanów, ale w razie potrzeby odpowiedź f może być zamiast b, gdyby b nie było wśród odpowiedzi (zawsze przynajmniej 1 jest poprawna)

3. **Wybierz, które z poniższych stwierdzeń są prawdziwe:**
   a. **jeśli język L spełnia tezę lematu o pompowaniu dla języków regularnych, to język L jest regularny**
   b. **jeśli język L jest skończony, to L* musi być regularny (X)**
   c. **każdy język, który nie jest regularny, jest nieskończony (X)**
   d. **pomimo, że języki regularne w trywialny sposób są bezkontekstowe, to istnieją języki regularne nie spełniające lematu o pompowaniu dla języków bezkontekstowych**

e. **żeby sprawdzić, czy dany automat akceptuje język nieskończony, można iterując po wszystkich słowach złożonych z alfabetu sprawdzić, czy liczba słów przez niego akceptowanych jest nieskończona**
f. **zupełny deterministyczny automat skończony, który nie akceptuje żadnego słowa, nie ma osiągalnych stanów akceptujących (X)**

a. - lemat o pompowaniu to warunek konieczny, a nie wystarczający
b., c. - bo każdy skończony jest regularny, do tego języki regularne są zamknięte ze względu na gwiazdkę Kleene'a
e. - pętla nieskończona
f. - gdyby miał, to akceptowałby minimum słowo puste

4. **Dla języków i gramatyk formalnych, odnośnie postaci normalnej Chomsky'ego oraz postaci normalnej Greibach można sformułować następujące stwierdzenia (duże litery alfabetu łacińskiego to symbole nieterminalne, a litery małe to symbole terminalne):**
   a. **gramatyka w postaci Chomsky'ego zawiera produkcje postaci A -> BC, A -> a, a gramatyka w postaci Greibach zawiera produkcje postaci A -> aX (gdzie X to ciąg symboli nieterminalnych, może być pusty), oraz każdą gramatykę bezkontekstową w postaci normalnej Chomsky'ego można przekształcić do postaci normalnej Greibach (X)**
   b. **dla dowolnej gramatyki bezkontekstowej G istnieje taka gramatyka bezkontekstowa G' będąca w postaci normalnej Chomsky'ego, że L(G') = L(G) - {ε} (X)**
   c. **dla każdego języka bezkontekstowego istnieje gramatyka w postaci normalnej Chomsky'ego**
   d. **każdą gramatykę bezkontekstową można przekształcić do postaci normalnej Chomsky'ego**
   e. **każdą gramatykę bezkontekstową można przekształcić do postaci normalnej Greibach**

Postać normalna Chomsky'ego:
- produkcje postaci A -> BC lub A -> a
- na wejście dostaje gramatykę bez ε-produkcji, bez produkcji łańcuchowych, bez symbolu pustego (bez produkcji S -> ε)
- język generowany przez gramatykę nie ulega zmianie

Postać normalna Greibach:
- produkcje postaci A -> aX, gdzie X to ciąg 0, 1 lub więcej symboli nieterminalnych
- na wejście dostaje gramatykę w postaci normalnej Chomsky'ego
- język generowany przez gramatykę nie ulega zmianie

a. - prawda, bo podane definicje postaci normalnych są prawidłowe, a możliwość przekształcenia Chomsky -> Greibach wynika z samego faktu, że postać normalna Greibach dostaje na wejście gramatykę w postaci normalnej Chomsky'ego

b. - tak, bo dowolną gramatykę można przekształcić do postaci wymaganej przez algorytm przekształcania do postaci normalnej Chomsky'ego, wystarczy usunąć ε-produkcje, produkcje łańcuchowe i S -> ε (zmieniając język, ale zgodnie z odpowiedzią)

c., d., e. - nie możemy zmienić języka, a może mieć słowo puste

5. **Odnośnie lematu o pompowaniu dla języków regularnych prawdziwe są następujące stwierdzenia:**
   a. **lemat służy pokazaniu, że określone języki są regularne**
   b. **schemat postępowania jest następujący: skoro język posiada pewne własności regularności, to jest regularny**
   c. **lemat służy do dowodzenia, że dany język nie jest językiem regularnym (X)**
   d. **lemat służy do dowodzenia, że dany język jest językiem regularnym**

Lemat o pompowaniu to warunek konieczny regularności języka. Służy do dowodzenia, że określone języki nie są regularne.

6. **Jeżeli *r* oraz *s* są wyrażeniami regularnymi dla języków odpowiednio *R* oraz *S*, to *(r + s)*, *rs* oraz *r** są wyrażeniami regularnymi reprezentującymi odpowiednio zbiory:**
   a. **R ∪S, R × S, R+**
   b. **R ∪S, R × S, R***
   c. **R ∪S, RS i R* (X)**
   d. **R ∪S, RS i R+**

Konkatenacja (RS) to nie to samo, co iloczyn kartezjański (R × S).

7. **Mamy języki *L1 = {a^(2^n), n > 0}* oraz *L2 = {a^{2n}, n > 0}*. Które z tych języków są regularne?**
   a. **L1 - nie, L2 - nie**
   b. **L1 - tak, L2 - nie**
   c. **L1 - nie, L2 - tak (X)**
   d. **L1 - tak, L2 - tak**

Z lematu o pompowaniu L1 nie jest regularny. Odpowiedź formalna: zadanie 11.4. Odpowiedź nieformalna: języki, których liczba liter rośnie eksponencjalnie, nie są regularne, bo liniowy wzrost n (np. z 2 do 3) powoduje eksponencjalny wzrost liczby liter, które trzeba by dołożyć do słowa, a to nie jest możliwe dla języków regularnych.
Język L2 jest regularny, bo opisuje go wyrażenie regularne aa(aa)*.

Teoria kompilacji

1. **Eliminując niejednoznaczność gramatyki poprzez konwersję do innej gramatyki musimy zachować bez zmian:**
   a. **drzewo wyprowadzenia oryginalnej gramatyki dla każdego jednoznacznego słowa wejściowego**
   b. **wyprowadzenia oryginalnej gramatyki dla każdego słowa wejściowego**
   c. **zbiór słów generowanych przez oryginalną gramatykę (X)**
   d. **drzewo wyprowadzenia oryginalnej gramatyki dla każdego słowa wejściowego**
   e. **wyprowadzenia oryginalnej gramatyki dla każdego jednoznacznego słowa wejściowego**

2. **Która z następujących metod parsingu może przetworzyć najszerszą klasę gramatyk:**
   a. **parser SLR**
   b. **zstępujący parser rekurencyjny bez nawracania**
   c. **kanoniczny parser LR**
   d. **parser Earleya (X)**
   e. **parser LL(1)**
   f. **parser CYK (X)**
   g. **parser LALR(1)**
   h. **parser oparty na priorytecie operatorów**

3. **Wybierz stwierdzenia, które są prawdziwe:**
   a. **LR wymaga usunięcia lewostronnej rekursji**
   b. **LL wymaga na początku faktoryzacji lewostronnej**
   c. **gramatyki parsowane przez LR są nadzbiorem właściwym gramatyk parsowanych przez LL (X)**
   d. **LR parsuje od prawej do lewej, LL od lewej do prawej**
   e. **dla każdej gramatyki jednoznacznej można skonstruować parser LR(1)**
   f. **żadna gramatyka niejednoznaczna nie może być gramatyką LR(1) (X)**
   g. **tworzenie parsera jest sensowną procedurą tylko dla jednoznacznej gramatyki bezkontekstowej**

4. **Wybierz stwierdzenia dotyczące gramatyk, które są prawdziwe:**
   a. **dla każdej gramatyki jednoznacznej można skonstruować parser LR(1)**
   b. **żadna gramatyka niejednoznaczna nie może być gramatyką LR(1) (X)**
   c. **tworzenie parsera jest sensowną procedurą tylko dla jednoznacznej gramatyki bezkontekstowej (nie ma sensu konstruowanie parsera dla gramatyki niejednoznacznej)**
   d. **konstruując parser dla gramatyki niejednoznacznej i usuwając konflikty uzyskujemy bardziej efektywne rozwiązania, niż opierając parser na równoważnej gramatyce jednoznacznej (X)**

a., b. - tylko dla gramatyk jednoznacznych klasy LR(1) można skonstruować taki parser
d. - bo zmiana gramatyki na jednoznaczną przed konstrukcją parsera może eksponencjalnie zwiększyć złożoność

5. **Typowy skaner języka formalnego ma za zadanie:**
   a. **wyodrębnić symbole leksykalne (X)**
   b. **zliczyć słowa kluczowe i sprawdzić ich kolokacje**
   c. **zliczyć lewe i prawe nawiasy sprawdzając wstępnie ich sparowanie**
   d. **wczytać kod źródłowy programu do postaci tokenów (X)**

Zadania skanera (leksera):
- wyodrębnianie symboli leksykalnych (tokenów)
- ignorowanie komentarzy i białych znaków
- korelowanie błędów zgłaszanych przez kompilator z numerami linii
- tworzenie kopii zbioru wejściowego wraz z komunikatami o błędach

6. **Typowy parser języka formalnego ma za zadanie:**
   a. **usunąć komentarze zagnieżdżone w innych komentarzach**

Zadania parsera:
- rozbiór gramatyczny ciągu tokenów
- zbudowanie drzewa rozbioru - AST
- opcjonalnie: kontrola typów

7. **W odniesieniu do pracy parserów klasy *LR(k)* i funkcji *action* prawdziwe są stwierdzenia:**
   a. **funkcja *action* przyjmuje wartości ze zbioru {*shift*, *reduce*, *goto*, *accept*, *error*}**
   b. **na stosie trzymane są prefiksy i sufiksy form zdaniowych, co do których jest nadzieja na ich wykorzystanie**
   c. **funkcja *action* przyjmuje wartości ze zbioru {*shift*, *reduce*, *accept*, *error*} (X)**
   d. **na stosie trzymane są prefiksy wszystkich form zdaniowych**

Funkcja *action* przyjmuje shift z numerem stanu, reduce z numerem produkcji i jest związana z napotykanymi terminalami (nieterminale to funkcja GOTO).

