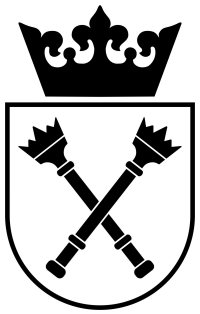
Uniwerystet Jagielloński

Pytania do egzaminu licencjackiego

na kierunku Informatyka

Małgorzata Dymek



Rok akademicki 2019/2020

Spis treści

1 Zasada indukcji matematycznej. 9

2 Porządki częściowe i liniowe. Elementy największe, najmniejsze, maksymalne i minimalne. 10

3 Relacja równoważności i zbiór ilorazowy. 12

4 Metody dowodzenia twierdzeń: wprost, nie wprost, przez kontrapozycję. 13

5 Metody numeryczne rozwiązywania równań nieliniowych: bisekcji, siecznych, Newtona. 14
5.1 Metoda połowienia (bisekcji) 14
5.2 Metoda siecznych 14
5.3 Metoda Newtona (stycznych) 15

6 Rozwiązywanie układów równań liniowych: metoda eliminacji Gaussa, metody iteracyjne Jacobiego i Gaussa-Seidla. 17
6.1 Metoda eliminacji Gaussa 17
6.2 Metoda iteracyjna Jacobiego 18
6.2.1 Algebraicznie 18
6.2.2 Macierzowo 18
6.3 Metoda iteracyjna Gaussa-Seidla 18
6.3.1 Algebraicznie 18
6.3.2 Macierzowo 18

7 Wartości i wektory własne macierzy: numeryczne algorytmy ich wyznaczania. 19
7.1 Metoda potęgowa 19

8 Interpolacja wielomianowa: metody Lagrange'a i Hermite'a. Efekt Rungego. 20
8.1 Interpolacja wielomianowa 20
8.2 Wzór interpolacyjny Lagrange'a 20
8.3 Interpolacja Hermite'a 21
8.3.1 Algorytm 21
8.4 Efekt Rungego 22

9 Zmienne losowe dyskretne. Definicje i najważniejsze rozkłady. 24

10 Zmienne losowe ciągłe. Definicje i najważniejsze rozkłady. 25

11 Łancuchy Markowa. Rozkład stacjonarny. 27

12 Testy statystyczne: test z, test t-Studenta, test chi-kwadrat. 28

13 Wzór Bayesa i jego interpretacja. 30

14 Istnienie elementów odwrotnych względem mnożenia w strukturze $(Zm, +, *)$ w zależności od liczby naturalnej m. Rozszerzony algorytm Euklidesa. 31
14.1 Algorytm Euklidesa 31
14.2 Rozszerzony algorytm Euklidesa 32

15 Ortogonalność wektorów w przestrzeni R_n; związki z liniową niezależnością. Metoda ortonormalizacji Grama-Schmidta 33
15.1 Ortonormalizacja Grama-Schmidta 35

16 Liczby Stirlinga I i II rodzaju i ich interpretacja. 36

17 Twierdzenia Eulera i Fermata; funkcja Eulera. 37

18 Konfiguracje i t-konfiguracje kombinatoryczne. 39

19 Cykl Hamiltona, obwód Eulera, liczba chromatyczna - definicje i twierdzenia. 41
19.1 Cykl Hamiltona 41
19.2 Obwód Eulera 41
19.3 Liczba chromatyczna 42

20 Algorytm Forda-Fulkersona wyznaczania maksymalnego przepływu. 43

21 Rozwiązywanie równań rekurencyjnych przy użyciu funkcji tworzących (generujących) oraz przy użyciu równania charakterystycznego. 45
21.1 Funkcje tworzące. 45
21.2 Równanie charakterystyczne. 48

22 Ciąg i granica ciągu liczbowego, granica funkcji. 49
22.1 Ciągi. 49
22.2 Funkcje. 51

23 Ciągłość i pochodna funkcji. Definicja i podstawowe twierdzenia. 54
23.1 Ciągłość. 54
23.2 Pochodna. 56

24 Ekstrema funkcji jednej zmiennej. Definicje i twierdzenia. 59

25 Całka Riemanna funkcji jednej zmiennej. 61

26 Pochodne cząstkowe funkcji wielu zmiennych; różniczkowalność i różniczka funkcji. 62

27 Ekstrema funkcji wielu zmiennych. Definicje i twierdzenia. 64

28 Twierdzenie o zmianie zmiennych w rachunku całkowym; współrzędne walcowe i sferyczne. 66

29 Metody dowodzenia poprawności pętli. 69

30 Odwrotna Notacja Polska: definicja, własności, zalety i wady, algorytmy. 70

31 Modele obliczen: maszyna Turinga. 70

32 Modele obliczen: automat skończony, automat ze stosem. 70

33 Złożoność obliczeniowa - definicja notacji: O, Ω, Θ. 71

34 Złożoność obliczeniowa - pesymistyczna i średnia. 72

35 Metoda "dziel i zwyciężaj"; zalety i wady. 73

36 Lista: ujęcie abstrakcyjne, możliwe implementacje i ich złożoności. 73
36.1 Implementacja za pomocą tablic (array) 73
36.2 Implementacja za pomocą Listy Wiązanej (Linked List) 74

37 Kolejka i kolejka priorytetowa: ujęcie abstrakcyjne, możliwe implementacje i ich złożoności. 77
37.1 Kolejka 77
37.2 Sposoby implementacji kolejki 78
37.3 Kolejka Priorytetowa 78
37.4 Sposoby implementacji kolejki priorytetowej 79

38 Algorytmy sortowania QuickSort i MergeSort: metody wyboru pivota w QS; złożoności. 80
38.1 QuickSort. 80
38.1.1 MergeSort. 81

39 Algorytm sortowania bez porównań (sortowanie przez zliczanie, sortowanie kubełkowe oraz sortowanie pozycyjne). 82
39.1 CountSort. 82
39.2 BucketSort. 82
39.3 RadixSort. 83

40 Reprezentacja drzewa binarnego za pomocą porządków (preorder, inorder, postorder). 84

41 Algorytmy wyszukiwania następnika i poprzednika w drzewach BST; usuwanie węzła. 85

42 B-drzewa: operacje i ich złożoność. 85

43 Drzewa AVL: rotacje, operacje z wykorzystaniem rotacji i ich złożoność. 85

44 Algorytmy przeszukiwania wszerz i w głąb w grafach. 85

45 Algorytmy wyszukiwania najkrótszej ścieżki (Dijkstry oraz Bellmana-Forda). 85

46 Programowanie dynamiczne: podział na podproblemy, porównanie z metodą "dziel i zwyciężaj". 85

47 Algorytm zachłanny: przykład optymalnego i nieoptymalnego wykorzystania. 85

48 Kolorowania wierzchołkowe (grafów planarnych) i krawędziowe grafów, algorytmy i ich złożoności. 85

49 Algorytmy wyszukiwania minimalnego drzewa rozpinającego: Boruvki, Prima i Kruskala. 85

50 Najważniejsze algorytmy wyznaczania otoczki wypukłej zbioru punktów w układzie współrzędnych (Grahama, Jarvisa, algorytm przyrostowy (quickhull)). 85

51 Problemy P, NP, NP-zupełne i zależności między nimi. Hipoteza P vs. NP. 85

52 Automat minimalny, wybrany algorytm minimalizacji. 85

53 Lemat o pompowaniu dla języków regularnych. 85

54 Warunki równoważne definicji języka regularnego: automat, prawa kongruencja syntaktyczna, wyrażenia regularne. 85

55 Automaty niedeterministyczne i deterministyczne (w tym ze stosem); determinizacja. 85

56 Problemy rozstrzygalne i nierozstrzygalne w teorii języków. 85

57 Klasy języków w hierarchii Chomsky'ego oraz ich zamkniętość ze względu na operacje boolowskie, homomorfizmy, itp. 85

58 Reprezentacja liczb całkowitych; arytmetyka. 86

59 Reprezentacja liczb rzeczywistych; arytmetyka zmiennopozycyjna. 86

60 Różnice w wywołaniu funkcji statycznych, niestatycznych i wirtualnych w C++. 86

61 Sposoby przekazywania parametrów do funkcji (przez wartość, przez referencję). Zalety i wady. 86

62 Wskaźniki, arytmetyka wskaźników, różnica między wskaźnikiem a referencją w C++. 86

63 Podstawowe założenia paradygmatu obiektowego: dziedziczenie, abstrakcja, enkapsulacja, polimorfizm. 86

64 Funkcje zaprzyjaźnione i ich związek z przeładowaniem operatorów w C++. 86

65 Programowanie generyczne na podstawie szablonów w języku C++. 86

66 Podstawowe kontenery w STL z szerszym omówieniem jednego z nich. 86

67 Obsługa sytuacji wyjątkowych w C++. 86

68 Obsługa plików w języku C. 86

69 Model wodospadu a model spiralny wytwarzania oprogramowania. 86

70 Diagram sekwencji i diagram przypadków użycia w języku UML. 86

71 Klasyfikacja testów. 86

72 Model Scrum: struktura zespołu, proces wytwarzania oprogramowania, korzyści modelu. 86

73 Wymagania w projekcie informatycznym: klasyfikacja, źródła, specyfikacja, analiza. 86

74 Analiza obiektowa: modele obiektowe i dynamiczne, obiekty encjowe, brzegowe i sterujące. 86

75 Wzorce architektury systemów. 86

76 Relacyjny model danych, normalizacja relacji (w szczególności algorytm doprowadzenia relacji do postaci Boyce'a-Codda), przykłady. 87

77 Indeksowanie w bazach danych: drzewa B+, tablice o organizacji indeksowej, indeksy haszowe, mapy binarne. 87

78 Podstawowe cechy transakcji (ACID). Metody sterowania współbieżnością transakcji, poziomy izolacji transakcji, przykłady. 87

79 Złączenia, grupowanie, podzapytania w języku SQL. 87

80 Szeregowalność harmonogramów w bazach danych. 87

81 Definicja cyfrowego układu kombinacyjnego - przykłady układów kombinacyjnych i ich implementacje. 87

82 Definicja cyfrowego układu sekwencyjnego - przykłady układów sekwencyjnych i ich implementacje. 87

83 Minimalizacja funkcji logicznych. 87

84 Programowalne układy logiczne PLD (ROM, PAL, PLA). 87

85 Schemat blokowy komputera (maszyna von Neumanna). 87

86 Zarządzanie procesami: stany procesu, algorytmy szeregowania z wywłaszczaniem. 87

87 Muteks, semafor, monitor jako narzędzia synchronizacji procesów. 87

88 Pamięć wirtualna i mechanizm stronicowania. 87

89 Systemy plikowe - organizacja fizyczna i logiczna (na przykładzie wybranego systemu uniksopodobnego). 87

90 Model ISO OSI. Przykłady protokołów w poszczególnych warstwach. 87

91 Adresowanie w protokołach IPv4 i IPv6. 87

92 Najważniejsze procesy zachodzące w sieci komputerowej od momentu wpisania adresu strony WWW do wyświetlenia strony w przeglądarce (komunikat HTTP, segment TCP, system DNS, pakiet IP, ARP, ramka). 87

93 Działanie przełączników Ethernet, sieci VLAN, protokół STP. 87

94 Rola routerów i podstawowe protokoły routingu (RIP, OSPF). 87

95 Szyfrowanie z kluczem publicznym, podpis cyfrowy, certyfikaty. 87

Matematyczne podstawy informatyki

1 Zasada indukcji matematycznej.

Twierdzenie 1.1 ***Zasada indukcji matematycznej.*** *Niech T(n) - funkcja/forma zdaniowana zmiennej $n \in \mathbb{N}$. Jeżeli:*

1. *zachodzi $T(0)$*
2. *$\forall n \in \mathbb{N} \;\; T(n) \Rightarrow T(n+1)$*

to wtedy $T(n)$ jest prawdziwa dla każdego $n \in \mathbb{N}$.

Schemat dowodu:

Niech $M = \{n \in \mathbb{N} : \; T(n) \; zachodzi\}$, $M \subset \mathbb{N}$. Wtedy wg twierdzenia:

1. $\Rightarrow 0 \in M$
2. $\Rightarrow n \in M \Rightarrow n+1 \in M$

Zatem z **aksjomatu 5** wnioskujemy M = N.

Twierdzenie 1.2 ***Aksjomat 5 liczb naturalnych (Peano)****. Niech będzie dany zbiór, którego elementami są liczby naturalne, o następujących właściwościach:*

1. *J jest elementem tego zbioru.*
2. *Wraz z liczbą naturalną należącą do tego zbioru, należy do niego również jej następnik.*

Wtedy zbiór ten zawiera wszystkie liczby naturalne. $(Z \subset \mathbb{N}) \wedge (J \in Z) \wedge (\forall k \in \mathbb{N} k* \in Z) \Rightarrow Z = \mathbb{N}$.