8. **Budowa tablic sterujących dla analizatorów klasy *LR* może stwarzać pewne trudności, szczególnie w zakresie automatyzacji, co ma pośredni wpływ na istnienie wielu odmian tych parserów. Które z poniższych stwierdzeń są prawdziwe:**
   a. **pierwsza litera w nazwie *SLR* oznacza *Shift***
   b. **pierwsza litera w nazwie *SLR* oznacza *Simple* (X)**
   c. **pierwsze litery w nazwie LALR oznaczają LookAhead (X)**
   d. **pierwsza litera w nazwie GLR oznacza Grammar**

Wyróżniamy 2 typy parserów wywodzące się od LR, akceptujące podzbiór właściwy języków klasy LR(1): SLR(1) i LALR(1).

SLR(1):
- Simple LR(1)
- właściwy podzbiór LALR(1)

- generuje znacznie mniejsze tablice niż LR(1)
- rozszerza nieco LR(0)
- wymaga albo gramatyki jednoznacznej, albo dodatkowych zasad rozwiązywania konfliktów (pojawiają się przy gramatykach niejednoznacznych), np. pierwszeństw operatorów

LALR(1):
- Look Ahead LR(1)
- generuje parser o rozmiarze tablicy sterującej taki sam jak SLR(1), a zatem też znacznie mniejszy niż dla LR(1)
- właściwy nadzbiór SLR(1)
- najważniejsza podklasa LR(1), bo obejmuje znaczną większość języków programowania

9. **Porównując gramatyki LL i LR można powiedzieć, że:**
   a. **gramatyki LL opisują szerszą klasę niż LR**
   b. **gramatyki LR opisują szerszą klasę niż LL (X)**
   c. **każda gramatyka LL jest również gramatyką LR (X)**
   d. **każda gramatyka LR jest również gramatyką LL**

Gramatyki LR(k) to ścisły nadzbiór gramatyk LL(k)

Architektury komputerów

1. **Korzystając z układu FPGA można wykonać:**
   a. **na przykład dowolny układ kombinacyjny, ograniczony jedynie wielkością struktury FPGA (X)**
   b. **na przykład dowolny układ sekwencyjny, ograniczony jedynie wielkością struktury FPGA (X)**

Układy kombinacyjne i sekwencyjne

Układ kombinacyjny - taki, w którym każda kombinacja sygnałów wejściowych jednoznacznie określa kombinację sygnałów wyjściowych. Innymi słowy działa jak funkcja - dostaje wejście i deterministycznie zwraca dla niego wyjście. Przykłady: multiplekser / demultiplekser, koder / dekoder / transkoder, sumator, komparator.

Układ sekwencyjny - taki, w którym wyjście zależy nie tylko od wejścia, ale także od poprzednich stanów wejściowych (stanu pamięci) i kolejności występowania słów wejściowych. Wymaga wyposażenia w pamięć. Przykłady: procesory, przerzutniki, cokolwiek wyposażonego w liczniki / rejestry.

2. **Układ kombinacyjny to:**
   a. **układ logiczny nie pamiętający stanów poprzednich (X)**
   b. **w jego skład mogą wchodzić bramki logiczne w połączeniu z przerzutnikami JK**
   c. **układ cyfrowy, w którym stan wyjść zależy wyłącznie od stanu wejść (X)**

a., c. - może mieć pamięć, ważne jest tylko, żeby wyjście zależało tylko od wejścia

3. **Układ sekwencyjny to:**
   a. **układ logiczny nie pamiętający stanów poprzednich**
   b. **może składać się z samych bramek logicznych (X)**
   c. **może składać się z samych bramek logicznych bez sprzężeń zwrotnych**
   d. **w skład jego mogą wchodzić bramki logiczne w połączeniu z przerzutnikami JK (X)**

a. - no właśnie pamięta, bo posiada własny wewnętrzny stan nazywany stanem poprzednim
b. - czemu nie, przerzutniki to w końcu same bramki
c. - bez feedbacku nie da się zrobić pamięci

Układ sekwencyjny wykorzystuje wejście oraz stan (pamięć) do obliczenia wyjścia.

4. **Przykłady układów sekwencyjnych to:**
   a. **multiplekser oraz transkoder**
   b. **rejestr przesuwający szeregowy oraz dekoder**
   c. **licznik dwukierunkowy oraz rejestr przesuwający (X)**
   d. **żadne z pozostałych**

5. **Pamięć RAM:**
   a. **posiada wejścia adresowe, wejścia sterujące oraz wejście/wyjście danych (X)**
   b. **można wykonać z bramek NAND (X)**
   c. **można wykonać z bramek NAND bez sprzężeń zwrotnych**

6. **Pamięć RAM dwuportowa:**
   a. **możemy wykonać z bramek NAND bez sprzężeń zwrotnych**
   b. **to pamięć RAM z dwoma interfejsami, pozwalającymi niezależnie uzyskać dostęp do tych samych komórek pamięci (X)**
   c. **w układach FPGA taki rodzaj pamięci nie występuje**
   d. **to rodzaj pamięci RAM umożliwiający dwóm niezależnym procesom dostęp do wspólnych danych (X)**
   e. **można ją wykorzystać wyłącznie w procesorach wielordzeniowych**

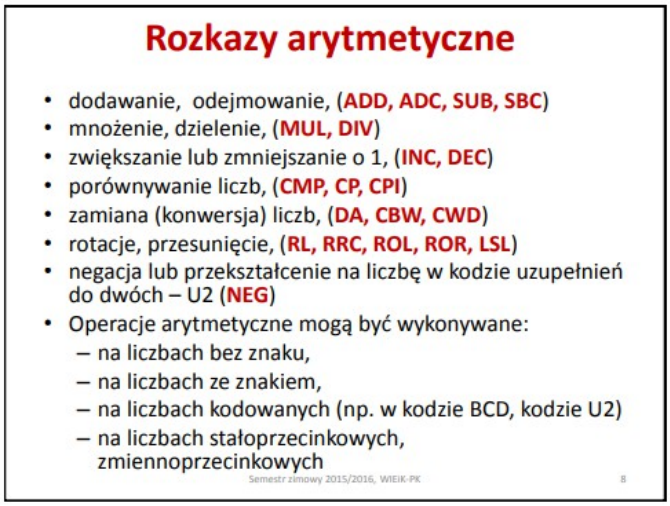
7. **Procesor:**
   a. **możemy wykonać przy użyciu FPGA, ale tylko jednordzeniowy**
   b. **żadne z pozostałych**
   c. **to sekwencyjne urządzenie cyfrowe, które pobiera dane z pamięci operacyjnej, interpretuje je i wykonuje jako rozkazy (X)**
   d. **tryby adresowania wykonywanego przy użyciu FPGA muszą być zgodne z trybami przewidzianymi przez producenta układu**
   e. **możemy wykonać przy użyciu FPGA (X)**

a. - po wykonaniu jednordzeniowego procesora wystarczy taki blok skopiować N razy i mamy N-rdzeniowy

Rozkazy procesora

Rozkazy tworzące listę rozkazów można podzielić na kilka podstawowych grup, w zależności od ich przeznaczenia (prezentacja):

- rozkazy przesłań, kopiowania
- rozkazy arytmetyczne i logiczne
- rozkazy sterujące wykonywaniem programu
- skoków, obsługi pętli, wywołań i powrotów z podprogramu
- wykonujące operacje na ciągach słów
- rozkazy wejścia/wyjścia
- rozkazy sterujące pracą procesora
- inne



Rozkazy mikroprocesorów mogą być:

- jednobajtowe 8 bitów,
- jednosłowowe, np. 12-bitowe, 14-bitowe, 24-bitowe
- wielobajtowe, np. 16-bitowe, 32-bitowe lub więcej

8. **Lista rozkazów procesora:**
   a. **w skład listy rozkazów zawsze wchodzi mnożenie**
   b. **musi zawierać rozkazy z różnymi trybami adresowania**
   c. **w procesorze wykonywanym przy użyciu FPGA musi być zgodna z listą rozkazów przewidzianą przez producenta układu FPGA**
   d. **projektowana jest w zależności od potrzeb związanych z zastosowaniem procesora (X)**

a. - nie musi, patrz np. procesor Z80
b. - mogą, ale nie muszą
c. - to my projektujemy tak, jak chcemy

9. **Karta graficzna:**
   a. **może być układem kombinacyjnym**
   b. **przy użyciu FPGA nie można zbudować karty graficznej ze sprzętowym wspomaganiem OpenGL**
   c. **prostą wersję można zapisać w dwudziestu kilku liniach VHDL (X)**
   d. **OpenGL to specyfikacja otwartego i uniwersalnego API do tworzenia grafiki; jest to zestaw podstawowych funkcji umożliwiających tworzenie grafiki (X)**

a. - nie, bo nie mogłaby używać pamięci
b. - można, za pomocą FPGA można zbudować dowolnie skomplikowany układ

10. **Licznik rozkazów:**
    a. **jest to licznik z wejściem równoległym wykorzystywanym wyłącznie przy skokach bezwarunkowych**
    b. **służy do pamiętania adresu mającego się wykonać rozkazu lub adresu aktualnie pobieranego argumentu z pamięci programu (X)**