Przykład: $2^1 + 2^2 + \cdots + 2^n = 2^{n+1} - 2$, Nierówność Bernoulliego *dla* $h \geqslant -1 \;\; (1+h)^2 \geqslant 1 + n * h, \;\; \forall n \in \mathbb{N}^+$, $1 + 2 + \cdots + n = \frac{n(n+1)}{2} \forall n \in \mathbb{N}$

2 Porządki częściowe i liniowe. Elementy największe, najmniejsze, maksymalne i minimalne.

Definicja 2.1 ***Częściowy porządek***. *Niech X zbiór, $R \subset X \times X$ relacja. Wtedy R nazywamy* ***relacją częściowego porządku*** *w $X \Leftrightarrow$*

1. *R* ***zwrotna*** *($\forall x \in X \; xRx$),*
2. *R* ***przechodnia*** *($\forall x, y, z \in X \; xRy \wedge yRz \Rightarrow xRz$),*
3. *R* ***antysymetryczna*** *($\forall x, y \in X \; xRy \wedge yRx \Rightarrow x = y$).*

Piszemy: $\leqslant, \preceq, \prec$ gdy $\neq$. Przykład: $(\mathbb{R}, \leqslant)$, gdzie $x \preceq y \Leftrightarrow x \leqslant y \; \forall x, y \in \mathbb{R}$. Wtedy $(\mathbb{R}, \preceq)$ jest cześciowym porządkiem.

Jeżeli $(X, \mathbb{R})$ jest częściowym porządkiem, to elementy $x, y \in X$ nazywamy ***porównywalnymi*** *$\Leftrightarrow \; xRy \vee yRx$.*

Diagram Hassego *- graf skierowany przedstawiający częściowy porządek w zbiorze, w odpowiedni sposób przedstawiony graficznie.*

Definicja 2.2 ***Liniowy porządek***. *Niech X zbiór, $R \subset X \times X$ relacja. Wtedy R nazywamy* ***relacją częściowego porządku*** *w $X \Leftrightarrow$*

1. *R* ***zwrotna*** *($\forall x \in X \; xRx$),*
2. *R* ***przechodnia*** *($\forall x, y, z \in X \; xRy \wedge yRz \Rightarrow xRz$),*
3. *R* ***antysymetryczna*** *($\forall x, y \in X \; xRy \wedge yRx \Rightarrow x = y$),*
4. *R* ***spójna*** *($\forall x, y \in X \; xRy \vee yRx \vee x = y$).*

Definicja 2.3 *Niech $\preceq$ jest relacją częściowego porządku wówczas element m jest to:*

1. ***Element maksymalny**, jeśli* $\forall a \in A \;\; m \preceq a \;\Rightarrow\; a = m$,
2. ***Element minimalny**, jeśli* $\forall a \in A \;\; a \preceq m \;\Rightarrow\; a = m$,
3. ***Element największy**, jeśli* $\forall a \in A \;\; a \preceq m$,
4. ***Element najmniejszy**, jeśli* $\forall a \in A \;\; m \preceq a$.

Przykłady - sprawdź czy porządek: $xRy \Leftrightarrow x|y$

3 Relacja równoważności i zbiór ilorazowy.

Definicja 3.1 *Relację $R \subset X \times X$ nazywamy* ***relacją równoważości*** $\Leftrightarrow$ *relacja R jest:*

1. ***zwrotna*** *($\forall x \in X \; xRx$),*
2. ***symetryczna*** *($\forall x, y \in X \; xRy \Rightarrow yRx$),*
3. ***przechodnia*** *($\forall x, y, z \in X \; xRy \wedge yRz \Rightarrow xRz$).*

Definicja 3.2 *Niech $R \subset X \times X$ będzie relacją równoważności, $X \neq \varnothing$, $x \in X$.* ***Klasą abstrakcji*** *elementu x (wzgledem relacji R) nazywamy:*
$[x]_R = y \in X : xRy$
Element x nazywamy reprezentantem klasy abstrakcji $[x]_R$.

Definicja 3.3 ***Zbiorem ilorazowym*** *zbioru X przez relację R nazywamy zbiór wszystkich klas abstrakcji.*
$X/R = [y]_R : x \in X \quad \subset X$

Przykład: $xRy \;\Leftrightarrow x \equiv_3 y$.

4 Metody dowodzenia twierdzeń: wprost, nie wprost, przez kontrapozycję.

Dla prawdziwości zdania:

$$p \Rightarrow q$$

Definicja 4.1 ***Dowód wprost****. Metoda dowodu wprost polega na założeniu, że p jest prawdą i pokazaniu, że wówczas q jest prawdą.*

Definicja 4.2 ***Dowód nie wprost****. Metoda dowodu nie wprost opiera się na następującej tautologii rachunku zdań, zwanej prawem kontrapozycji:*

$$(p \Rightarrow q) \Leftrightarrow (\neg q \Rightarrow \neg p).$$

Zatem stosując tę metodę zakładamy, że q jest zdaniem fałszywym i pokazujemy, że p jest również zdaniem fałszywym.

Definicja 4.3 ***Dowód przez zaprzeczenie*** *Metoda dowodu przez zaprzeczenie opiera się na następującej tautologii rachunku zdań:*

$$(p \Rightarrow q) \;\Leftrightarrow\; (\neg p \vee q) \;\Leftrightarrow\; \neg(p \wedge \neg q)$$

Stosując to podejście zakładamy, że p jest prawdą a q fałszem i pokazujemy, że prowadzi to do sprzeczności, to znaczy, pokazujemy że $(p \wedge \neg q)$ jest fałszem.

5 Metody numeryczne rozwiązywania równań nieliniowych: bisekcji, siecznych, Newtona.

5.1 Metoda połowienia (bisekcji)

Założenia:

- f jest funkcją ciągłą w przedziale $[a, b]$,
- $f(a)f(b) < 0$.

Z własności Darboux funkcji ciągłych, funkcja f ma miejsce zerowe w przedziale $[a, b]$.

Definicja 5.1 ***Algorytm bisekcji*** *polega na obliczeniu $f(c_k)$, gdzie $c_k = \frac{a_k+b_k}{2}$ i zastąpieniu przez c_k tej z liczb a_k, b_k dla której funkcja f ma taki sam znak.*

$$(a_{k+1}, b_{k+1}) = (c_k, b_k) \text{ jeżeli } f(a_k)f(c_k) > 0$$
$$(a_{k+1}, b_{k+1}) = (a_k, c_k) \text{ jeżeli } f(b_k)f(c_k) > 0$$

Jeżeli $f(c_k) = 0$ to kończymy obliczenia.

- Metoda bisekcji jest **niezawodna**, ale **wolno zbieżna**.
- W każdym kroku szerokość przedziału jest dzielona przez dwa.
- Kryteria zakończednia obliczeń:
  - osiągnięto dokładność $\delta : |e_n| < \delta$
  - wartość funkcji jest bliska 0: $f(c_n) < \epsilon$
  - wykonano M iteracji
- Algorytm bisekcji w k-tej iteracji przybliża rozwiązanie α z **dokładnością:** $|x_k - \alpha| \leqslant \frac{|b-a|}{2^k}$.

5.2 Metoda siecznych

Założenia:

- f jest funkcją ciągłą w przedziale $[a, b]$,

- $f(a)f(b) < 0$.

$$x_{k+1} = x_k - \frac{f(x_k)(x_k - x_{k-1})}{f(x_k) - f(x_{k-1})}$$

gdzie $x_0 = a$, $x_1 = b$.

- W metodzie siecznych rezygnujemy z założenia, że funkcja na końcach przedziału ma rózne znaki.
- Należy kontrolować zachowanie otrzymanego ciągu. **Może się zdarzyć, że metoda wyprodukuje ciąg rozbieżny!**
- korzyćci płynące ze stosowania tej metody to zdecydowanie **szybsza zbieżność** ciągu iteracji, jeśli x_n, x_{n+1} juz są dobrymi **przybliżeniami pierwiastka**.

5.3 Metoda Newtona (stycznych)

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Zmodyfikowana metoda Newtona:

$$x_{n+1} = x_n - \frac{f'(x_n) \pm \sqrt{(f'(x_n))^2 - 2f(x_n)f''(x_n)}}{f''(x_n)}$$

- Metoda Newtona w wielu sytuacjach daje **bardzo szybką zbieżność**.
- Podstawową wadą tej metody jest konieczność obliczenia pochodnej funkcji.
- Zmodyfikowana metoda Newtona daje bardzo szybką zbieżność, jeśli x_n jest już dobrym przybliżeniem pierwiastka.
- Koszt wyznaczenia kolejnego przybliżenia w zmodyfikowanej metodzie może przerastać korzyści płynące z szybszej zbieżności.

Wielowymiarowa metoda Newtona.

Jeśli $f = (f_1, f_2, \ldots, f_n) : \mathbb{R}^n \to \mathbb{R}^n$ jest różniczkowalną funkcją wielu zmiennych możemy przybliżyć ją lokalnie:

$$f(x) \approx f(x_0) + Df(x_0)(x - x_0)$$

gdzie

$$Df(x_0) = \begin{bmatrix} \frac{\delta f_1}{\delta x_1}(x_0) & \frac{\delta f_1}{\delta x_2}(x_0) & \ldots & \frac{\delta f_1}{\delta x_n}(x_0) \\ \frac{\delta f_2}{\delta x_1}(x_0) & \frac{\delta f_2}{\delta x_2}(x_0) & \ldots & \frac{\delta f_2}{\delta x_n}(x_0) \\ \ldots & \ldots & \ldots & \ldots \\ \frac{\delta f_n}{\delta x_1}(x_0) & \frac{\delta f_n}{\delta x2}(x_0) & \ldots & \frac{\delta f_n}{\delta x_n}(x_0) \end{bmatrix}$$

Rozwiązując równanie $f(x_0) + Df(x_0)(x - x_0) = 0$ otrzymujemy:

$$x^{(i+1)} = x^{(i)} - [Df(x^{(i)})]^{-1} f(x^{(i)})$$

$$[Df(x^{(i)})](x^{(i+1)} - x^{(i)}) = -f(x^{(i)})$$

6 Rozwiązywanie układów równań liniowych: metoda eliminacji Gaussa, metody iteracyjne Jacobiego i Gaussa-Seidla.

6.1 Metoda eliminacji Gaussa

Obliczając rząd macierzy metodą Gaussa należy za pomocą operacji elementarnych na wierszach sprowadzić macierz do macierzy schodkowej. Wtedy wszystkie niezerowe wiersze są liniowo niezależne i można łatwo odczytać rząd macierzy.

$$\begin{bmatrix} 1 & -1 & 2 & 2 \\ 2 & -2 & 1 & 0 \\ -1 & 2 & 1 & -2 \\ 2 & -1 & 4 & 0 \end{bmatrix} \overset{w_2-2w_1, w_3+w_1, w_4-2w_1}{\sim} \begin{bmatrix} 1 & -1 & 2 & 2 \\ 0 & 0 & -3 & -4 \\ 0 & 1 & 3 & 0 \\ 0 & 1 & 0 & -4 \end{bmatrix} \overset{w_2 \leftrightarrow w_3}{\sim} \begin{bmatrix} 1 & -1 & 2 & 2 \\ 0 & 1 & 3 & 0 \\ 0 & 0 & -3 & -4 \\ 0 & 1 & 0 & -4 \end{bmatrix} \sim$$

$$\overset{w4-w_2}{\sim} \begin{bmatrix} 1 & -1 & 2 & 2 \\ 0 & 1 & 3 & 0 \\ 0 & 0 & -3 & -4 \\ 0 & 0 & -3 & -4 \end{bmatrix} \overset{w4-w_3}{\sim} \begin{bmatrix} 1 & -1 & 2 & 2 \\ 0 & 1 & 3 & 0 \\ 0 & 0 & -3 & -4 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Metody iteracyjne

Ogólna postać metody iteracyjnej:

$$Ax = b$$

$$Qx^{n+1} = (Q - A)x^n + b = \tilde{b}$$

$$x^0 = (0, 0, 0)$$

$$\begin{bmatrix} 5 & -2 & 3 \\ 2 & 4 & 2 \\ 2 & -1 & -4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 10 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{cases} 5x_1 + (-2)x_2 + 3x_3 = 10 \\ 2x_1 + 4x_2 + 2x_3 = 0 \\ 2x_1 + (-1)x_2 + (-4)x_3 = 0 \end{cases}$$

6.2 Metoda iteracyjna Jacobiego

6.2.1 Algebraicznie

$$\begin{cases} x_1^{N+1} = \frac{1}{5}(10 + 2x_2^N - 3x_3^N) \\ x_2^{N+1} = \frac{1}{4}(-2x_1^N - 2x_3^N) \\ x_3^{N+1} = -\frac{1}{4}(x_2^N - 2x_1^N) \end{cases}$$

6.2.2 Macierzowo

$$Q = D \ \text{ (diagonalna)}$$

6.3 Metoda iteracyjna Gaussa-Seidla

6.3.1 Algebraicznie

$$\begin{cases} x_1^{N+1} = \frac{1}{5}(10 + 2x_2^N - 3x_3^N) \\ x_2^{N+1} = \frac{1}{4}(-2x_1^N - 2x_3^{N+1}) \\ x_3^{N+1} = -\frac{1}{4}(x_2^{N+1} - 2x_1^{N+1}) \end{cases}$$

6.3.2 Macierzowo

$$Q = L + D \ \text{ (diagonalna i dolnotrójkątna)}$$

7 Wartości i wektory własne macierzy: numeryczne algorytmy ich wyznaczania.

$$A \in \mathbb{C}^{n \times n} \quad \text{szukamy} \quad \lambda \in \mathbb{C}$$

Definicja 7.1 *Jeżeli*

$$Ax = \lambda x$$

dla $x \neq 0$ *to* λ *jest* ***wartością własną*** A*, a* x *-* ***wektorem własnym*** A *odpowiadającym* λ*.*

7.1 Metoda potęgowa

$$Ax = \lambda x$$
$$x^{N+1} = A^{N+1}x^0$$

8 Interpolacja wielomianowa: metody Lagrange'a i Hermite'a. Efekt Rungego.

8.1 Interpolacja wielomianowa

Interpolacja wielomianowa jest metodą numeryczną przybliżania funkcji. Wielomian p o możliwie najniższym stopniu interpoluje wartości y_k w węzłach x_k gdy dla danych $n+1$ punktów (x_i, y_i)

$$p(x_i) = y_i \quad (0 \leqslant i \leqslant n)$$

Jeśli są to wartości pewnej funkcji f to mówimy też że p interpoluje f.