11. **Rozkaz skoku bezwarunkowego procesora:**
    a. **powoduje wpisanie do licznika rozkazów adresu rozkazu mającego się wykonać po skoku niezależnie od warunku (X)**
    b. **powoduje wpisanie do licznika rozkazów adresu rozkazu mającego się wykonać po skoku, ale tylko w przypadku spełnienia warunku skoku**
    c. **nie wpływa na stan licznika**

a. - dokładnie na tym polega bezwarunkowość
b. - to jest skok warunkowy

12. **Rozkaz skoku warunkowego procesora:**
    a. **nie wpływa na stan licznika rozkazów procesora**
    b. **powoduje wpisanie do licznika rozkazów adresu rozkazu mającego się wykonać po skoku niezależnie od warunku**
    c. **powoduje wpisanie do licznika rozkazów adresu rozkazu mającego się wykonać po skoku, ale tylko w przypadku spełnienia warunku skoku (X)**
    d. **nie wpływa na stan licznika**

a. - wypływa, bo adres instrukcji do wykonania po skoku musi się znaleźć w liczniku rozkazów
b. - to jest skok bezwarunkowy

13. **Rozkaz procesora wykonujący dodawanie dwóch liczb:**
    a. **powoduje dodanie dwóch liczb, wynik zapisuje do licznika rozkazów**
    b. **wykorzystuje ALU (X)**
    c. **żadne z pozostałych**

ALU - Arithmetic and Logical Unit, układ liczący w procesorze

14. **W procesorze wykorzystującym przetwarzanie potokowe:**
    a. **wykonanie pojedynczej instrukcji rozkłada się na ciąg prostszych etapów (X)**
    b. **rozpoczęcie wykonania pierwszego etapu rozkazu może nastąpić dopiero po zakończeniu wykonania pierwszego etapu poprzedniego rozkazu (X)**
    c. **rozpoczęcie wykonania rozkazu może nastąpić dopiero po zakończeniu wykonania poprzedniego rozkazu**

b. - etapy muszą być wykonywane 1 naraz, przetwarzanie potokowe umożliwia wykonywanie różnych etapów dla wielu zadań równolegle

15. **W procesorze wykorzystującym superskalarność:**
    a. **rozpoczęcie wykonania pierwszego etapu rozkazu może nastąpić dopiero po zakończeniu wykonania pierwszego etapu poprzedniego rozkazu**
    b. **ten sam etap dwóch kolejnych rozkazów może być wykonywany w tej samej chwili (X)**
    c. **możliwe jest jednoczesne wykonanie więcej niż jednej instrukcji (X)**

b., c. - procesor superskalarny ma wiele równoległych jednostek (np. kilka ALU obok siebie), dzięki czemu może faktycznie robić jednocześnie wiele zadań

16. **Rejestr rozkazów:**
    a. **przechowuje adres rozkazu wczytany z pamięci programu**
    b. **jego zawartość wykorzystywana jest przez jednostkę sterującą (X)**
    c. **w trakcie wykonywania rozkazu zawartość rejestru rozkazów musi zmienić się bezpośrednio przed pobraniem argumentu rozkazu z pamięci programu (X)**
    d. **przechowuje kod rozkazu wczytany z pamięci programu (X)**

17. **Transmisja asynchroniczna:**
    a. **układy nadawczy i odbiorczy synchroniowane są wspólną dodatkową linią z sygnałem synchronicznym**
    b. **podczas transmisji asynchronicznej występują kolejno po sobie wycinki czasu zawierające i nie zawierające informacji (X)**
    c. **aby przygotować odbiorcę na przyjęcie sygnału zawierającego informacje, są wysyłane sygnały startu oraz sygnały stopu, rozpoczynające i kończące przesyłanie porcji informacji (X)**
    d. **żadne z pozostałych**

a. - to jest transmisja synchroniczna

18. **Jeżeli moc pobierana przez układ jest najważniejszym kryterium branym pod uwagę przy projektowaniu to którą rodzinę układów wybierzesz:**
    a. **CMOS (X)**
    b. **ECL**
    c. **standard TTL**
    d. **ALS TTL**

19. **Wyrażenie Boolowskie może być zrealizowane z:**
    a. **kombinacji NAND i NOR (X)**
    b. **samych NANDów (X)**
    c. **samych NORów (X)**
    d. **samych XORów**
    e. **kombinacji XOR i NOT**

20. **Dowolną funkcję logiczną można zrealizować na bramkach:**
    a. **AND i NOR (X)**
    b. **XOR**
    c. **OR**
    d. **AND**

a. - bo można ułożyć z nich NAND

21. **Rolą zegara doprowadzonego do przerzutnika bistabilnego jest:**
    a. **dążenie do zmiany na wyjściu w zależności od stanu wejść przygotowujących (S-R, J-K, D) (X)**
    b. **wyzerowanie układu**
    c. **ustawienie układu**
    d. **zawsze wywołanie zmiany na wyjściu**

Przerzutniki bistabilne to takie, które kasują zapamiętany stan przez podanie na wejście kasujące odpowiedniego sygnału. Nazwa stąd, że mają 2 stabilne stany: zapamiętany albo skasowany.
W wersji synchronicznej mają wejście zegarowe. Dopiero po podaniu na wejście zegarowe sygnału pojawia się stan wyjściowy.

a. - zmiana następuje w takt zegara
b. - to jest rola wejścia resetującego (np. R)
c. - to jest rola wejścia ustawiającego (np. S)
d. - tylko, gdy stan innych wejść jest odpowiedni

22. **Przerzutnik JK jest w stanie stałych zmian na wyjściu gdy:**
    a. **J = 1, K = 1 (X)**
    b. **J = 1, K = 0**
    c. **J = 0, K = 1**
    d. **J = 0, K = 0**

J	K	Opis
0	0	podtrzymanie aktualnego wyjścia
0	1	przejście na 0
1	0	przejście na 1
1	1	zmiana na przeciwne wyjście

23. **Multiwibrator astabilny:**
    a. **jest oscylatorem (X)**
    b. **wymaga okresowego impulsu wymuszającego**
    c. **produkuje nieperiodyczne impulsy na wyjściu**
    d. **nie ma stanu stabilnego (X)**
    e. **generuje sygnał sinusoidalny**
    f. **może generować sygnał o różnym współczynniku wypełnienia (X)**

a. - taka jego rola
b. - używa feedbacku
c. - periodyczne
d. - bo jest astabilny
e. - może też inne, np. kwadratowy
f. - np. używając dzielnika napięcia

24. **Który stan jest nieważny w liczniku BCD:**
    a. **1100 (X)**
    b. **0010**
    c. **0101**
    d. **1000**

Kodowanie BCD koduje cyfry binarne 0-9 na 4 bitach. Używa zatem tylko pierwszych 10 wartości, od 0000 (cyfra 0) do 1001 (cyfra 9). Wartości wyższe są nieważne.

25. **10 MHz zegar jest podłączony do kaskady liczników zawierającej - licznik modulo-5, modulo-8 i dwa liczniki modulo 10. Jaka jest częstotliwość wyjścia:**
    a. **2,5 kHz (X)**
    b. **25 kHz**
    c. **5 kHz**
    d. **10 kHz**

26. **Licznik asynchroniczny różni się od synchronicznego:**
    a. **liczbą stanów w jego sekwencji**
    b. **typem użytych przerzutników**
    c. **metodą zegarowania (X)**
    d. **liczbą podziału**

27. **Stopnie w rejestrze przesuwnym zawierają:**
    a. **bajt pamięci**
    b. **4 bity pamięci**
    c. **przerzutniki (X)**
    d. **zatrzaski**

28. **Przy użyciu 1 MHz zegara, osiem bitów równolegle może być wprowadzonych do rejestru przesuwnego w czasie:**
    a. **1 µs (X)**
    b. **w czasie propagacji 8 przerzutników**
    c. **8 µs**
    d. **w czasie propagacji 1 przerzutnika (X)**

W rejestrach równoległych w jednym cyklu zegara zapisujemy i odczytujemy całe słowo, np. 8-bitowe jak w tym przykładzie. Zajmuje to tyle, ile 1 przerzutnik potrzebuje czasu. Jako że mamy zegar $1\ MHz = 10^6$, to zajmuje to $10^{-6}\ s = 1\ \mu s$.

29. **Grupa bitów 10110101 jest szeregowo przesuwana w prawo w ośmiobitowym rejestrze ze stanem początkowym 11100100. Po dwóch impulsach zegara stan wyjścia jest:**
    a. **01011110**
    b. **01111001 (X)**
    c. **10110101**
    d. **00101101**

Po prostu przesuwamy stan obecnego rejestru o 2 pozycje w prawo, dodając po lewej 2 bity z prawej strony wejścia:
10110101
11100100 -> 111001 -> 01111001

30. **8-bitowy przetwornik DAC ma rozdzielczość:**
    a. **1%**
    b. **0,392% (X)**
    c. **3,92%**
    d. **0,1%**

Rozdzielczość przetworników DAC to $100\% / 2^N$, gdzie N - liczba bitów.
$100\% / 2^8 = 100\% / 256 \approx 0,39$

31. **W przetworniku DAC typu R-2R są:**
    a. **rezystory o jednej wartości**
    b. **2 wartości rezystorów (X)**
    c. **4 wartości rezystorów**
    d. **liczba wartości rezystorów jest równa liczbie wejść**
    e. **liczba rezystorów jest równa liczbie wejść**
    f. **liczba rezystorów jest równa dwukrotności liczby wejść (X)**

Właściwie wszystkie odpowiedzi można wywnioskować z samej nazwy R-2R.