8.2 Wzór interpolacyjny Lagrange'a

Wyrażamy wielomian p jako sumę

$$p(x) = \sum_{k=0}^{n} y_i l_k(x)$$

w której l_k są wielomianami zależnymi od węzłów $x_0, x_1, \ldots x_n$ ale nie od wartości $y_0, y_1, \ldots, y_n$. Gdyby jedna z nich, mianowicie y_i, była równa 1, a pozostałe by znikały, to mielibyśmy równość

$$\delta_{ij} = p_n(x_j) = \sum_{k=0}^{n} y_i l_k(x_j) = \sum_{k=0}^{n} \delta_k l_k(x_j) = l_i(x_j) \quad (0 \leqslant j \leqslant n)$$

($\delta_i j$ jest deltą Kroneckera). Łatwo znaleźć wielomian l_i o tej własności. Jego zerami są wszystkie węzły oprócz x_i, czyli dla pewniej stałej c jest

$$l_i(x) = c(x - x_0) \ldots (x - x_{i-1})(x - x_{i+1}) \ldots (x - x_n)$$

a c wynika z warunku $l_i(x_i) = 1$:

$$l_i(x) = \prod_{j=0 \wedge j \neq i}^{n} \frac{x - x_j}{x_i - x_j}. \quad (0 \leqslant i \leqslant n).$$

Zatem wzór interpolacyjny Lagrange'a ma postać:

$$w(x) = \sum_{i=0}^{n} y_i \prod_{j=0 \wedge j \neq i}^{n} \frac{x - x_j}{x_i - x_j}.$$

8.3 Interpolacja Hermite'a

Interpolacja umożliwiająca znalezienie wielomianu przybliżającego według wartości

$$y_1 = f(x_1), y_2 = f(x_2), \ldots, y_n = f(x_n)$$

na n zadanych węzłach $x_1, x_2, \ldots, x_n$ oraz na wartościach pochodnych na wybranych węzłach

$$f'(x_1), f''(x_1), \ldots, f^{(k_1)}(x_1), \ldots, f'(x_n), \ldots, f^{(k_n)}(x_n).$$

Węzeł zadany bez pochodnej jest węzłem pojedynczym, a węzeł z zadanymi pochodnymi $1, 2, \ldots, k$ jest węzłem $k+1$-krotnym.

8.3.1 Algorytm

Algorytm jest podobny jak przy interpolacji Newtona. Kolumnę wypełnia się wszystkimi wartościami węzłów (jeżeli węzeł jest k-krotny, to umieszczamy go w tabeli k razy)

x_i	$f(x_i)$
x_0	$f(x_0)$
x_1	$f(x_1)$
x_2	$f(x_2)$
$\vdots$	$\ddots$
x_n	$f(x_n)$

Następnie dopisuje się do każdej kolumny kolejne różnice dzielone, z tym wyjątkiem, że przy węzłach k-krotnych, $k > 1$, gdzie, de facto, nie można obliczyć różnicy dzielonej, podstawia się wartości kolejnych pochodnych na węzłach podzielone przez silnię z liczby tych samych węzłów. (W tabeli przedstawiony jest 3-krotny węzeł x_1).

x_i	$f(x_i)$	$f[x_{i-1}]$	$f[x_{i-2}, x_{i-1}, x_i]$
x_0	$f(x_0)$		
x_1	$f(x_1)$	$f[x_0, x_1]$	
x_1	$f(x_1)$	$f'(x_1)$	$f[x_0, x_1, x_1]$
x_1	$f(x_1)$	$f'(x_1)$	$\frac{f''(x_1)}{2!}$
x_2	$f(x_2)$	$f[x_1, x_2]$	$f[x_1, x_1, x_2]$
$\vdots$	$\vdots$	$\vdots$	
x_n	$f(x_n)$	$f[x_{n-1}, x_n]$	$f[x_{n-2}, x_{n-1}, x_n]$

Tabelę uzupełnia się do końca jak przy interpolacji Newtona, uznając ciągłe pochodne na węzłach wielokrotnych jako różnice dzielone rzędu drugiego.

x_i	$f(x_i)$	$f[x_{i-1}]$	$f[x_{i-2}, x_{i-1}, x_i]$	$f[x_{i-3}, x_{i-2}, x_{i-1}, x_i]$	$\dots$	$f[x_{i-n}, \dots, x_i]$
x_0	$f(x_0)$					
x_1	$f(x_1)$	$f[x_0, x_1]$				
x_1	$f(x_1)$	$f'(x_1)$	$f[x_0, x_1, x_1]$			
x_1	$f(x_1)$	$f'(x_1)$	$\frac{f''(x_1)}{2!}$	$f[x_0, x_1, x_1, x_1]$		
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	
x_n	$f(x_n)$	$f[x_{n-1}, x_n]$	$f[x_{n-2}, x_{n-1}, x_n]$	$f[x_{n-3}, x_{n-2}, x_{n-1}, x_n]$	$\dots$	$f[x_0, \dots, x_n]$

Definiując a_i jako wartości na przekątnej, $i = 1, 2, 3, \dots, m$, gdzie m to suma krotności węzłów, otrzymuje się wielomian:

$$w(x) = \sum_{i=0}^{m} a_i \prod_{j=0}^{i-1} (x - \bar{x}_j),$$

gdzie $\bar{x}_i = x_1, x_1, \dots, x_1, x_2, x_2, \dots, x_2, \dots, x_n$, przy czym każdy k-krotny węzeł występuje k razy.

8.4 Efekt Rungego

Efekt Rungego - pogorszenie jakości interpolacji wielomianowej, mimo zwiększenia liczby jej węzłów. Początkowo ze wzrostem liczby węzłów n przybliżenie poprawia się, jednak po dalszym wzroście n, zaczyna się pogarszać, co jest szczególnie widoczne na końcach przedziałów.

Takie zachowanie się wielomianu interpolującego jest zjawiskiem typowym dla interpolacji za pomocą wielomianów wysokich stopni przy stałych odległościach węzłów. Występuje ono również, jeśli interpolowana funkcja jest nieciągła albo odbiega znacząco od funkcji gładkiej.

Aby uniknąć tego efektu, stosuje się interpolację z węzłami coraz gęściej upakowanymi na krańcach przedziału interpolacji. Np. węzłami interpolacji n-punktowej wielomianowej powinny być miejsca zerowe wielomianu Czebyszewa n-tego stopnia.

9 Zmienne losowe dyskretne. Definicje i najważniejsze rozkłady.

Definicja 9.1 ***Zmienne dyskretne.***

$$P(x) = P(X = x)$$

Obliczanie prawdopodobieństwa: $P(X \in A) = \sum_{x \in A} P(x)$.
Skumulowana funkcja rozkładu: $F(x) = P(X \leqslant x) = \sum_{y \leqslant x} P(y)$
Całkowite prawdopodobieństwo: $\sum_x P(x) = 1$.
Wartość oczekiwana: $EX = \sum_x xP(x)$.
Wariancja: $VarX = \sigma^2 = E[(X - \mu)^2]$.

Rozkład	P(x)	EX	VarX	
Bernoulli(p)	$P(x) = \begin{cases} p, & \text{for } x = 1 \\ q = (1-p), & \text{for } x = 0 \end{cases}$	p	pq	próba
Binomial(n, p)	$P(x) = \binom{n}{x} p^x (1-p)^{n-x}$ for $x = 0, 1, \ldots$	np	npq	liczba sukcesów z n prób
Geometric(p)	$P(x) = (1-p)^{x-1} p$ for $x = 1, 2, \ldots$ $P(X > k) = (1-p)^k$	$\frac{1}{p}$	$\frac{1-p}{p^2}$	liczba prób do sukcesu
Poiss(λ)	$P(x) = e^{-\lambda} \frac{\lambda^x}{x!}$ for $x = 0, 1, \ldots$	λ	λ	rozkład zdarzeń rzadkich

10 Zmienne losowe ciągłe. Definicje i najważniejsze rozkłady.

Definicja 10.1 ***Zmienne ciągłe.***

$$f(x) = F'(x)$$

Obliczanie prawdopodobieństwa: $P(X \in A) = \int_A f(x)dx.$
Skumulowana funkcja rozkładu: $F(x) = P(X \leqslant x) = \int_{-\infty}^{x} f(y)dy.$
Całkowite prawdopodobieństwo: $\int_{-\infty}^{\infty} f(x)dx = 1.$
Wartość oczekiwana: $EX = \int xf(x)dx.$
Wariancja: $VarX = \int_{-\infty}^{\infty}(x-\mu)^2 f(x)dx.$

Rozkład	f(x), F(x)	EX	VarX	
Unif(a,b)	$f(x) = \frac{1}{b-a}$ for $a \leqslant x \leqslant b$ $F(x) = \begin{cases} 0, & \text{for } x < a \\ \frac{x-a}{b-a}, & \text{for } a \leqslant x < b \\ 1, & \text{for } x \geqslant b \end{cases}$	$\frac{a+b}{2}$	$\frac{(b-a)^2}{12}$	
Exp(λ)	$f(x) = \lambda e^{-\lambda x}$ for $x \geqslant 0$ $F(x) = 1 - e^{-\lambda x}$	$\frac{1}{\lambda}$	$\frac{1}{\lambda^2}$	modelowanie czasu, brak pamięci
Gamma(α, λ)	$f(x) = \frac{\lambda^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\lambda x}$ $F(x) = \frac{\lambda^\alpha}{\Gamma(\alpha)} \int_0^x t^{\alpha-1} e^{-\lambda t} dt$	$\frac{\alpha}{\lambda}$	$\frac{\alpha}{\lambda^2}$	łączny czas α niezależnych zdarzeń $\sim Exp(\lambda)$
N(μ, σ)	$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$ $F(x) = \Phi(x)$ dla N(0,1)	μ	σ^2	

$$Bin(n,p) \approx Poiss(\lambda) \tag{1}$$

$$P(T \leqslant t) = P(X \geqslant \alpha) \tag{2}$$

$$T \sim Gamma(\alpha, \lambda), X \sim Poiss(\lambda t)$$

$$Binomial(n, p) = N(np, \sqrt{np(1-p)}) \tag{3}$$

$$X_i \sim Bernoulli(p), S_n = \sum_{i=1}^{n} X_i, 0.05 \leqslant p \leqslant -0.95$$

11 Łancuchy Markowa. Rozkład stacjonarny.

$$P = \begin{bmatrix} p_{11} & p_{12} & \dots & p_{1n} \\ p_{21} & p_{22} & \dots & p_{2n} \\ \dots & \dots & \dots & \dots \\ p_{n1} & p_{n2} & \dots & p_{nn} \end{bmatrix}$$

Rozkład w czasie h: $P_h = P_0 * P^h$
Rozkład stacjonarny: $\pi P = \pi$, $\sum \pi_i = 1$