32. **Która zależność jest poprawna:**
    a. **na jeden cykl maszynowy składa się jeden lub kilka cykli rozkazowych a na jeden cykl rozkazowy składa się jeden lub kilka cykli zegarowych**
    b. **na jeden cykl zegarowy składa się jeden lub kilka cykli maszynowych a na jeden cykl maszynowy składa się jeden lub kilka cykli rozkazowych**
    c. **na jeden cykl rozkazowy składa się jeden lub kilka cykli maszynowych a na jeden cykl maszynowy składa się jeden lub kilka cykli zegarowych (X)**

33. **Co wpływa na fakt szybszego dostępu do danych przy użyciu DMA zamiast trybu programowalnego?**
    a. **szybszy zegar**
    b. **czysto sprzętowa realizacja dostępu (X)**
    c. **krótsze ścieżki na płycie głównej**
    d. **większy bufor**

34. **Który rodzaj pamięci wymaga odświeżania?**
    a. **flash**
    b. **DRAM (X)**
    c. **ROM**
    d. **EAPROM**
    e. **SDRAM (X)**
    f. **EEPROM**
    g. **SRAM**

35. **Połączenie procesora z magistralą odbywa się poprzez bufor 3 stanowy aby:**
    a. **uzyskać wysoką impedancję bufora w razie potrzeby (X)**
    b. **uzyskać stan rozwarcia aby odłączyć procesor (X)**
    c. **wykorzystać stałą wysoką impedancję bufora**

36. **Dla utrzymania logicznego stanu komórki pamięci dynamicznej niezbędne okazuje się**
    a. **odświeżanie jej zawartości (X)**
    b. **dostarczanie stałego napięcia do komórki**
    c. **nic nie trzeba robić**

37. **Układ sterujący procesora jest to:**
    a. **układ generujący funkcje sterujące wykonywaniem programu (X)**
    b. **rejestry konfiguracyjne, których stan określa aktualny tryb pracy procesora**
    c. **zestaw funkcji ułatwiających śledzenie przebiegu wykonywania programu**

38. **Procesor otrzymuje sygnał HOLD, oznacza to:**
    a. **wymuszenie wykonania specjalnego programu**
    b. **zerowanie licznika programu**
    c. **chwilowe zawieszenie pracy w celu zwolnienia magistrali (X)**
    d. **przejście do podprogramu**

39. Co to jest mikrooperacja?
- **a. elementarna operacja do wykonania w potoku**
- **b. najprostsza operacja do wykonania w jednym takcie zegara (X)**
- **c. element składowy rozkazu (X)**

40. Po co w procesorach zastosowano pamięć typu ROM w jednostce sterującej?
- **a. dla zorganizowania mikroprogramowanego dekodera (X)**
- **b. dla zachowania programu boot'ującego**
- **c. dla zachowania ustawień procesora**

41. Model pamięci współdzielonej:
- **a. jest typowy dla maszyn SIMD (X)**
- **b. występuje w odniesieniu do klastrów komputerów**
- **c. jest typowy dla maszyn SMP (X)**
- **d. może być wykorzystany w maszynach typu MIMD (X)**

42. Do maszyn typu MIMD należą:
- **a. ccNUMA (X)**
- **b. maszyny wykorzystujące model obliczeniowy z wymianą komunikatów (X)**
- **c. maszyny SMP (X)**
- **d. najczęściej rozwiązania wykorzystujące specjalizowane procesory**

43. Magistrala systemowa:
- **a. jest rozwiązaniem najbardziej zaawansowanym technologicznie**
- **b. charakteryzuje się wysoką przepustowością w porównaniu z innymi rozwiązaniami warstwy komunikacyjnej**
- **c. umożliwia łatwą rozbudowę systemu (X)**
- **d. jest typowa dla maszyn SMP (X)**

44. Najwydajniejszym rozwiązaniem warstwy komunikacyjnej w maszynach o wielu procesorach jest
- **a. Myrinet**
- **b. przełącznica krzyżowa (X)**
- **c. magistrala systemowa**
- **d. sieć przełączająca (switching network)**

45. Model obliczeń równoległych z wymianą komunikatów:
- **a. stosowany jest najczęściej w maszynach o pamięci współdzielonej**
- **b. w obliczeniach jest zazwyczaj bardziej efektywny od modelu pamięci współdzielonej (X)**
- **c. charakterystyczny dla architektur klastrowych (X)**
- **d. charakteryzuje się łatwością programowania**

46. Architektury klastrowe:

a. charakteryzują się wysoką dostępnością i istnieniem zasobów zastępczych (X)
b. współdzielą wyłącznie zasoby dyskowe
c. występuje w nich współdzielenie zasobów (X)
d. są przeznaczone wyłącznie do obliczeń naukowo-technicznych

b., c. - współdzielą zasoby obliczeniowe
d. - też do trzymania danych

46. Wielowątkowość i wielordzeniowość procesora

a. są charakterystyczne tylko dla procesorów AMD
b. zostały rozwinięte niezależnie przez producentów (X)
c. są synonimami
d. zostały rozwinięte specjalnie dla technologii Centrino4

Teoria współbieżności

1. **Sekcja krytyczna procesu to:**
   a. **część procesu, w której wykorzystywany jest zasób dzielony (X)**
   b. **część procesu, w której wykonywana jest komunikacja z użytkownikiem**
   c. **część procesu, w której wykonywana jest nieskończona pętla**
   d. **część procesu, w której deklarowane są lokalne zmienne**

2. **Wzajemnym wykluczaniem nazywamy problem współzawodnictwa procesów o zasób, który:**
   a. **musi zostać zainicjalizowany przez jeden z procesów**
   b. **jest zmienną przechowywaną w rejestrach procesora**
   c. **jednocześnie może być wykorzystywany tylko przez jeden z nich (X)**
   d. **może być wykorzystywany tylko przez z góry określony czas**

3. **Komunikację nazywamy asynchroniczną jeżeli:**
   a. **nadawca żąda, by odbiorca był gotów do odebrania komunikatu**
   b. **nadawca może wysłać wiadomość i kontynuować pracę bez wstrzymania (X)**
   c. **przy komunikacji mamy do czynienia z synchronizacją procesów**
   d. **nadawca nie żąda, by odbiorca był gotów do odebrania komunikatu (X)**

4. **Komunikację nazywamy synchroniczną jeżeli:**
   a. **nadawca żąda, by odbiorca był gotów do odebrania komunikatu (X)**
   b. **nadawca nie żąda, by odbiorca był gotów do odebrania komunikatu**
   c. **przy komunikacji mamy do czynienia z synchronizacją procesów (X)**
   d. **nadawca może wysłać wiadomość i kontynuować pracę bez wstrzymania**

5. **Zjawisko blokady w programie współbieżnym jest:**
   a. **zatrzymaniem pracy jednego ze współbieżnych procesów**
   b. **stanem, w którym każdy proces oczekuje na działanie innego procesu (X)**
   c. **przejawem braku bezpieczeństwa programu współbieżnego (X)**
   d. **stanem, w którym proces czekający na dane zdarzenie nie zostaje wznowiony mimo, że występuje ono dowolną liczbę razy**

6. **Zjawisko zagłodzenia w programie współbieżnym jest:**
   a. **stanem, w którym każdy proces oczekuje na działanie innego procesu**
   b. **przejawem braku bezpieczeństwa programu współbieżnego**
   c. **stanem, w którym proces czekający na dane zdarzenie nie zostaje wznowiony mimo, że występuje ono dowolną liczbę razy (X)**
   d. **nienaturalnym przerwaniem pracy jednego ze współbieżnych procesów**
   e. **sytuacją kiedy nie ma gwarancji że określone zdarzenie wykona się w skończonym czasie (X)**
   f. **pojawieniem się cyklu w relacji Lamporta "happen-before"**

f. - nie może być cyklu, bo to mocny porządek, relacja jest przeciwzwrotna

TOiZO

1. **Jakie byłyby konsekwencje znalezienia wielomianowego deterministycznego algorytmu dla problemu NP-zupełnego?**
   a. **żadna z pozostałych odpowiedzi**
   b. **oznaczałoby to, że P =/= NP**
   c. **nie miałoby to żadnych konsekwencji dla teorii złożoności obliczeniowej**
   d. **stanowiłoby to dowód, że P = NP (X)**
   e. **każdy problem algorytmiczny można byłoby rozwiązać w czasie wielomianowym**

e. - fałsz, bo zgodnie z odpowiedzią np. problem stopu też dałoby się wtedy rozwiązać, bo jest problemem algorytmicznym, ale jest on nierozstrzygalny

2. **Jakie problemy zaliczamy do klasy problemów NP-zupełnych?**
   a. **z definicji należą jednocześnie do NP i co-NP**
   b. **należą do klasy NP, ale nie należą do P**
   c. **żadna z pozostałych odpowiedzi**
   d. **wszystkie problemy z klasy NP redukują się do nich i same należą do klasy NP (X)**
   e. **takie, które są w NP i redukuje się do nich problem SAT (X)**
   f. **takie, które są w NP i redukują się do problemu SAT**

a. - problem NP = coNP jest otwarty
b. - problem NP = P jest otwarty
d. - z definicji
e. - w tę stronę, bo to udowadnia, że nasz problem jest co najmniej tak samo trudny jak SAT

3. **Co można powiedzieć o podproblemach problemu π, wiedząc, że π nie należy do P?**
   a. **żaden jego podproblem nie należy do P**
   b. **każdy jego podproblem należy do klasy P**
   c. **pojęcie podproblemu nie jest zdefiniowane**
   d. **istnieje taki jego podproblem, który należy do P (X)**

Odpowiedź wynika z najsłabszego twierdzenia o aproksymacji, które mówi, że każdy język nierozstrzygalny, ale akceptowalny ma nieskończony rozstrzygalny podzbiór.