12 Testy statystyczne: test z, test t-Studenta, test chi-kwadrat.

Rozkład t-studenta

$t = \frac{\hat{\theta}=\theta}{s(\hat{\theta})} \leftarrow$ zastępujemy $Std(\hat{\theta})$ przez $s(\hat{\theta})$, n-1 stopni swobody

$$\bar{X} \pm t_{\frac{\alpha}{2}}^{(n-1)} \frac{s(\hat{\theta})}{\sqrt{n}}$$

Z-testy

Hipoteza zerowa	Parametr, estymator	jeśli H_0 jest prawdziwa:		Statystyka
H_0	$\theta, \hat{\theta}$	$E(\hat{\theta})$	$Var(\hat{\theta})$	$Z = \frac{\hat{\theta}-\theta_0}{\sqrt{Var(\hat{\theta})}}$
$\mu = \mu_0$	μ, $\bar{X}$	μ_0	$\frac{\sigma^2}{n}$	$Z = \frac{\bar{X}-\mu_0}{\frac{\sigma}{\sqrt{n}}}$
$p = p_0$	p, $\hat{p}$	p_0	$\frac{p_0(1-p_0)}{n}$	$Z = \frac{\hat{p}-p_0}{\sqrt{\frac{\hat{p}(1-\hat{p})}{n}}}$
$\mu_X - \mu_Y = D$	$\mu_X - \mu_Y$, $\bar{X} - \bar{Y}$	D	$\frac{\sigma_X^2}{n} + \frac{\sigma_Y^2}{n}$	$Z = \frac{\bar{X}+\bar{Y}-D}{\sqrt{\frac{\sigma_X^2}{n}+\frac{\sigma_Y^2}{m}}}$
$p_1 - p_2 = D$	$p_1 - p_2$, $\hat{p_1} - \hat{p_2}$	D	$\frac{p_1(1-p_1)}{n} + \frac{p_2(1-p_2)}{m}$	$Z = \frac{\hat{p_1}-\hat{p_2}-D}{\sqrt{\frac{\hat{p_1}(1-\hat{p_1})}{n}+\frac{\hat{p_2}(1-\hat{p_2})}{m}}}$
$p_1 = p_2$	$p_1 - p_2$, $\hat{p_1} - \hat{p_2}$	0	$p(1-p)(\frac{1}{n}+\frac{1}{m})$ gdzie $p = p_1 = p_2$	$Z = \frac{\hat{p_1}-\hat{p_2}}{\sqrt{\hat{p}(1-\hat{p})(\frac{1}{n}+\frac{1}{m})}}$ gdzie $\hat{p} = \frac{n\hat{p_1}+m\hat{p_2}}{n+m}$

T-testy

Hipoteza zerowa	Warunki	Statystyka	Stopnie swobody
$\mu = \mu_0$	Rozmiar próby n; nieznana σ	$t = \frac{\bar{X}-\mu_0}{\frac{s}{\sqrt{n}}}$	$n-1$
$\mu_X - \mu_Y = D$	Rozmiary prób n, m nieznane, równe $\sigma_X = \sigma_Y$	$t = \frac{\bar{X}-\bar{Y}-D}{s_p\sqrt{\frac{1}{n}+\frac{1}{m}}}$	$n+m-2$
$\mu_X - \mu_Y = D$	Rozmiary prób n, m; nieznane, różne $\sigma_X \neq \sigma_Y$	$t = \frac{\bar{X}-\bar{Y}-D}{\sqrt{\frac{s_X^2}{n}+\frac{s_Y^2}{m}}}$	aproksymacja Satterthwaite

Rozkład obserwacji o rozkładzie normalnym i wspólnej wariancji σ^2

$$\frac{(n-1)s^2}{\sigma^2} = \sum_{i=1}^{n}\left(\frac{X_i - \bar{X}}{\sigma}\right)^2 \sim Chi - square(n-1) \sim Gamma\left(\frac{n-1}{2}, \frac{1}{2}\right)$$

Zatem przedział ufności:

$$\left[\frac{(n-1)s^2}{\chi^2_{\frac{\alpha}{2}}}, \frac{(n-1)s^2}{\chi^2_{1-\frac{\alpha}{2}}}\right]$$

Testy Chi kwadrat

H_0	H_A	Test statistic	Rejection region	P-value
$\sigma^2 = \sigma_0^2$	$\sigma^2 > \sigma_0^2$ $\sigma^2 < \sigma_0^2$ $\sigma^2 \neq \sigma_0^2$	$\frac{(n-1)s^2}{\sigma_0^2}$	$\chi^2_{obs} > \chi^2_{\alpha}$ $\chi^2_{obs} < \chi^2_{\alpha}$ $\chi^2_{obs} \geqslant \chi^2_{\frac{\alpha}{2}}$ or $\chi^2_{obs} \leqslant \chi^2_{\frac{\alpha}{2}}$	$P\chi^2 \geqslant \chi^2_{obs}$ $P\chi^2 \leqslant \chi^2_{obs}$ $2min(P\chi^2 \geqslant \chi^2_{obs}, P\chi^2 \leqslant \chi^2_{obs})$

Statystyka Chi-kwadrat

$$\chi^2 = \sum_{k=1}^{N}\frac{(Obs(k) - Exp(k))^2}{Exp(k)}, R = [\chi^2_{\alpha}, +\infty], P = P\chi^2 \geqslant \chi^2_{obs}$$

Rule of thumb: $Exp(k) \geqslant 5$ for all $k = 1, \ldots, N$.

Test Chi-kwadrat niezależności A i B

$$\chi^2_{obs} = \sum_{i=1}^{k}\sum_{j=1}^{m}\frac{(Obs(i,j) - \hat{Exp}(i,j))^2}{\hat{Exp}(i,j)}, \hat{Exp}(i,j) = \frac{(n_{i.})(n_{.j})}{n}$$

13 Wzór Bayesa i jego interpretacja.

Prawdopodobieństwo warunkowe

$$P(E|F) = \frac{P(E \cap F)}{P(F)} \tag{4}$$

$$P(E_1 \cap \cdots \cap E_n) = \prod_{i=1}^{n} P(X_i|X_1, \ldots, X_{i-1}) \tag{5}$$

$$P(E) = \sum_{i=1}^{n} P(E|F_i)P(F_i) \text{ dla } \bigcap_{i=1}^{n} F_i = \Omega \tag{6}$$

Wzór Bayesa:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \quad \text{przy } P(B) > 0$$

dowód:

$$P(B|A) = \frac{P(A \cap B)}{P(A)} \;\rightarrow\; P(A \cap B) = P(B|A) * P(A)$$

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \;\rightarrow\; P(A|B) * P(B) = P(A \cap B) = P(B|A) * P(A) \;\; / : P(B)$$

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

14 Istnienie elementów odwrotnych względem mnożenia w strukturze $(Zm, +, *)$ w zależności od liczby naturalnej m. Rozszerzony algorytm Euklidesa.

Twierdzenie 14.1 ***Tożsamość Bézouta***
Dla niezerowych liczb całkowitych a oraz b o największym wspólnym dzielniku d, istnieją liczby całkowite x oraz y, nazywane współczynnikami Bézouta, które spełniają równanie

$$ax + by = d$$

Definicja 14.2 ***Elementy odwrotne modulo***
W strukturze $(Z_m, +, *)$ element odwrotny względem mnożenia dla x to taki x^{-1}, że:

$$x \cdot x^{-1} = 1 \ mod \ m$$

Element odwrotny istnieje, jeśli $NWD(x, m) = 1$

Aby obliczyć element odwrotny do x w Z_m należy użyć rozszerzonego algorytmu Euklidesa.

14.1 Algorytm Euklidesa

Oblicza największy wspólny dzielnik a, b

```
def euclid(a, b):
    while b != 0:
        r = a % b
        a, b = b, r
    return a
```

Wersja z odejmowaniem:

```
def euclid(a, b):
    while a != b:
        if a > b:
```

```
            a = a - b
        else:
            b = b - a
    return a
```

14.2 Rozszerzony algorytm Euklidesa

Oprócz obliczenia największego wspólnego dzielnika, algorytm znajduje także współczynniki Bézouta.
Poprzedni algorytm obliczał ciąg r_n będący resztami z dzielenia, tzn.:

$$r_0 = a, r_1 = b \quad r_{i+1} = r_{i-1} - q_i r_i$$

gdzie q_i oznacza wynik całkowitego dzielenia r_{i-1} przez r_i.
Wprowadźmy dwa dodatkowe ciągi: x_i oraz y_i:

$$x_0 = 1, x_1 = 0 \quad x_{i+1} = x_{i-1} - q_i x_i$$

$$y_0 = 0, y_1 = 1 \quad y_{i+1} = y_{i-1} - q_i y_i$$

Możemy zauważyć, że:

$$r_i = x_i * a + y_i * b$$

```
def euclid(a, b):
    x, x' = 0, 1
    y, y' = 1, 0
    r, r' = b, a
    while r != 0:
        q = r' // r
        r', r = r, (r' - q * r)
        x', x = x, (x' - q * x)
        y', y = y, (y' - q * t)
    return r', x', y'
```

Ponadto ten algorytm pozawala obliczyć takie x', y', że: $ax' + by' = 0$ (wystarczy zwrócić x i y, zamiast x', y').

15 Ortogonalność wektorów w przestrzeni R_n; związki z liniową niezależnością. Metoda ortonormalizacji Grama-Schmidta

Definicja 15.1 ***Ortogonalność.***
Dla wektorów x, y w przestrzeni R_n z zadanym iloczynem skalarnym $\cdot$

$$x \perp y \Leftrightarrow x \cdot y = 0$$

Definicja 15.2 ***Liniowa niezależność***
Wektory $x_1, x_2, \ldots, x_n$ w przestrzeni R_n nazywamy liniowo niezależnymi jeśli

$$a_1 * x_1 + a_2 * x_2 + \ldots + a_n * x_n = 0 \Rightarrow a_1 = a_2 = \ldots = a_n = 0$$

tzn., nie istnieją taki układ skalarów $a_1, a_2 \ldots a_n$ dla których suma $\sum_{i=1}^{n} a_i * x_i = 0$ poza i przynajmniej jeden skalar nie równa się zeru.

Zbiór wektorów, który nie jest liniowo niezależny, jest zbiorem liniowo zależnym. Oznacza to, że przynajmniej jeden jego element jest kombinacją liniową pozostałych wektorów.

W przestrzeni R_n zbiór liniowo niezależnych nie może zawierać więcej niż n wektorów.

Twierdzenie 15.3 *Każdy układ ortogonalny wektorów w przestrzeni R_n jest liniowo niezależny.*

Definicja 15.4 ***Baza przestrzeni*** R_n
Zbiór wektorów $B \subset R_n$ *nazywamy bazą przestrzeni* R_n *jeśli:*

1. *Jest liniowo niezależny*
2. *Generuje przestrzeń* R_n *tzn. każdy wektor* $\in R_n$ *można zapisać jako kombinację liniową wektorów z* B

Twierdzenie 15.5 *Każdy ortogonalny układ* n *wektorów w* R_n *jest bazą.*

Definicja 15.6 ***Ortonormalność***
Zbiór wektorów A *przestrzeni* R_n *z zadanym iloczynem skalarnym* $\cdot$ *jest ortonormalny, jeśli*

$$\forall x, y \in A \quad x \cdot y = \begin{cases} 0, & x \neq y \\ 1, & x = y \end{cases}$$

Zatem wektory należące do A *są wersorami.*

15.1 Ortonormalizacja Grama-Schmidta

Ortonormalizacja to proces przekształcenia układu niezależnych liniowo wektorów w układ wektorów ortonormalnych.