4. **Która z poniższych złożoności czasowych jest wykładnicza:**
   a. **$O(n^{100})$**
   b. **O(n^(1/n!))**
   c. **$O(\log(10)^n)$ (X)**
   d. **O(n!)**

5. **Co nazywamy mostem grafu?**
   a. **minimalną liczbę krawędzi grafu, których usunięcie zmienia graf w niespójny lub trywialny**
   b. **krawędź grafu spójnego, której usunięcie z grafu rozspójnia go (X)**
   c. **krawędź, której usunięcie zwiększa liczbę spójnych składowych grafu (X)**
   d. **maksymalną liczbę krawędzi grafu, których usunięcie zmienia graf w niespójny**

6. **W teorii złożoności obliczeniowej wszystkie problemy decyzyjne, które w wielomianowym czasie rozwiązuje niedeterministyczna maszyna Turinga, tworzą pewną klasę problemów. Jak brzmi jej nazwa?**
   a. **klasa NP (X)**
   b. **klasa P**
   c. **klasa coNP**
   d. **klasa RE**

Technologie Obiektowe

1. **Generalizacja:**
   a. **organizuje obiekty w hierarchię uogólniania/uszczegóławiania**
   b. **organizuje klasy w hierarchię całości/części**
   c. **znajduje swoje odzwierciedlenie w mechanizmie dziedziczenia interfejsów (X)**
   d. **znajduje swoje odzwierciedlenie w mechanizmie dziedziczenia implementacji**
   e. **organizuje klasy w hierarchię uogólniania/uszczegóławiania (X)**

a., e. - klasy, nie obiekty
c., d. - bo tak mówi klucz

2. **Kompozycja:**
   a. **jest to struktura agregacji, w której klasy części mogą być powiązane tylko z jedną klasą całości**
   b. **ogranicza powiązanie klasy części do jednej klasy całości**
   c. **ogranicza powiązanie obiektu części do jednego obiektu całości (X)**
   d. **jest silnym powiązaniem z czasem życia części ograniczonym do czasu życia całości (X)**
   e. **całość jest odpowiedzialna za zarządzanie tworzeniem i usuwaniem części (X)**
   f. **ściśle związana z generalizacją**
   g. **ściśle związana z delegacją (X)**

a., b., c. - klasy-części mogą należeć do dowolnej liczby klas-całości, natomiast obiekt-część może należeć do tylko jednego obiektu-całości
d. - dlatego nazywa się też silną agregacją

3. **Relacja zależności wskazuje, że**
   a. **klasa dostarcza implementacji usług interfejsu**
   b. **istnieje szczególne powiązanie między klasami**
   c. **obiekt klasy może być argumentem wywołania usługi innej klasy (X)**
   d. **implementacja klasy może się zmienić, gdy zmieni się implementacja innej klasy (X)**
   e. **przypadek użycia zależy od innego przypadku użycia**
   f. **zmiana specyfikacji jednej usługi może mieć wpływ na poprawność działania innej**
   g. **zmiana stanu obiektu zależy od stanu innego obiektu**

a. - realizacja
b. - asocjacja

4. **Relacja realizacji wskazuje, że:**
   a. **obiekt klasy może być argumentem wywołania usługi innej klasy**
   b. **klasa dostarcza implementacji usług interfejsu (X)**
   c. **klasa realizuje usługi delegowane z innej klasy**
   d. **klasa wykorzystuje usługi interfejsu**

5. **Wielokrotne użycie wspierane jest w modelu obiektowym przez:**
   a. **kwalifikowane powiązania**
   b. **kompozycje z delegacją roli (X)**
   c. **istnienie klas obiektów (X)**
   d. **dziedziczenie implementacji (X)**

Inżynieria Oprogramowania

1. **Celem testowania oprogramowania jest:**
   a. **zbadanie zgodności z wymaganiami (X)**
   b. **wykrycie błędów w oprogramowaniu (X)**
   c. **zbadanie zgodności z oczekiwaniami użytkownika (X)**
   d. **sprawdzenie poprawności komentarzy w kodzie**
   e. **ocena jakości oprogramowania (X)**
   f. **zdefiniowanie jakości kodu**

a. - testy akceptacyjne
e., f. - jakość kodu narzucamy sobie przed całym procesem (może być to np. wymaganie niefunkcjonalne zewnętrzne), natomiast po stworzeniu (przy testach) oceniamy jak nam wyszło

2. **Jakie są główne aktywności w modelu spiralnym?**
   a. **Planowanie, Analiza Ryzyka, Konstrukcja, Walidacja (X)**
   b. **Szybki Projekt, Budowa Prototypu, Ocena Prototypu, Redefinicja Prototypu**
   c. **Analiza Wymagań, Projektowanie, Kodowanie, Testowanie**
   d. **Definiowanie, Prototypowanie, Testowanie, Dostarczenie produktu**

b. - metoda prototypowa
c. - metoda iteracyjna
d. - uproszczony Waterfall

3. **Jakiego widoku nie znajdziesz w modelu architektonicznym Kruchtena?**
   a. **przypadków użycia**
   b. **konstrukcji**
   c. **logicznego**
   d. **przepływu danych (X)**

Model Kruchtena zawiera widoki:
- przypadków użycia - wszystkie pozostałe 4 na nim polegają, stąd pełna nazwa "model Kruchtena 4 + 1"
- fizyczny
- logiczny
- konstrukcji
- procesu

4. **Jaką rolę na diagramach klas UML pełni kompozycja?**
   a. **jest związkiem typu gen-spec między instancjami klasy**
   b. **jest związkiem typu część-całość między obiektami (X)**
   c. **wspomaga graficzne rozlokowanie symboli klas na diagramie**
   d. **oznacza ukrywanie złożoności obiektów przed użytkownikiem**

a. - dziedziczenie

Kompozycja (agregacja całkowita) to relacja typu część-całość, w której całość jest wyłącznym właścicielem części, tworzy je i zarządza nimi.

5. **Jednym z celów inżynierii oprogramowania jest tworzenie oprogramowania, które jest:**
   a. **wolne od błędów (X)**
   b. **dostarczane zgodnie z harmonogramem (X)**
   c. **niezawodne i efektywne (X)**
   d. **niezgodne z wymaganiami**

6. **Przykładem ryzyka występującego podczas wytwarzania oprogramowania jest:**
   a. **odejście kluczowych osób z zespołu projektowego, zanim produkt zostanie wdrożony (X)**
   b. **konkurenci mogą sprzedawać taniej system o podobnej funkcjonalności**
   c. **problemy ze zdefiniowaniem specyfikacji produktu (X)**
   d. **wykonywanie kolejnych faz zgodnie z ustalonym harmonogramem**

7. **Tworzenie modelu obiektowego z istniejącej relacyjnej bazy danych jest określane jako:**
   a. **forward engineering**
   b. **backward engineering**
   c. **reverse engineering (X)**
   d. **inverse engineering**

8. **Stosowanie techniki prototypowania jest zalecane dla:**
   a. **aplikacji budowanej w oparciu o frameworki**
   b. **gdy liczy się czas dostarczenia**
   c. **zespołów programistów, którym brakuje znajomości dziedziny (X)**
   d. **gdy istnieje trudność uzyskania pełnej informacji o wymaganiach systemu (X)**

9. **Które z poniższych stwierdzeń nie jest celem budowy modeli analitycznych:**
   a. **opracowanie rozwiązań problemów (X)**
   b. **ustalenie wszystkich czynników / warunków w dziedzinie przedmiotowej / w otoczeniu projektu, które mogą wpłynąć na decyzje projektowe**
   c. **stworzenie logicznego modelu systemu, opisującego sposób realizacji wymagań, ale bez szczegółów implementacyjnych**

Modele analityczne mają ustalić wszystkie czynniki/warunki wokół projektu (istniejące systemy, firmy, programiści robiący projekt), które mogą wpłynąć na decyzje projektowe, przebieg procesu projektowego i realizację wymagań.
Wynik analizy to logiczny model systemu, opisujący sposób realizacji postawionych wymagań, ale bez szczegółów implementacyjnych.