Projekcję wektora $\mathbf{v}$ na wektor $\mathbf{u}$ definiujemy jako:

$$\text{proj}_{\mathbf{u}}\mathbf{v} = \frac{\mathbf{v} \cdot \mathbf{u}}{\mathbf{u} \cdot \mathbf{u}}\mathbf{u}$$

Algorytm:

1. Dla zbioru wektorów $A = \{\mathbf{v}_1, \ldots, \mathbf{v}_n\}$ oblicz $\{\mathbf{u}_1, \ldots, \mathbf{u}_n\}$ gdzie

$$\mathbf{u}_i = \mathbf{v}_i - \sum_{j=1}^{i-1} \text{proj}_{\mathbf{u_j}}\mathbf{v}_i$$

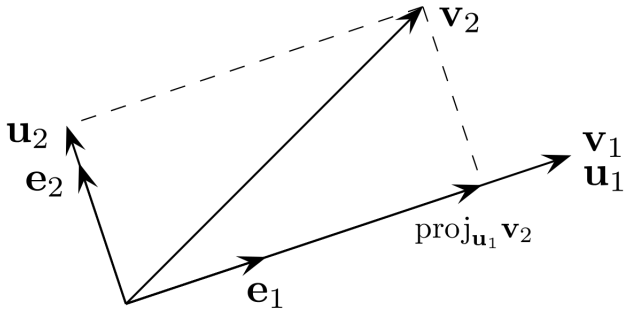
   tzn:

$$\mathbf{u}_1 = \mathbf{v}_1$$

$$\mathbf{u}_2 = \mathbf{v}_2 - \text{proj}_{\mathbf{u_1}}\mathbf{v}_2$$

$$\mathbf{u}_3 = \mathbf{v}_3 - \text{proj}_{\mathbf{u_1}}\mathbf{v}_3 - \text{proj}_{\mathbf{u_2}}\mathbf{v}_3$$

2. Wektory $\{\mathbf{u}_1, \ldots, \mathbf{u}_n\}$ podziel przez ich normy, uzyskując $\{\mathbf{e}_1, \ldots, \mathbf{e}_n\}$



16 Liczby Stirlinga I i II rodzaju i ich interpretacja.

Definicja 16.1 ***Liczby Stirling I rodzaju***.
Dla dowolnego $n \geqslant 1$ mamy:

1. $c(0,0) = 1$
2. $c(n,0) = c(0,n) = 0$
3. $c(n,k) = c(n-1,k-1) + (n-1) * c(n-1,k)$

W szczególności:

1. $c(n,n) = 1, c(n,1) = (n-1)!$
2. $\sum_{k=0}^{n} c(n,k) = n!$

Interpretacja: *Przez $c(n,k)$ ($\left[{n \atop k}\right]$) oznaczamy liczbę permutacji zbioru n-elementowego, które mają rozkład na dokładnie k cykli rozłącznych.*

Definicja 16.2 ***Liczby Stirlinga II rodzaju***.
Dla dowolnego $n \geqslant 1$ mamy:

1. $S(0,0) = 1$,
2. $S(n,0) = S(0,n) = 0$,
3. $S(n,k) = S(n-1,k-1) + k \times S(n-1,k)$.

w szczególności $S(n,n) = S(n,1) = 1$.

Interpretacja: *Przez $S(n,k)$ ($\left\{{n \atop k}\right\}$) oznaczamy liczbę rozmieszczeń n rozróżnialnych kul na k nierozróżnialnych stosach w taki sposób, aby żaden stos nie był pusty.*

17 Twierdzenia Eulera i Fermata; funkcja Eulera.

Definicja 17.1 ***Funkcja Eulera (Tocjent)***
Funkcja przypisująca każdej liczbie naturalnej liczbę liczb względnie pierwszych z nią i nie większych od niej.
$\sum_{m|n} \varphi(m) = n$

Własności

1. $\varphi(n) \leqslant n-1$ *dla każdego* $n > 1$
2. $\varphi(p) \leqslant p-1$ *dla każdego* p *będącego liczbą pierwszą*
3. $\varphi(mn) = \varphi(m)\varphi(n)$ *jeśli NWD(m,n) = 1 (m i n względnie pierwsze)*
4. $\varphi(p^k) = p^{k-1} \cdot (p-1)$ *jeśli* p *jest liczbą pierwszą*
5. $\varphi(n) = n\left(1-\frac{1}{p_1}\right)\left(1-\frac{1}{p_2}\right)\cdots\left(1-\frac{1}{p_n}\right)$ *gdzie* $p_1, p_2, ..., p_n$ *są czynnikami pierwszymi liczby* n
6. *jeżeli* $n = \prod_{i=1}^{k} p_i^{k_i}$ *jest rozkładem liczby n na czynniki pierwsze* $\varphi(n) = \prod_{i=1}^{k} \varphi\left(p_i^{k_i}\right)$

Definicja 17.2 ***Małe Twierdzenie Fermata***
Jeżeli p jest liczbą pierwszą, to dla dowolnej liczby całkowitej a, liczba $a^p - a$ *jest podzielna przez p.*

1. $a^{p-a} \equiv 0 \pmod{p}$

Jeśli p jest liczbą pierwszą, a a jest taką liczbą całkowitą, że liczby a i p są względnie pierwsze to $a^{p-1} - 1$ *dzieli się przez p.*

1. $a^{p-1} - 1 \equiv \pmod{p}$
2. $a^{p-1} \equiv 1 \pmod{p}$

Definicja 17.3 ***Twierdzenie Eulera***
Jeżeli $m \in \mathbb{Z}_+$ oraz $a \in \mathbb{Z}$ są liczbami względnie pierwszymi to m dzieli liczbę $a^{\varphi(m)} - 1$, gdzie $\varphi(m)$ oznacza wartość funkcji Eulera.

1. $a^{\varphi(m)} \equiv 1 \pmod{m}$

18 Konfiguracje i t-konfiguracje kombinatoryczne.

Definicja 18.1 ***Konfiuracją kombinatoryczną*** *B o parametrach (n, k, r) nazywamy rodzinę k-elementowych podzbiorów w X, jeżeli każdy element $x \in X$ występuje w dokładnie r podbiorach. Zachodzą warunki:*

1. $n = |X|$
2. $|B_i| = k \quad \forall i = 1, \ldots, b$
3. $n \cdot r = b \cdot k$ *gdzie b to liczba k-elementowych podzbiorów zbioru n-elementowego*

Twierdzenie 18.2 *Istnieje konfiguracja kombinatoryczna B o parametrach (n, k, r) wtedy i tylko wtedy, gdy:*

1. $k | n \cdot r$
2. $\frac{n \cdot r}{k} \leqslant \binom{n}{k}$

Definicja 18.3 *Niech X dowolny zbiór, $|X| = n$. B rodzina $k-$ podzbiorów X jest* ***t-konfiguracją kombinatoryczną*** *o parametrach (n, k, r_t) wtw gdy dla każdego $t-$podzbioru $T \subset X$ liczba bloków z B zawierających T jest równa r_t.*

Twierdzenie 18.4 *Jeśli B jest t-konfiguracją kombinatoryczną, to B jest s-konfiguracją kombinatoryczną dla $s = 1, 2, \dots\ t-1$. Uwagi:*

1. $r_{t-1} = r_t \cdot \frac{n-t+1}{k-t+1}$
2. *Jeżeli B jest $t-$konfiguracją o parametrach (n, k, r_t), to dla $1 \leqslant s \leqslant (t-1)$*

$$r_s = r_t \cdot \frac{(n-s)(n-s-1)\dots(n-t+1)}{(k-s)(k-s-1)\dots(k-t+1)}$$

3. *Jeśli B jest t-konfiguracją o parametrach (n, k, r_t), to*

$$(k-s)(k-s-1)\dots(k-t+1) | r_t(n-s)(n-s-1)\dots(n-t+1)$$
$$dla\ 0 \leqslant s \leqslant (t-1)$$

warunek 3. dla $t \leqslant 2$ jest konieczny dla istnienia t-konfiguracji, ale nie wystarczający

19 Cykl Hamiltona, obwód Eulera, liczba chromatyczna - definicje i twierdzenia.

19.1 Cykl Hamiltona

Definicja 19.1 ***Ścieżka Hamiltona***. *Ścieżką Hamiltona w grafie G nazywamy ścieżkę, która przechodzi przez wszystkie wierzchołki G.*

Definicja 19.2 ***Cykl Hamiltona***. *Powiemy, że graf G ma cykl Hamiltona, jeśli istnieje w nim cykl przechodzący przez wszystkie wierzchołki. Taki graf nazwyamy hamiltonowskim.*

Twierdzenie 19.3 *Niech $G = (V, E)$ będzie grafem o $n \geqslant 3$ wierzchołkach. Jeśli dla dwolonych różnych, niesąsiednich wierzchołków $u, v \in V$ zachodzi warunek $d(u) + d(v) \geqslant n$, to G jest hamiltonowski.*

Twierdzenie 19.4 *Jeśli G jest grafem o $n \geqslant 3$ wierzchołkach w którym minimalny stopień wierzchołka wynosi co najmniej $\frac{n}{2}$, to G jest hamiltonowski.*

Twierdzenie 19.5 *Niech $G = (V, E)$ będzie grafem o $n \geqslant 2$ wierzchołkach i takim, że $d(u) + d(v) \geqslant n - 1$ dla dwóch dowolnych różnych, niesąsiednich wierzchołków $u, v \in V$. Wtedy G ma ścieżkę hamiltona.*

19.2 Obwód Eulera

Definicja 19.6 ***Droga Eulera***. *Drogą Eulera w grafie G nazywamy drogę $v_1, v_2, \ldots, v_m$, w której każda krawędź grafu G użyta jest dokładnie raz.*

Definicja 19.7 ***Obwód Eulera***. *Jeśli w grafie G istnieje droga Eulera, w której pierwszy i ostatni wierzchołek są identyczne, to nazywamy ją obwodem Eulera. Graf ten nazywamy wtedy eulerowskim.*

Twierdzenie 19.8 *Niech G będzie grafem spójnym. Wówczas następujące warunki są równoważne:*

1. *G jest grafem eurelowskim,*
2. *stopień każdego wierzchołka w G jest parzysty.*

Twierdzenie 19.9 *Niech G będzie grafem spójnym. Wówczas G ma drogę Eulera $\Leftrightarrow$ w G są dokładnie zero lub dwa wierzchołki stopnia nieparzystego.*

19.3 Liczba chromatyczna

Definicja 19.10 ***Kolorowanie wierchołkowe***. *Kolorowaniem wierzchołkowym grafu $G = (V, E)$ przy użyciu (co najwyżej) k kolorów nazywamy funkcję $c : V \to \{1, \ldots, k\}$ spełniającą warunek $c(u) \neq c(v) \quad \forall u, v \in V : uv \in E$.*

Definicja 19.11 ***Liczba chromatyczna***. *Liczbą chromatyczną grafu G nazywamy najmniejszą liczbę $k \in \mathbb{N}$, dla której istnieje kolorwanie wierzchołkowe G przy użyciu k kolorów. Oznaczamy $\chi(G)$.*

1. $\chi(G) = 1 \Leftrightarrow |E(G)| = 0$
2. $\chi(T) = 2$ *dla każdego drzewa T o przynajmniej dwóch wierzchołkach*
3. $\chi(C_{2k}) = 2, \chi(C_{2k+1}) = 3$
4. $\chi(K_n) = n$

Twierdzenie 19.12 *Niech G będzie dowolnym grafem. Wtedy $\chi(G) \leqslant d_{max}(G) + 1$.*

Twierdzenie 19.13 *Niech G będzie grafem spójnym. Wówczas $\chi(G) \leqslant d_{max}(G)$, o ile G nie jest grafem pełnym ani cyklem o nieparzystej liczbie wierzchołków.*

20 Algorytm Forda-Fulkersona wyznaczania maksymalnego przepływu.

Definicja 20.1 ***Siecią przepływową*** *nazywamy zestaw* $(V, \vec{E}, s, t, c)$*, gdzie* $(V, \vec{E})$ *jest grafem skierowanym wraz z wyróżnionymi wierzchołkami* s *-* ***źródłem*** *i* t *-* ***ujściem*** *oraz* ***funkcją pojemności krawędzi*** $c : \vec{E} \to \mathbb{R}$*.*

Definicja 20.2 ***Przepływem*** *w sieci* $(V, \vec{E}, s, t, c)$ *nazywamy funkcję* $f : \vec{E} \to \mathbb{R}$ *spełniającą następujące warunki:*

1. $f(\vec{uv}) \leqslant c(\vec{uv})$ *dla każdej krawędzi* $\vec{uv} \in \vec{E}$*,*
2. *dla każdego ustalonego* $v \in V \ \{s, t\}$ *:*

$$\sum_{\vec{uv} \in \vec{E}} f(\vec{uv}) \quad = \quad \sum_{\vec{vw} \in \vec{E}} f(\vec{vw}).$$

Liczbę $|f| = \sum_{\vec{su} \in \vec{E}} f(\vec{su}) = \sum_{\vec{wt} \in \vec{E}} f(\vec{wt})$ *nazywamy* ***wartością przepływu*** *f.*

Przepływ f nazywamy ***maksymalnym****, jeśli jego wartość jest największa spośród wszystkich przepływów w danej sieci.*

Definicja 20.3 *Niech* $(V, \vec{E}, s, t, c)$ *będzie siecią przepływową oraz* f *pewnym przepływem tej sieci.* ***Siecią rezydualną*** *nazywamy sieć* $G_f = (V, E_f, s, t, c_f)$*, gdzie* $E_f = \vec{E} \cup \{\vec{vu} : \vec{uv} \in \vec{E}\}$ *z funkcją pojemności określoną wzorem:*

$$c_f(\vec{uv}) = c(\vec{uv}) - f(\vec{uv}) \quad dla\ \vec{uv} \in \vec{E}\ t.\ że\ c(\vec{uv}) > 0$$

$$c_f(\vec{uv}) = f(\vec{uv}) \quad dla\ \vec{uv} \notin \vec{E}\ t.\ że\ \vec{vu} \in \vec{E} if(\vec{uv}) > 0$$

oraz $c_f(\vec{uv}) = 0$ *w pozostałych przypadkach.*

Definicja 20.4 ***Ścieżką rozszerzającą*** *dla przepływu f nazywamy dowolną ścieżkę (skierowaną) $v_0 \to v_1 \to \cdots \to v_k$ w sieci rezydualnej G_f łączącą wierzchołki $v_0 = s$ i $v_k = t$, w której $c_f(v_i\vec{v_{i+1}}) > 0$ dla każdego $i = 0, \cdots, k-1$.*