10. **Wstępna próba zdefiniowania elementów systemu oraz ich wzajemnych relacji, organizowanie tych elementów w dobrze określone warstwy z wyraźnie nakreślonymi zależnościami nazywa się analizą:**
    a. **przypadków użycia**
    b. **architektoniczną (X)**
    c. **strukturalną**
    d. **systemową**

11. **Którego z poniższych narzędzi nie używa się podczas analizy systemowej?**
    a. **Data Flow Diagram**
    b. **Decision Tree (X)**
    c. **Object Modelling Technique**
    d. **HOOD**

12. **Które z podanych rodzajów wymagań nie są określane podczas przygotowywania definicji wymagań systemowych?**
    a. **abstrakcyjne wymagania funkcjonalne**
    b. **szczegółowe wymagania funkcjonalne (X)**
    c. **cechy lub zachowania, których system nie powinien przejawiać**
    d. **właściwości systemu**

13. **Które z wymienionych czynności występują dla wszystkich procesów wytwarzania oprogramowania?**
    a. **budowa i ulepszanie prototypów**
    b. **retrospektywa celem polepszenia procesu**
    c. **pozyskiwanie wymagań (X)**
    d. **ciągła integracja**
    e. **testowanie wydajności**
    f. **walidacja oprogramowania (X)**
    g. **specyfikowanie oprogramowania (X)**
    h. **zarządzanie konfiguracjami**

a. - prototypowanie
b. - Scrum

d. - Kanban
e. - wszędzie może być (i powinno), nigdzie nie musi
h. - zależy od systemu

14. **Zapis wymagania powinien bezwzględnie zawierać:**
    a. **informację o kolejności wprowadzenia danego wymagania w stosunku do innych wymagań**
    b. **informację o uzależnieniu danego wymagania od innych wymagań lub powiązaniu z nimi (X)**
    c. **informację o pochodzeniu danego wymagania i jego uzasadnienie (X)**
    d. **informację o uzależnieniu projektu od wymagania, tj. powiązaniu danego wymagania ze składnikami systemu, które je implementują (X)**

a., b. - nie jest ważne, kiedy dane wymaganie dokładnie zostanie zrealizowane, ale ważne jest, żeby wiedzieć, co najpierw trzeba zrobić, zanim się w ogóle będziemy mogli się za nie zabrać

15. **Prototypowanie z porzuceniem jest wariantem ewolucyjnego tworzenia oprogramowania, w którym:**
    a. **tworzony jest fragment systemu zwany prototypem, odpowiadający ograniczonej części wymagań, celem eksperymentalnego rozeznania wymagań klienta i ustalenia lub walidacji ich specyfikacji oraz zbadania możliwości ich spełnienia (X)**
    b. **ze względu na dwukrotną realizację systemu - jako prototypu i systemu finalnego - wydłużony jest znacznie czas realizacji i powiększony koszt przedsięwzięcia, przez co ta metoda stosowana jest głównie wtedy, gdy przez oddanie prototypu zamiast finalnej wersji systemu można oddalić groźbę katastrofalnego opóźnienia i całkowitego załamania projektu**
    c. **nie powstaje prawie żadna dokumentacja projektowa i stąd używana niekiedy nazwa "metoda studencka"**
    d. **nie przywiązuje wagi do jakości oprogramowania wytworzonego prototypu ani nawet jego zdolności do funkcjonowania, o ile nie przeszkadza to w wykorzystaniu go do realizacji jego głównego celu (X)**

b. - prototypowania używa się np. wtedy, kiedy wymagania klienta są bardzo niejasne, a prototypy pozwalają je doprecyzować
c. - może powstawać dokumentacja
d. - prototyp jest atrapą dla rozpoznania wymagań, nie ma działać wybitnie

16. **Do modeli iteracyjnych tworzenia oprogramowania zaliczamy:**
    a. **metodę przyrostową (X)**
    b. **model kaskadowy**
    c. **model spiralny tworzenia oprogramowania (X)**
    d. **ewolucyjne tworzenie oprogramowania**
    e. **metodykę Kanban**
    f. **programowanie ekstremalne (X)**

a. - pełna nazwa to iteracyjne przyrastanie
c. - obroty spirali to kolejne iteracje
e. - Kanban to tworzenie ciągłe, bez konkretnych iteracji

17. **Inspekcja (przegląd) wymagań jest:**
    a. **metodą walidacji wymagań (X)**
    b. **nazywana formalną bądź nieformalną, zależnie od tego, czy zespół twórców systemu wyjaśnia znaczenie każdego wymagania, czy tylko prowadzi luźną rozmowę z przedstawicielami klienta (X)**
    c. **procesem "ręcznym" polegającym na grupowym czytaniu dokumentacji wymagań (X)**
    d. **niepodobna do inspekcji programów (kodu)**

18. **Systemy odziedziczone:**
    a. **to systemy starsze niż kilka-kilkanaście lat, których działanie pozostaje krytycznym czynnikiem z punktu widzenia funkcjonowania instytucji, ale technologia nie odpowiada aktualnym potrzebom (X)**
    b. **mają oprogramowanie i sprzęt niezmieniony od czasów ich zainstalowania, ale ich przeróbka i unowocześnienie nie jest możliwe z braku kompatybilnego sprzętu, odpowiednich kompilatorów języków programowania, bibliotek i odpowiednik specjalistów**
    c. **można tylko całkowicie wycofać i/lub zastąpić nowymi systemami**
    d. **obejmują oprogramowanie, sprzęt na którym może ono działać, dane użytkowe przez nie udostępniane i gromadzone, procedury biznesowe, prowadzone z użyciem oprogramowania (X)**

b. - oprogramowanie czy sprzęt mogły się zmieniać
c. - niestety często trzeba to utrzymywać
d. - cały bagaż historyczny, nie tylko kod

19. **Jakie aspekty powinien uwzględniać pełny model systemu w podejściu strukturalnym do analizy i projektowania?**
    a. **aspekt funkcjonalny, aspekt danych i ich powiązań**
    b. **aspekt funkcjonalny i aspekt dynamiki systemu**
    c. **aspekt danych i ich powiązań, aspekt dynamiki systemu**
    d. **aspekt funkcjonalny, aspekt danych i ich powiązań, oraz dynamiki systemu (X)**

Dlatego pełny model systemu składa się minimalnie z 3 diagramów:
- funkcjonalnego - diagram przepływu danych (DFD, Data Flow Diagram)
- danych - diagram encji (ERD, Entity Relationship Diagram)
- dynamiki - diagram przejść stanów (STD, State Transition Diagram, ew. starsze ELH)

20. **Aspekt funkcjonalny systemu informatycznego modelowany jest w podejściu strukturalnym przy pomocy:**
    a. **Data Flow Diagrams (X)**
    b. **Entity Relationship Diagrams**
    c. **State Transition Diagrams**
    d. **Structure Charts**
    e. **Behaviour Diagrams**

21. **Które z wymienionych podziałów wymagań niefunkcjonalnych są wyczerpujące i rozłączne?**
    a. **produktowe, organizacyjne, prawne**
    b. **produktowe, prawne, parametryczne**
    c. **produktowe, zewnętrzne, strategiczne**
    d. **produktowe, organizacyjne, zewnętrzne (X)**

22. **Wymień elementy modelu środowiskowego w strukturalnym podejściu do analizy i projektowania systemów informatycznych:**
    a. **ERD, DFD, STD**
    b. **diagram kontekstowy, słownik danych, specyfikacja funkcji**
    c. **opis celu systemu, wymagania funkcjonalne, wymagania niefunkcjonalne**
    d. **opis celu systemu, lista zdarzeń, diagram kontekstowy (X)**

Model środowiskowy opisuje granicę między systemem a otaczającą go rzeczywistością (środowiskiem). Opisuje po co system istnieje, zdarzenia (wejście/wyjście danych, zdarzenia zależne od czasu) oraz ma wysokopoziomowy diagram DFD, czyli diagram kontekstowy.

23. **Jaki zestaw diagramów jest używany w metodyce strukturalnej analizy i projektowania systemów informatycznych do pokazania wszystkich aspektów modelowanego systemu?**
    a. **DFD, ERD i STC**
    b. **DFD i ERD**
    c. **DFD, ERD i STD (lub ELH) (X)**
    d. **DFD i STD**

24. **Które obiekty graficzne są używane do tworzenia diagramu kontekstowego w metodyce strukturalnej analizy i projektowania systemów informatycznych?**
    a. **proces, przepływ danych (data flow), magazyn danych (data store)**
    b. **proces, obiekt zewnętrzny (external entity), magazyn danych (data store)**
    c. **przepływ danych (data flow), obiekt zewnętrzny (external entity), magazyn danych**
    d. **proces, przepływ danych (data flow), obiekt zewnętrzny (external entity) (X)**

Diagram kontekstowy to wysokopoziomowa, ogólna wersja diagramu DFD. Pokazuje ona granice systemu, wejście/wyjście danych i nadawców/odbiorców danych, czyli obiekty zewnętrzne.

25. **Wybierz nazwy wszystkich obiektów graficznych używanych do konstruowania DFD (Data Flow Diagram) - diagramów przepływu danych w metodyce strukturalnej:**
    a. **proces (process) (X)**
    b. **obiekt zewnętrzny (external entity) (X)**
    c. **magazyn danych (data store) (X)**
    d. **przepływ danych (data flow) (X)**

26. **Wskaż ten element dokumentacji projektowej, który zawiera zapis w postaci pseudokodu (metodyka strukturalna analizy i projektowania systemów informatycznych):**
    a. **opis wymagań niefunkcjonalnych**
    b. **opis wymagań funkcjonalnych**
    c. **specyfikacja funkcji (X)**
    d. **opis celu przedsięwzięcia projektowego**

27. **Który element DFD w metodyce strukturalnej stanowi podstawę konstruowania ERD (Entity Relationship Diagram) - diagramu związków encji?**
    a. **przepływ danych (data flow)**
    b. **diagram kontekstowy (context diagram)**
    c. **obiekt zewnętrzny (external entity)**
    d. **magazyn danych (data store) (X)**

28. **Zaznacz, które stwierdzenia odnoszące się do studium realizowalności (wykonalności) są prawdziwe:**
    a. **studium realizowalności musi udzielać odpowiedzi na pytania "Czy system przyczyni się do realizacji ogólnych celów przedsiębiorstwa?" i "Czy może być zintegrowany z innymi, już będącymi w użyciu?" (X)**
    b. **studium realizowalności musi zawierać wskazanie technologii, jakie będą zastosowane podczas realizacji projektu**
    c. **studium realizowalności ma na celu zaplanowanie końcowego budżetu przedsięwzięcia projektowego**
    d. **po wykonaniu studium realizowalności zawsze następuje faza określania i analizowania wymagań**

c. - może zawierać oszacowanie budżetu, ale początkowe, wraz z odpowiedzią na pytanie czy dla danej organizacji jest to realizowalne pod względem finansowym
d. - nie, jeżeli projekt zostanie odrzucony już przy studium