Twierdzenie 20.5 ***Algorytm Forda-Fulkersona****. W celu znalezienia maksymalnego przepływu f w sieci przepływowej $(V, \vec{E}, s, t, c)$:*

1. *przyjmujemy początkowo dowolny (np. zerowy) przepływ f,*
2. *budujemy sieć rezydualną G_f,*
3. *dopóki w G_f istnieje ścieżka rozszerzająca $P = v_0 \to v_1 \to \cdots \to v_k$ określamy $c_p = min_{i=0,\cdots,k-1} c_f(v_i\vec{v_{i+1}})$, a następni zwiększamy o c_p wartość przepływu f na krawędziach $v_i\vec{v_{i+1}}$ tej ścieżki i uaktualniamy wartośi funkcji c_f sieci rezydualnej.*

Definicja 20.6 ***Przekrojem*** *w sieci przepływowej $(V, \vec{E}, s, t, c)$ nazywamy parę zbiorów (S, T) spełniającą warunki:*

1. *$S \cup T = V$,*
2. *$S \cap T = \emptyset$,*
3. *$s \in S, t \in T$.*

Pojemnością *przekroju (S, T) nazywamy liczbę*

$$c(S, T) = \sum_{(u,v) \in \vec{E} \cap (S \times T)} c(\vec{uv}).$$

Twierdzenie 20.7 *Niech $(V, \vec{E}, s, t, c)$ będzie siecią przepływową. Wówczas* ***maksymalna wartość przepływu jest jest równa minimalnej pojemności*** *przekroju w tej sieci.*

21 Rozwiązywanie równań rekurencyjnych przy użyciu funkcji tworzących (generujących) oraz przy użyciu równania charakterystycznego.

Definicja 21.1 ***Liniowym równaniem rekurencyjnym rzędu r*** *(o stałych współczynnikach) nazywamy równanie postaci:*

$$x_{n+r} = c_1 x_{n+r-1} + \cdots + c_r x_n + f(n)$$

w którym $c_i \in \mathbb{R}, c_r \neq 0$ oraz $f : \mathbb{N} \to \mathbb{R}$ jest dowolną funkcją. Jeśli funkcja f jest stale równa zero, to powyższe równanie nazywamy ***jednorodnym****.*

Definicja 21.2 ***Rozwiązaniem szczególnym*** *równania nazywamy dowolny ciąg (x_n) spełniający zależność rekurencyjną. Dla każdego równania jednorodnego rozwiązanie szczególnym jest ciąg stały równy zero.*
Rozwiązaniem ogólnym *równania nazywamy wzór ogólny (zależny od pewnych parametrów) pozwalający wyznaczyć wszystkie rozwiązania szczególne.*

21.1 Funkcje tworzące.

Definicja 21.3 ***Funkcją tworzącą*** *(generujacą) ciągu (a_n) nazywamy szereg*

$$\sum_{n=0}^{\infty} a_n x^n.$$

Jest to tylko inny zapis ciągu (wyraz a_n to współczynnik przy x^n) - nie interesują nas kwestie zbieżności, podstawiania wartości za x, ciągłości/różniczkowalności funkcji określonej takim szeregiem itd.
Jeżeli jednak funkcja tworząca pewnego ciągu "wygląda" jak zbieżny szereg określający funkcję znaną z analizy to możemy używać skróconego zapisu, np. $\sum_{n=0}^{\infty} \frac{1}{n!} x^n = e^x$.

Twierdzenie 21.4 ***Operacje na funkcjach tworzących.***

1. *Funkcje tworzące można* ***dodawać,odejmować i mnożyć przez liczbę*** *- odpowiadają temu operacje dodawania, odejmowania i mnożenia przez liczbę wyrazów ciągu.*

2. ***Mnożeniu dwóch funkcji tworzących*** *odpowiada operacja "splotu", tzn. jeśli* $A(x) = \sum_{n=0}^{\infty} a_n x^n$ *i* $B(x) = \sum_{n=0}^{\infty} b_n x^n$*, to* $A(x) * B(x) = \sum_{n=0}^{\infty} c_n x^n$*, gdzie* $c_n = \sum_{k=0}^{n} a_k b_{n-k}$*.*

3. *Jeśli funkcja tworząca A(x) ma "element odwrotny", tzn. taką funkcję tworzącą B(x) dla której* $A(x) * B(x) = 1$*, to mówimy że A(x) jest* ***odwracalna*** *i piszemy* $\frac{1}{A(x)} = B(x)$*.*

4. ***Mnożeniu funkcji*** *tworzącej* ***przez x*** *odpowiada przesunięcie ciągu (dopisanie na początek wyrazu zerowego). Operajcją odwrotną jest odjęcie wyrazu zerowego i podzielenie przez x.*

5. *Funkcje tworzące można* ***różniczkować*** *według wzoru:*

$$(\sum_{n=0}^{\infty} a_n x^n)' = \sum_{n=1}^{\infty} n a_n x^{n-1} = \sum_{n=0}^{\infty} (n+1) a_{n+1} x^n$$

6. *Zachodzą wzory na* ***pochodne*** *znane z analizy, np.* $(A(x) * B(x))' = A'(x)B(x) + A(x)B'(x)$*.*

7. *Operacja* ***"całkowania"*** *(odwrotna do różniczkowania):*

$$\int (\sum_{n=0}^{\infty} a_n x^n) = \sum_{n=0}^{\infty} \frac{a_n}{n+1} x^{n+1} = \sum_{n=0}^{\infty} \frac{a_n - 1}{n} x^n$$

Wyraz a_n *można "odzyskać" z funkcji tworzącej A(x) wykonując jej n-krotne różniczkowanie.*

Twierdzenie 21.5 *Dla dowolnej liczby naturalnej $m \geqslant 1$ zachodzi wzór:*

$$\frac{1}{(1-x)^m} = \sum_{n=0}^{\infty} \binom{n+m-1}{n} x^n.$$

Dla dowolneego $\alpha \in \mathbb{R}$ oraz $|x| < 1$ zachodzi wzór:

$$(1+x)^\alpha - \sum_{n=0}^{\infty} \binom{\alpha}{n} x^n$$

21.2 Równanie charakterystyczne.

Definicja 21.6 ***Równanie charakterystyczne*** *równania rekurencyjnego. Weźmy równanie rekurencyjne jednorodne*

$$a_n = Aa_{n-1} + Ba_{n-2}$$

gdzie dane są współczynniki A, B. Załóżmy, że ma ono rozwiązanie postaci $a_n = t^n$*. Podstawiając otrzymujemy:*

$$t^n = At^{n-1} + Bt^{n-2}.$$

Dzielimy obie strony przez t^{n-2}*:*

$$t^2 = At^1 + B$$

$$t^2 - At^1 - B = 0.$$

Równanie to nazywamy równaniem charakterystycznym równania rekurencyjnego. W tym przypadku jest to równanie kwadratowe.
Jeżeli nie ma ono pierwiastków podwójnych, wówczas:

$$a_n = Cr_1^n + Dr_2^n.$$

Jeżeli ma pierwiastki podwójne, to:

$$a_n = (C + Dn)r_1^n.$$

C i D są dowolnymi stałymi a r_1 *i* r_2 *są pierwiastkami równania charakterystycznego. Jeżeli dane jest* a_1 *i* a_2 *wówczas można łatwo ułożyć układ równań i otrzymać ich wartość.*

22 Ciąg i granica ciągu liczbowego, granica funkcji.

22.1 Ciągi.

Definicja 22.1 ***Ciąg liczbowy***. *Ciągiem liczbowym nazywamy funkcję* $\mathbb{N} \to \mathbb{R}$. *Wartość tej funkcji dla liczby naturalnej* n *nazywamy* n*-tym wyrazem ciągu i oznaczamy przez* a_n, b_n *itp. Ciągi o takich wyrazachoznaczamy odpowiednio przez* (a_n), (b_n) *itp. Zbiór wyrazów ciągu* (a_n), *tj.* $\{a_n : n \in \mathbb{N}\}$ *oznaczamy krótko przez* $\{a_n\}$.

Definicja 22.2 ***Granica właściwa ciągu***. *Ciąg* (a_n) *jest zbieżny do granicy właściwej* $a \in \mathbb{R}$, *co zapisujemy:*

$$lim_{n\to\infty}\ a_n = a$$

wtedy i tylko wtedy, gdy

$$\forall \varepsilon > 0 \quad \exists n_0 \in \mathbb{N} \quad \forall n \in \mathbb{N} \quad [(n > n_0) \Rightarrow (|a_n - a| < \varepsilon)]$$

Definicja 22.3 ***Granice niewłaściwe ciągu***.
Ciąg (a_n) *jest zbieżny do granicy niewłaściwej* ∞, *co zapisujemy:*

$$lim_{n\to\infty}\ a_n = \infty$$

wtedy i tylko wtedy, gdy:

$$\forall \varepsilon > 0 \quad \exists n_0 \in \mathbb{N} \quad \forall n \in \mathbb{N} \quad [(n > n_0) \Rightarrow (a_n > \varepsilon)]$$

Ciąg (a_n) *jest zbieżny do granicy niewłaściwej* $-\infty$, *co zapisujemy:*

$$lim_{n\to\infty} a_n = -\infty$$

wtedy i tylko wtedy, gdy:

$$\forall \varepsilon < 0 \quad \exists n_0 \in \mathbb{N} \quad \forall n \in \mathbb{N} \quad [(n > n_0) \Rightarrow (a_n < \varepsilon)]$$

Twierdzenie 22.4 ***O ograniczoności ciągu zbieżnego***. *Jeśli ciąg jest zbieżny do granicy właściwej, to jest ograniczony.*

Twierdzenie 22.5 ***O równoważności granic.***

$$lim_{n\to\infty}\ a_n = 0 \quad \Leftrightarrow \quad lim_{n\to\infty}\ |a_n| = 0.$$

Twierdzenie 22.6 ***O dwóch ciągach***. *Jeśli ciągi (a_n), (b_n) spełniają warunki:*

1. $a_n \leqslant b_n \quad \forall n \geqslant n_0$
2. $lim_{n\to\infty} a_n = \infty$

to

$$lim_{n\to\infty}\ b_n = \infty.$$

Prawdziwe jest także analogiczne twierdzenie dla ciągów zbieżnych do granicy niewłaściwej $-\infty$.

Twierdzenie 22.7 ***O trzech ciągach***. *Jeśli ciągi (a_n), (b_n), (c_n) spełniają warunki:*

1. $a_n \leqslant b_n \leqslant c_n \quad \forall n \geqslant n_0$
2. $lim_{n\to\infty} a_n = lim_{n\to\infty} c_n = b$

to

$$lim_{n\to\infty}\ b_n = b.$$

Twierdzenie 22.8 ***O ciągu monotonicznym i ograniczonym***. *Jeżeli ciąg (a_n) jest niemalejący dla $n \geqslant n_0$ oraz ograniczony z góry, to jest zbieżny do granicy właściwej $sup\{a_n : n \geqslant n_0\}$.*
Prawdziwe jest także analogiczne twierdzenie dla ciągu nierosnącego i ograniczonego z dołu.

22.2 Funkcje.

Definicja 22.9 ***Heinego granicy właściwej funkcji w punkcie***. *Niech $x_0 \in \mathbb{R}$ oraz niech funkcja f będzie określona przynajmniej na sąsiedztwie $S(x_0)$. Liczba g jest granicą właściwą funkcji f w punkcie x_0, co zapisujemy*

$$lim_{x\to x_0}\ f(x) = g$$

wtedy i tylko wtedy, gdy

$$\forall_{(x_n):\ \{x_n\}\subset S(x_0)}\quad [(lim_{n\to\infty} x_n = x_0) \Rightarrow (lim_{n\to\infty} f(x_n) = g)].$$

Definicja 22.10 ***Cauchy'ego granicy właściwej funkcji w punkcie***. *Niech $x_0 \in \mathbb{R}$ oraz niech funkcja f będzie określona przynajmniej na sąsiedztwie $S(x_0)$. Liczba g jest granicą właściwą funkcji f w punkcie x_0, co zapisujemy*

$$lim_{x\to x_0}\ f(x) = g$$

wtedy i tylko wtedy, gdy

$$\forall \varepsilon > 0\ \ \exists \delta > 0\ \ \forall x \in S(x_0)\quad [(|x - x_0| < \delta) \Rightarrow (|f(x) - g| < \varepsilon)].$$

Funkcja f ma granicę niewłaściwą ∞ w punkcie x_0 wtedy i tylko wtedy, gdy

$$\forall \varepsilon > 0\ \ \exists \delta > 0\ \ \forall x \in S(x_0)\quad [(|x - x_0| < \delta) \Rightarrow (f(x) > \varepsilon)].$$

Twierdzenie 22.11 ***O nieistnieniu granicy funkcji w punkcie***. *Jeśli istnieją ciągi (x'_n), (x''_n) spełniające warunki:*

1. *$lim_{n\to\infty} x'_n = x_0$, przy czym $x'_n \neq x_0\ \ \forall n \in \mathbb{N}$ oraz $lim_{n\to\infty}\ f(x'_n) = g'$,*
2. *$lim_{n\to\infty} x''_n = x_0$, przy czym $x''_n \neq x_0\ \ \forall n \in \mathbb{N}$ oraz $lim_{n\to\infty}\ f(x''_n) = g''$,*
3. *$g' \neq g''$,*

to granica $lim_{x\to x_0}\ f(x)$ nie istnieje (właściwa ani niewłaściwa).