29. **Zaznacz, które z poniższych stwierdzeń są prawdziwe:**
    a. **w każdym przyroście musi być realizowana funkcjonalność w obrębie jednego podsystemu w taki sposób, aby możliwa była integracja nowego przyrostu z wcześniej wytworzoną częścią systemu**
    b. **tworzenie przyrostowe systemu polega na podzieleniu procesu budowy systemu na pewną liczbę etapów, z których każdy prowadzi do realizacji kolejnej części funkcjonalności wedle tego samego modelu tworzenia oprogramowania**

c. **kolejność realizacji i integracji przyrostów zostaje ustalona na początku projektu, przy określaniu ważności poszczególnych usług, które system ma oferować (X)**
d. **w trakcie realizacji danego przyrostu wymagania w odniesieniu do przypisanej do niego funkcjonalności systemu są zamrożone, ale prowadzi się analizę wymagań dla pozostałych przyrostów (X)**

a. - może być w ramach wielu podsystemów
b., c. - nie dzielimy z góry na konkretną liczbę, mniej-więcej przyrosty i kolejność już tak
d. - patrz np. Scrum

30. **Ewolucyjne tworzenie oprogramowania w wariancie projektowania eksploracyjnego (badawczego):**
    a. **jest realizacją przedsięwzięcia w kolejności od najsłabiej do najlepiej rozpoznanych części systemu**
    b. **polega na systematycznym ulepszaniu utworzonej na początku wersji systemu, przez stopniowe dodawanie cech proponowanych przez klienta, aż do uzyskania zadowalającej go wersji systemu (X)**
    c. **charakteryzuje się opartym o systematyczną współpracę z klientem, przyrostowym wypracowywaniem specyfikacji produktu (X)**
    d. **bywa nazywane metodą przyrostową**

d. - metody iteracyjna i ewolucyjna to 2 różne metodyki

31. **Skrajny wariant ewolucyjnego tworzenia oprogramowania zwany "metodą studencką", charakteryzujący się prawie całkowitym brakiem dokumentacji i selektywnym testowaniem (przy optymistycznych założeniach co do poprawności danych wejściowych):**
    a. **nadaje się do realizacji oprogramowania "jednorazowego użytku" (X)**
    b. **może być wykorzystywany do tworzenia bardzo rzadko używanych komponentów wielkich systemów**
    c. **może być używany do przygotowywania wersji demonstracyjnych i treningowych (X)**
    d. **nie może być stosowany jako metoda tworzenia prototypów przeznaczonych do porzucenia**

a., c., d. - to wszystko jednorazowe prototypy
b. - w dużych systemach każda część ma być utrzymywalna

Sieci komputerowe

1. **Zaletami prowadzenia transmisji w trybie pełnego dupleksu w standardzie Ethernet są:**
   a. **całkowita likwidacja kolizji w segmencie sieci (X)**
   b. **możliwość dwukrotnego zwiększenia sumarycznej przepustowości transmisji (X)**
   c. **brak ograniczenia długości medium transmisyjnego w segmencie warstwy fizycznej wynikającego z wymogów protokołu CSMA/CD (X)**
   d. **możliwości wykorzystania urządzeń hub**

2. **Model odniesienia OSI/ISO:**
   a. **jest modelem czterowarstwowym**
   b. **definiuje standardy w ramach każdej warstwy (np. protokół IP w warstwie sieciowej)**
   c. **określa zadania poszczególnych warstw (X)**
   d. **pozwala na niezależny rozwój sprzętu i oprogramowania w ramach poszczególnych warstw (X)**

a. - model OSI/ISO jest 7-warstwowy, model TCP/IP jest 4-warstwowy
b., c. - definiuje zadania poszczególnych warstw, protokoły same wpisują się w poszczególne warstwy

3. **Osiągnięcie przepustowości 1 Gb/s na kablu UTP kategorii 5e (tym samym, który jest używany w transmisji 100 Mb/s) w standardzie Ethernet jest:**
   a. **możliwe dzięki wykorzystaniu do transmisji wszystkich czterech par (X)**
   b. **możliwe dzięki wykorzystaniu bardziej założonego kodowania (X)**
   c. **nie jest możliwe - wymaga kabla o lepszych parametrach**
   d. **nie jest możliwe - taką przepustowość można osiągnąć tylko używając światłowodu**

4. **Kodowanie sygnałów w transmisji w sieciach komputerowych:**
   a. **wpływa na uzyskiwaną przepustowość (X)**
   b. **jest zależne od wykorzystywanego medium (X)**
   c. **jest przedmiotem zainteresowania warstwy sieciowej modelu OSI/ISO**
   d. **może pozwalać na osiągnięcie samosynchronizacji nadajnika i odbiornika (X)**

c. - warstwy fizycznej

5. **Istnienie ograniczenia maksymalnej odległości między komunikującymi się hostami wynika z:**
   a. **w ramach jednego segmentu sieci: wymogów protokołu warstwy łącza danych (X)**
   b. **w ramach jednego segmentu sieci: parametrów medium fizycznego (X)**
   c. **w ramach jednego segmentu sieci: wymogów protokołu warstwy sieciowej**
   d. **w ramach kilku segmentów sieci rozdzielonych przełącznicami: wymogów protokołu warstwy sieciowej**

a. - warstwa łącza danych może narzucać ograniczenia np. protokołem CSMA/CD
b. - warstwa fizyczna zawsze narzuca ograniczenia w segmentach
c., d. - warstwa sieciowa nie narzuca ograniczeń w segmentach, bo jest bardziej wysokopoziomowa, segmenty mają być już ogarnięte przez warstwy 1-2

6. **Które z wymienionych funkcji realizuje warstwa łącza danych modelu OSI/ISO?**
   a. **dostarczenie adresacji fizycznej (X)**
   b. **umożliwienie definiowania adresacji logicznej**
   c. **detekcja kolizji w medium transmisyjnym (X)**
   d. **definicja strategii dostępu do łącza (X)**

a. - adresacja MAC
c. - np. protokół CSMA/CD

7. **Wirtualne sieci lokalne:**
   a. **można definiować w oparciu o adresy fizyczne (X)**
   b. **mogą być rozpięte na wielu urządzeniach warstwy drugiej modelu OSI/ISO (X)**
   c. **stanowią mechanizm alternatywny dla budowy drzewa rozpinającego**
   d. **stanowią podstawowy mechanizm zapobiegania przeciążeniom mostka**
   e. **wyznaczają domenę rozgłoszeniową (X)**
   f. **poprawiają bezpieczeństwo sieci (X)**
   g. **można definiować w oparciu o adresy logiczne (X)**
   h. **można definiować w oparciu o porty routera**
   i. **służą do dynamicznego uzyskiwania adresu IP**
   j. **mogą być używane zamiast protokołu ARP**

a. - bo VLANy są tworzone na warstwie 2, która używa adresów fizycznych (MAC)
c. - STP działa w ramach VLANu
f. - bo pakiety latają po mniejszych, łatwiejszych do kontroli sieciach
g. - można zdefiniować VLAN dla podsieci IP, czyli używając adresu logicznego

8. **Algorytm budowy drzewa rozpinającego (STA):**
   a. **jest stosowany w przypadku mostków uczących się (X)**
   b. **służy zapobieżeniu występowania zapętleń transmisji ramek (X)**
   c. **jest jednym ze sposobów elekcji głównego routera w sieci**
   d. **służy zrównoważeniu obciążenia pomiędzy łączami o takich samych przepustowościach**

STA = Spanning Tree Algorithm, element STP (Spanning Tree Protocol)

a. - zasadniczo wszystkie współczesne, ale istniały takie czysto statyczne

9. **Które warstwy modelu OSI/ISO są różne dla sieci WAN i LAN?**
   a. **fizyczna i łącza danych (X)**
   b. **fizyczna i sieci**
   c. **łącza danych i sieci**
   d. **fizyczna, łącza danych i sieci**

Od warstwy sieciowej (warstwa 3) w górę nie obchodzi nas to, gdzie dokładnie ktoś jest, mamy adres IP i to wystarcza. Poniżej są różnice, choćby ze względu na skalę długości kabli.