Twierdzenie 22.12 ***Warunek konieczny i wystarczajacy istnienia granicy***. *Funckja f ma w punkcie x_0 granicę właściwą (niewłaściwą) wtedy i tylko wtedy, gdy*

$$lim_{x \to x_0^-} f(x) = lim_{x \to x_0^+} f(x)$$

Wspólna wartość granic jednostronnych jest wtedy granicą funkcji.

Twierdzenie 22.13 ***O niesitnieniu granicy funkcji w nieskończonośći***. *Jeżeli istnieją ciągi (x'_n), (x''_n) spełniające warunki:*

1. $lim_{n \to \infty} x'_n = \infty$ *oraz* $lim_{n \to \infty} f(x'_n) = g'$,
2. $lim_{n \to \infty} x''_n = \infty$ *oraz* $lim_{n \to \infty} f(x''_n) = g''$,
3. $g' \neq g''$,

to nie istnieeje granica $lim_{x \to x_0} f(x)$ (właściwa ani niewłaściwa).

Twierdzenie 22.14 ***O dwóch funkcjach***. *Jeśli funkcje f i g spełniają warunki:*

1. $f(x) \leqslant g(x) \quad \forall x \in S(x_0)$,
2. $lim_{x \to x_0} f(x) = \infty$,

to

$$lim_{x \infty x_0} g(x) = \infty.$$

Twierdzenie 22.15 ***Reguła de L'Hostpiala****. Jeżeli funkcja f i g spełniają warunki:*

1. *$lim_{x\to x_0} f(x) = lim_{x\to x_0} g(x) = 0 \ \ (\infty)$, przy czym $g(x) \neq 0$ dla $x \in S(x_0)$*
2. *istnieje granica $lim_{x\to x_0} \frac{f'(x)}{g'(x)}$ (właściwa lub niewłaściwa),*

to

$$lim_{x\to x_0} \frac{f(x)}{g(x)} = lim_{x\to x_0} \frac{f'(x)}{g'(x)}$$

23 Ciągłość i pochodna funkcji. Definicja i podstawowe twierdzenia.

23.1 Ciągłość.

Definicja 23.1 ***Funkcja ciągła w punkcie****. Niech $x_0 \in \mathbb{R}$ oraz niech funkcja f będzie określona przynajmniej na otoczeniu $O(x_0)$. Funkcja f jest ciągła w ounkcie x_0 wtedy i tylko wtedy, gdy*

$$lim_{x \to x_0}\ f(x) = f(x_0).$$

Funkcja jest ciągła na zbiorze*, jeżeli jest ciągła w każdym punkcie tego zbioru.*

Twierdzenie 23.2 ***Warunek konieczny i wystarczający ciągłości funkcji****. Funckja jest ciągła w punkcie wtedy i tylko wtedy, gdy jest w tym punkcie ciągła lewostronnie i prawostronnie.*

Definicja 23.3 ***Nieciągłość funkcji***. *Niech $x_0 \in \mathbb{R}$ oraz niech funkcja f będzie określona przynajmniej na otoczeniu $O(x_0)$. Funkcja f jest nieciągła w punkcie x_0 wtedy i tylko wtedy, gdy nie istnieje granica $lim_{x\to x_0}\ f(x)$ albo gdy $lim_{x\to x_0}\ f(x) \neq f(x_0)$.*

Nieciągłość pierwszego rodzaju. *Jeżeli istnieją granice skończone $lim_{x\to x_0^-\ f(x)}$, $lim_{x\to x_0^+}\ f(x)$ oraz*

$$lim_{x\to x_0^-}\ f(x) \neq f(x_0) \quad lub \quad lim_{x\to x_0^+}\ f(x) \neq f(x_0).$$

Mówimy, że funkcja f ma w punkcie x_0 nieciągłość pierwszego rodzaju typu "skok", jeżeli spełnia warunek

$$lim_{x\to x_0^-}\ f(x)\ \neq\ lim_{x\to x_0^+}\ f(x).$$

Mówi, że funkcja f ma w punkcie x_0 nieciągłość pierwszego rodzaju typu "luka", jeżeli spełnia warunek

$$lim_{x\to x_0^-}\ f(x)\ =\ lim_{x\to x_0^+}\ f(x)\ \neq\ f(x_0).$$

Nieciągłość drugiego rodzaju. *Jeżeli co najmniej jedna z granic*

$$lim_{x\to x_0^-}\ f(x),\quad lim_{x\to x_0^+}\ f(x)$$

nie istnieje lub jest niewłaściwa.

Twierdzenie 23.4 ***Weierstrassa o ograniczoności funkcji ciągłej***. *Jeżeli funkcja jest ciągła na przedziale domkniętym i ograniczonym, to jest na nim ograniczona.*

Twierdzenie 23.5 ***Weierstrassa o osiąganiu kresów***. *Jeżeli funkcja f jest ciągła na przedziale domkniętym $[a, b]$,to*

$$\exists c \in [a,b]\ \ f(c) =\ inf_{x\in[a,b]}\ f(x)\ \ oraz\ \ \exists d \in [a,b]\ \ f(d) =\ sup_{x\in[a,b]}\ f(x)$$

Twierdzenie 23.6 ***Darboux o przyjmowaniu wartości pośrednich***. *Jeżeli funkcja f jest ciągła na przedziale $[a, b]$ oraz spełnia warunek $f(a) < f(b)$, to*

$$\forall w \in (f(a, f(b))) \ \exists c \in (a, b) \quad f(c) = w.$$

23.2 Pochodna.

Definicja 23.7 ***Iloraz różnicowy***. *Niech $x_0 \in \mathbb{R}$ oraz niech funkcja f będzie określona przynajmniej na otoczeniu $O(x_0, r)$, gdzie $r > 0$. Ilorazem różnicowym funkcji f w punkcie x_0 odpowiadającym przyrostowi Δx, gdzie $0 < |\Delta x| < r$, zmiennej niezależnej nazywamy liczbę*

$$\frac{\Delta f}{\Delta x} \stackrel{def}{=} \frac{f(x_0 + \Delta x) - f(x_0)}{\Delta x}$$

Definicja 23.8 ***Pochodna właściwa funkcji***. *Niech $x_0 \in \mathbb{R}$ oraz niech funkcja f będzie określona przynajmniej na otoczeniu $O(x_0)$. Pochodną właściwą funkcji f w punkcie x_0 nazywamy granicę właściwą*

$$f'(x_0) \stackrel{def}{=} lim_{x \to x_0} \frac{f(x) - f(x_0)}{x - x_0}$$

Inaczej mówiąc pochodna funkcji f jest granicą ilorazu różnicowego gdy $\Delta x \to \infty$. Mamy zatem

$$f'(x_0) \stackrel{def}{=} lim_{\Delta x \to \infty} \frac{f(x_0 + \Delta x) - f(x_0)}{\Delta x}$$

Twierdzenie 23.9 ***Warunek konieczny istnienia pochodnej właściwej funkcji***. *Jeżeli funkcja ma pochodną właściwą w punkcie, to jest ciągła w tym punkcie. Implikacja odwrotna nie jest prawdziwa.*

Definicja 23.10 ***Pochodne jednostronne właściwe funkcji****. Niech $x_0 \in \mathbb{R}$ oraz niech funkcja f będzie określona przynajmniej na otoczeniu $O(x_0^-)$. Pochodną lewostronną właściwą funkcji f w punkcie x_0 nazywamy granicę właściwą*

$$f'_-(x_0) \stackrel{def}{=} lim_{x \to x_0^-} \frac{f(x) - f(x_0)}{x - x_0}$$

Analogicznie definiujemy $f'_+(x_0)$.

Jeżeli funkcja ma w punkcie pochodną lewostronną (prawostronną) właściwą, to jest w nim ciągła lewostronnie (prawostronnie).

Definicja 23.11 ***Pochodna funkcji na zbiorze****. Funkcja ma pochodną właściwą na zbiorze wtedy i tylko wtedy, gdy ma pochodną właściwą w każdym punkcie tego zbioru.*

Definicja 23.12 ***Pochodna niewłaściwa funkcji****. Niech f będzie funkcją ciągłą w punkcie $x_0 \in \mathbb{R}$. Funkcja f ma w punkcie x_0 pochodną niewłaściwą wtedy i tylko wtedy, gdy*

$$lim_{x \to x_0} \frac{f(x) - f(x_0)}{x - x_0} = \infty \quad albo \quad lim_{x \to x_0} \frac{f(x) - f(x_0)}{x - x_0} = -\infty$$

Podobnie definiujemy pochodne niewłaściwe jednostronne.

Twierdzenie 23.13 ***Zastosowanie różniczki do obliczeń przybliżonych***. *Jeżeli funkcja f ma pochodną właściwą w punkcie x_0, to*

$$f(x_0 + \Delta x) \approx f(x_0) + f'(x_0)\Delta x$$

Przy czym błąd, jaki popełniamy zastępując przyrost funkcji

$$\Delta f = f(x_0\Delta x) - f(x_0)$$

jej różniczką $df = f'(x_0)\Delta x$, dąży szybciej do zera niż przyrost zmiennej niezależnej Δx, tzn.

$$lim_{\Delta x \to 0} \frac{\Delta f - df}{\Delta x} = 0.$$

Twierdzenie 23.14 ***Rolle'a***. *Jeśli funkcja f spełnia warunki:*

1. *jest ciągła na $[a, b]$*
2. *ma pochodną właściwą lub niewłaściwą na (a, b),*
3. *$f(a) = f(b)$,*

to istnieje punkt $c \in (a, b)$ taki, że:

$$f'(c) = 0.$$

Twierdzenie 23.15 ***Lagrange'a***. *Jeżeli funkcja f spełnia warunki:*

1. *jest ciągła na $[a, b]$,*
2. *ma pochodną właściwą lub niewłaściwą na (a, b),*

to istnieje punkt $c \in (a, b)$ taki, że

$$f'(c) = \frac{f(b) - f(a)}{b - a}$$

24 Ekstrema funkcji jednej zmiennej. Definicje i twierdzenia.

Definicja 24.1 ***Minimum lokalne funkcji***. *Funkcja f ma w punkcie $x_0 \in \mathbb{R}$ minimum lokalne, jeżeli:*

$$\exists \delta > 0 \ \forall x \in S(x_0, \delta) \quad f(x) \geqslant f(x_0).$$

Analogicznie definiujemy ***maksimum lokalne***.

Minimum lokalne jest właściwe, *jeżeli:*

$$\exists \delta > 0 \ \forall x \in S(x_0, \delta) \quad f(x) > f(x_0).$$

Analogicznie definiujemy ***maksimum lokalne właściwe***.

Twierdzenie 24.2 ***Fermata, wranuek konieczny istnienia ekstremum***. *Jeżeli funkcja f ma:*

1. *esktremum lokalne w punkcie x_0,*
2. *pochodną $f'(x_0)$,*

to

$$f'(x_0) = 0.$$

Twierdzenie 24.3 ***I warunek wystarczający istnienia ekstremum***. *Jeżeli funkcja f spełnia warunki:*

1. $f'(x_0) = 0$,
2. $\exists \delta > 0 \begin{cases} f'x > 0 & \forall x \in S(x_0^-, \delta), \\ f'x < 0 & \forall x \in S(x_0^+, \delta), \end{cases}$

to w punkcie x_0 ma maksimum lokalne właściwe. Analogicznie dla minimum.