10. **Opisz, do czego jest wykorzystywany protokół ARP:**
    a. **uzyskania adresu IP**
    b. **uzyskania adresu MAC urządzenia spoza naszej sieci lokalnej**
    c. **określenie odwzorowania między adresami warstwy 3 i 2 modelu OSI/ISO (X)**
    d. **wyznaczania ścieżki pakietu**

Protokół ARP pozwala uzyskać odwzorowanie IP -> MAC. W drugą stronę kiedyś używano RARP (Reverse ARP), a obecnie używa się DHCP.

a. - używa się DHCP
d. - do tego służy traceroute używający ICMP

11. **Jeżeli dwie stacje znajdujące się w innych sieciach IP połączone są ze sobą dokładnie jednym routerem i chcą komunikować się ze sobą, to:**
    a. **obydwie stacje muszą posiadać jednakową maskę**
    b. **musi być uruchomiony mechanizm Proxy ARP na routerze**
    c. **na routerze konieczne jest dodanie pozycji routingu statycznego**
    d. **wymagane jest ustawienie na każdej stacji adresu IP (X)**

a. - prawie na pewno mają różne
b. - da się bez, dałoby się też używając tego
c. - da się dynamicznym

12. **Protokół IP w wersji 4 posiada następujące cechy:**
    a. **pozwala na fragmentację pakietów w węzłach pośrednich i u nadawcy (X)**
    b. **zawiera mechanizmy potwierdzania doręczenia pakietów**
    c. **umożliwia kontrolę nagłówka pakietu za pomocą sumy kontrolnej (X)**
    d. **jest protokołem połączeniowym**
    e. **umożliwia komunikację priorytetową (X)**
    f. **jest zawodny, ale każda utrata pakietu jest sygnalizowana**
    g. **posiada płaską adresację**

a. - fragmentacja dokonywana jest wtedy, kiedy MTU (Maximum Transmission Unit) następnego łącza jest zbyt mały dla aktualnego rozmiaru pakietu
b. - trzeba to zrobić na warstwie wyżej (4, transportowej) protokołem TCP, albo jeszcze wyżej zaprogramować
d. - wysyła niezależne pakiety
e. - pole ToS (Type of Service) w nagłówku
f. - nie każda utrata, np. przy przeciążeniu nie
g. - adresacja jest hierarchiczna

13. **W jakich przypadkach jest stosowany protokół ICMP?**
    a. **transmisji danych wysokopiorytetowych**
    b. **uzyskiwania informacji o konfiguracji sieci (X)**
    c. **sygnalizacji sytuacji awaryjnych (X)**
    d. **wyszukiwania odbiorców ruchu multicastowego**

b. - np. Router Advertisement, Router Solicitation
c. - np. Destination Unreachable, Time Exceeded

14. **Jaki mechanizm uniemożliwia nieskończone krążenie ramek IP w sieci w przypadku wystąpienia pętli?**
    a. **mechanizm split-horizon**
    b. **mechanizm trigger-update**
    c. **pole TTL (X)**
    d. **NAT**
    e. **STP**

a. - split horizon to mechanizm eliminujący pętle w protokołach routingu typu distance-vector, na etapie tworzenia tablic routingu
b. - trigger update to mechanizm aktualizacji w protokołach routingu typu link-state
c. - TTL to liczba całkowita oznaczająca liczbę routerów na trasie, która jeszcze została, jest dekrementowane po każdym skoku przez router

15. **Parametr "niezawodność łącza" wchodzi w skład metryki protokołu routingu:**
    a. **EIGRP (X)**
    b. **RIPv2**
    c. **IGRP (X)**
    d. **OSPF**

16. **Protokół TCP:**
    a. **może grupować dane otrzymane do wysłania w celu zoptymalizowania wykorzystania połączeń sieciowych (X)**
    b. **wymaga osobnego potwierdzenia każdego otrzymanego segmentu**
    c. **używa algorytmu Nagle'a w celu zapewnienia kontroli przepływu danych**
    d. **zezwala na wspólne potwierdzenie wielu przesłanych segmentów (X)**

Algorytm Nagle'a służy do zwiększenia efektywności protokołu TCP. Zwalcza on problem małych pakietów, czyli wysyłanie bardzo krótkich wiadomości, które mają często mniej informacji niż rozmiar nagłówka.

a., d. - algorytm Nagle'a służy do optymalizacji protokołu w praktyce
b., d. - można potwierdzać wiele segmentów naraz, bo protokół TCP nadąża za indeksem najdalszego potwierdzonego segmentu

17. **Protokół UDP:**
    a. **wprowadza mechanizmy fragmentacji dla danych użytkownika przekraczających maksymalny rozmiar datagramu UDP**
    b. **pozwala na grupowanie danych otrzymanych do wysłania w celu zoptymalizowania wykorzystania połączeń sieciowych**
    c. **stosuje porty jako adresy warstwy transportowej (X)**
    d. **nie zabezpiecza przed duplikacją i zmianą kolejności datagramów (X)**

a. - dane już mają być podzielone przez wyższe warstwy
b. - zawsze używa się datagramów o określonym maksymalnym rozmiarze

18. **Adres typu broadcast (rozgłoszenia) IP w wersji 4, w której znajduje się host 110.104.1.10 i którą określa maska 255.0.0.0, to:**
    a. **110.104.1.0**
    b. **110.255.255.255 (X)**
    c. **110.104.1.255**

19. **Pole o nazwie Time To Live (TTL) w datagramie IP, które zabezpiecza przed zapętleniem routowania datagramu pomiędzy kolejnymi routerami w sieci, zawiera:**
    a. **liczbę routerów, przez jakie datagram IP może zostać przekazany dalej (X)**
    b. **czas w sekundach, w którym datagram IP można jeszcze przekazywać dalej**

20. **Protokół UDP definiuje identyfikatory przesyłanych do hosta-odbiorcy datagramów zwane numerami portów, o długości w bitach:**
    a. **32**
    b. **16 (X)**
    c. **8**
    d. **4**

Numery portów to 16-bitowe liczby całkowite bez znaku.

21. **Wartości adresu IPv6 oraz maski, określające wszystkie hosty w internecie, to:**
    a. **::/0 (X)**
    b. **::/2000**
    c. **2000::/3**

Patrz ta dyskusja na Stacku.

22. **Istnienie zasady "Longest prefix match" w routingu IP spowoduje, że adres docelowy 200.200.200.1 datagramu IP przy istnieniu w tablicy routingu jednocześnie reguł o wzorcach i maskach (w notacji CIDR): 200.200.200.0/18, 200.200.200.0/20, 200.200.200.0/22, 200.200.200.0/24 zostanie dopasowany do:**
    a. **200.200.200.0/18**
    b. **200.200.200.0/24 (X)**
    c. **200.200.200.0/20**
    d. **200.200.200.0/22**

Pierwsze trzy "200" to 24 bity. Zgadzają się wszystkie, więc bierzemy jak najwięcej, czyli dokładnie /24.

23. **Maksymalna długość pakietu IP w wersji 4, licząc w bajtach, to:**
    a. **576**
    b. **1500**
    c. **1025**
    d. **65535 (X)**

Pole "Total packet length" ma 16 bitów, co daje $2^{16} - 1 = 65535$.

24. **Router iBGP (internal Border Gateway Protocol), którego wprowadzenie do systemu rutowania iBGP umożliwia znaczne zredukowanie ilości otwartych sesji BGP między innymi routerami (rezygnację z tzw. full-mesh) nosi nazwę:**
    a. **BGP Mirror**
    b. **Route Reflector Client (RRC) (X)**

25. **Liczba klas CoS (Class of Service), definiowanych przez podstawowy mechanizm implementacji QoS (Quality of Service) w Ethernet (czyli standard IEEE 802.1p), to:**
    a. **7**
    b. **255**
    c. **16**
    d. **8 (X)**
    e. **65536**

Pole Type Of Service (TOS), które wyznacza klasę, ma 3 bity, z czego $2^3 = 8$.

26. Wariant protokołu STP (Spanning Tree Protocol, IEEE 802.1d) pozwalający w technologii Ethernet na logiczne pogrupowanie sieci VLAN i budowanie mniejszej liczby drzew rozpinających (po 1 drzewie dla każdej grupy) to:
- **a. PVSTP (Per VLAN Spanning Tree Protocol)**
- **b. MSTP (Multiple Spanning Tree Protocol) (X)**
- **c. RSTP (Rapid Spanning Tree Protocol)**
- **d. SPB (Shortest Path Bridging)**

27. Nazwa procesu przekazywania wiedzy o trasach pomiędzy różnymi protokołami rutowania dynamicznego IP w routerach IP, to:
- **a. IP Route Spoofing**
- **b. redystrybucja (X)**

28. Co określa standard IEEE 802.1Q?
- **a. Private VLAN nadbudowana nad Ethernet**
- **b. wirtualne sieci LAN (VLAN) budowane w środowisku transportującym ramki (X)**
- **c. technologię tunelowania sieci VLAN o nazwie Q-in-Q**

29. Protokół umożliwiający konwersję adresu IP zdalnej stacji na jej adres MAC w Ethernet, to:
- **a. ARP (Address Resolution Protocol) (X)**
- **b. MLD (Multicast Listener Delivery)**
- **c. SLIP (Serial Line Internet Protocol)**

30. Dwie pod-warstwy definiowane w ramach warstwy drugiej modelu ISO-OSI to odpowiednio:
- **a. LLC (Logical Link Control) i MAC (Media Access Control) (X)**
- **b. LAN i WAN**
- **c. FDDI i CDDI**
- **d. LP (Link Pulse) i PHY (Physical)**

31. Rodzaj obszaru (area) w domenie OSPF (Open Shortest Path First) nie otrzymującego żadnych informacji o zewnętrznych (external) trasach routingu OSPF to:
- **a. internal**
- **b. backbone**
- **c. stub area (X)**
- **d. NSSA**

b. - backbone to właśnie zewnętrzne routery, budujące hierarchię sieci OSPF

32. Parametr o nazwie "Wielkość okna" (Window size), którego wartość przekazywana jest w datagramach potwierdzenia TCP w kierunku od odbiorcy do nadawcy ma na celu:

a. określenie długości następnego datagramu oraz wszystkich kolejnych
b. określenie ilości danych, jakie nadawca może w danej chwili wysłać (służy do sterowania przepływem) (X)
c. informowanie o wielkości datagramu, jaką może przyjąć host w aktualnym stanie (X)
d. określenie ilości danych w datagramie, w którym się znajduje - i w przypadku potwierdzeń TCP nie ma żadnego znaczenia

33. Dwa rodzaje obszarów (area) w protokole rutowania dynamicznego IS-IS (Intermediate System to Intermediate System), to:

a. intra-area i inter-area (X)
b. LAN i WAN
c. stub i backbone
d. Autonomous System i Internal System