Twierdzenie 24.4 ***II warunek wystarczający istnienia ekstremum***. *Jeżeli funkcja f spełnia warunki:*

1. $f'(x_0) = f''(x_0) = \cdots = f^{(n-1)}(x_0) = 0$,
2. $f^{(n)}(x_0) < 0$,
3. *n jest liczbą parzystą, gdzie* $n \geqslant 2$,

to w punkcie x_0 ma maksimum lokalne właściwe. Analogicznie dla minimum,

25 Całka Riemanna funkcji jednej zmiennej.

Definicja 25.1 ***Całka oznaczona Riemanna***. *Niech funkcja f będzie ograniczona na przedziale $[a,b]$. Całkę oznaczoną Riemanna z funkcji f na przedziale $[a,b]$ definiujemy wzorem*

$$\int_a^b f(x)\,dx \quad \stackrel{def}{=} \quad lim_{\delta(P)\to 0}\sum_{k=1}^{n} f(x_k^*)\Delta x_k,$$

o ile po prawej stronie znaku równości granica jest właściwa oraz nie zależy od sposoby podziałów P przedziału $[a,b]$ ani od sposobów wyboru punktów pośrednich x_k^*, gdzie $1 \leqslant k \leqslant n$. Ponadto przyjmujemy

$$\int_a^a f(x)\,dx \stackrel{def}{=} 0 \quad oraz \quad \int_b^a f(x)\,dx \stackrel{def}{=} - \int_a^b f(x)\,dx \quad dla\ a<b$$

Funkcję, dla której istnieje całka Riemanna, nazywamy całkowalną.

Twierdzenie 25.2 ***Warunek wystarczający całkowalności funkcji***. *Jeżeli funkcja f jest ograniczona na przedziale $[a,b]$ i ma na tym przedziale skończoną liczbę punktów nieciągłości I rodzaju, to jest na nim całkowalna.*

Twierdzenie 25.3 ***Obliczanie całek przy pomocy sumy całkowej podziału równomiernego***. *Jeżeli funkcja f jest całkowalna na przedziale $[a,b]$, to*

$$\int_a^b f(x)\,dx \;=\; lim_{n\to\infty}[\frac{b-a}{n}\sum_{k=1}^{n} f(a+k\frac{b-a}{n})]$$

Twierdzenie 25.4 ***Newtona - Leibniza, główne twierdzenie rachunku całkowego***. *Jeżeli funkcja f jest ciągła na przedziale $[a,b]$, to*

$$\int_a^b f(x)\,dx \;=\; F(b)-F(a) \;=\; [F(x)]_a^b,$$

gdzie F oznacza dowolną funkcję pierwotną funkcji f na tym przedziale.

26 Pochodne cząstkowe funkcji wielu zmiennych; różniczkowalność i różniczka funkcji.

Definicja 26.1 ***Pochodne cząstkowe***. *Niech funkcja f będzie określona przynajmniej na otoczeniu punktu (x_0, y_0). Pochodną cząstkową pierwszego rzędu funkcji f względem x w punkcie (x_0, y_0) określamy wzorem:*

$$\frac{\partial f}{\partial x}(x_0, y_0) \stackrel{def}{=} lim_{\Delta x \to 0} \frac{f(x_0 + \Delta x, y_0) - f(x_0, y_0)}{\Delta x}$$

Pochodną tą oznacza się także symbolami: $f_x(x_0, y_0)$, $D_1 f(x_0, y_0)$.

Jeżeli granica określające pochodną cząstkową jest właściwa (niewłaściwa), to mówimy że pochodna ta jest właściwa (niewłaściwa). Jeżeli granica nie istnieje to to samo mówimy o pochodnej cząstkowej.

Definicja 26.2 ***Funkcja różniczkowalna w punkcie***. *Niech istnieją pochodne cząstkowe $\frac{\partial f}{\partial x}(x_0, y_0)$ $\frac{\partial f}{\partial y}(x_0, y_0)$. Funkcja f jest różniczkowalna w punkcie (x_0, y_0) wtedy i tylko wtedy, gdy spełniony jest warunek:*

$$lim_{(h,k)\to(0,0)} \frac{f(x_0 + h, y_0 + k) - f(x_0, y_0) - \frac{\partial f}{\partial x}(x_0, y_0)h - \frac{\partial f}{\partial y}(x_0, y_0)k}{\sqrt{h^2 + k^2}} = 0$$

Definicja 26.3 ***Różniczka funkcji***. *Niech funkcja f ma pochodne cząstkowe pierwszego rzędu w punkcie (x_0, y_0). Różniczką funkcji f w punkcie (x_0, y_0) nazywamy funkcję $df(x_0, y_0)$ zmiennych $\Delta x, \Delta y$ określoną wzorem:*

$$df(x_0, y_0)(\Delta x, \Delta y) \stackrel{def}{=} \frac{\partial f}{\partial x}(x_0, y_0)\Delta x + \frac{\partial f}{\partial y}(x_0, y_0)\Delta y$$

Twierdzenie 26.4 ***Zastosowanie różniczki funkcji do obliczeń przybliżonych***. *Niech funkcja f ma ciągłe pochodne cząstkowe pierwszego rzędu w punkcie (x_0, y_0). Wtedy*

$$f(x_0 + \Delta x, y_0 + \Delta y) \approx f(x_0, y_0) + df(x_0, y_0)(\Delta x, \Delta y)$$

Przy czym błąd $\delta(\Delta x, \Delta y)$ powyższego przybliżenia, tj. różnica $\Delta f - df$, dąży szybciej do 0 niż wyrażenie $\sqrt{(\Delta x)^2 + (\Delta y)^2}$. Oznacza to, że spełnia równość:

$$lim_{(\Delta x, \Delta y) \to (0,0)} \frac{\delta(\Delta x, \Delta y)}{\sqrt{(\Delta x)^2 + (\Delta y)^2}} = 0$$

$$f(x_0 + \Delta x, y_0 + \Delta y) \approx f(x_0, y_0) + \frac{\partial f}{\partial x}(x_0, y_0)\Delta x + \frac{\partial f}{\partial y}(x_0, y_0)\Delta y$$

27 Ekstrema funkcji wielu zmiennych. Definicje i twierdzenia.

Definicja 27.1 ***Minimum lokalne funkcji dwóch zmiennych.***

1. *Funkcja f ma w punkcie (x_0, y_0) minimum lokalne, jeżeli istnieje otoczenie tego punktu takie, że dla dowolnego (x, y) z tego otoczenia zachodzi nierówność*

$$f(x, y) \geqslant f(x_0, y_0)$$

Przy ostrej nierówności mówimy o minimum lokalnym ***właściwym****.*

Definicja 27.2 ***Maksimum lokalne funkcji dwóch zmiennych.***

1. *Funkcja f ma w punkcie (x_0, y_0) maksimum lokalne, jeżeli istnieje otoczenie tego punktu takie, że dla dowolnego (x, y) z tego otoczenia zachodzi nierówność*

$$f(x, y) \leqslant f(x_0, y_0)$$

Przy ostrej nierówności mówimy o maksimum lokalnym ***właściwym****.*

Twierdzenie 27.3 ***Warunek konieczny istnienia ekstremum****. Jeżeli funkcja f spełnia warunki:*

1. *ma ekstremum lokalne w punkcie (x_0, y_0)*
2. *istnieją pochodne cząstkowe $\frac{\partial f}{\partial x}(x_0, y_0)$, $\frac{\partial f}{\partial y}(x_0, y_0)$*

to

$$\frac{\partial f}{\partial x}(x_0, y_0) = 0, \quad \frac{\partial f}{\partial y}(x_0, y_0) = 0$$

Funkcja może mieć ekstrema tylko w punktach, w których wszystkie jej pochodne cząstkowe pierwszego rzędu się zerują albo w punktach, w których choć jedna z nich nie istnieje.

Twierdzenie 27.4 ***Warunek wystarczający istnienia ekstremum.*** *Niech funcka f ma ciągłe pochodne cząstkowe rzędu drugiego na otoczeniu punktu (x_0, y_0) oraz niech*

1. $\frac{\partial f}{\partial x}(x_0, y_0) = 0, \frac{\partial f}{\partial y}(x_0, y_0) = 0$
2. $det \begin{bmatrix} \frac{\partial^2 f}{\partial^2 x}(x_0, y_0) & \frac{\partial^2 f}{\partial x \partial y}(x_0, y_0) \\ \frac{\partial^2 f}{\partial x \partial y}(x_0, y_0) & \frac{\partial^2 f}{\partial^2 y}(x_0, y_0) \end{bmatrix} > 0$

Wtedy w punkcie (x_0, y_0) funkcja f ma ekstremum lokalne i jest to:

1. *minimum, gdy $\frac{\partial^2 f}{\partial^2 x}(x_0, y_0) > 0$,*
2. *maksimum, gdy $\frac{\partial^2 f}{\partial^2 x}(x_0, y_0) < 0$.*

28 Twierdzenie o zmianie zmiennych w rachunku całkowym; współrzędne walcowe i sferyczne.

Definicja 28.1 ***Twierdzenie o zmianie zmiennych w rachunku całkowym***. *Niech*

1. *odwzorowanie* $T : \begin{cases} x = \phi(u,v,w) \\ y = \psi(u,v,w) \\ z = \chi(u,v,w) \end{cases}$ *przekształca różnowartościowo wnętrze obszaru regularnego* Δ *na wnętrze obszaru regularnego* V,
2. *funkcje* ϕ, ψ, χ *mają ciągłe pochodne cząstkowe rzędu pierwszego na pewnym zbiorze otwartym zawierającym obszar* Δ,
3. *funkcja* f *jest ciągła na obszarze* V,
4. *jakobian* J_T *jest różny od zera wewnątrz obszaru* Ω.

Wtedy

$$\iiint_V f(x,y,z)\,dx\,dy\,dz = \iiint_\Omega f(\phi(u,v,w),\psi(u,v,w),\chi(u,v,w))$$

$$|J_T(u,v,w)|\,du\,dv\,dw$$

gdzie

$$J_T(u,v) \stackrel{def}{=} det \begin{bmatrix} \frac{\partial\phi}{\partial u}(u,v,w) & \frac{\partial\phi}{\partial v}(u,v,w) & \frac{\partial\phi}{\partial w}(u,v,w) \\ \frac{\partial\psi}{\partial u}(u,v,w) & \frac{\partial\psi}{\partial v}(u,v,w) & \frac{\partial\psi}{\partial w}(u,v,w) \\ \frac{\partial\chi}{\partial u}(u,v,w) & \frac{\partial\chi}{\partial v}(u,v,w) & \frac{\partial\chi}{\partial w}(u,v,w) \end{bmatrix}$$

Definicja 28.2 ***Współrzędne walcowe***. *Położenie punktu P w przestrzeni można opisać trójką liczb (φ, ϱ, h), gdzie:*

φ - oznacza miarę kąta między rzutem promienia wodzącego punktu P na płaszczyznę xOy, a dodatnią częścią osi Ox, $0 \leqslant \varphi \leqslant 2\pi$ albo $-\pi < \varphi \leqslant \pi$

ϱ - oznacza odległość rzutu punktu P na płaszczyznę xOy od początku układu współrzędnych, $0 \leqslant \varrho < \infty$

h - oznacza odległość (dodatnią dla $z > 0$ i ujemną dla $z < 0$) punktu P od płaszczyzny xOy, $-\infty < h < \infty$

Zależność między współrzędnymi walcowymi i kartezajńskimi.

$$W : \begin{cases} x = \varrho cos\varphi \\ y = \varrho sin\varphi \\ z = h \end{cases}$$

Współrzędne walcowe w całce potrójnej. *Niech:*

1. *obszar Ω we współrzędnych walcowych będzie obszarem normalnym*
2. *funkcja f będzie ciągła na obszarze U, które jest obrazem obszaru Ω przy przekształceniu walcowym; $U = W(\Omega)$.*

Wtedy

$$\iiint_f (x, y, z)\, dx\, dy\, dz = \iiint_\Omega f(\varrho cos\varphi, \varrho sin\varphi, h)\varrho\, dh\, d\varrho\, d\varphi$$

Definicja 28.3 ***Współrzędne sferyczne****. Położenie punktu P przestrzeni można opisać trójką liczb (φ, ψ, ϱ), gdzie*

φ - oznacza miarę kąta między rzutem promienia wodzącego punktu P na płaszczyznę xOy, a dodatnią częścią osi Ox, $0 \leqslant \phi \leqslant 2\pi$ albo $-\pi < \phi \leqslant \pi$

ψ - oznacza miarę kąta między promieniem wodzącym punktu P, a płaszczyzną xOy, $-\frac{\pi}{2} \leqslant \psi \leqslant \frac{\pi}{2}$

ϱ - oznacza odległość punktu P od początku układu współrzędnych, $0 \leqslant \varrho < \infty$

Zależność między współrzędnymi sferycznymi i kartezjańskimi*.*

$$S : \begin{cases} x = \varrho cos\varphi cos\psi \\ y = \varrho sin\varphi cos\psi \\ z = \varrho sin\psi \end{cases}$$

Współrzędne sferyczne w całce potrójnej*. Niech:*

1. *obszar Ω we współrzędnych sferycznych będzie obszarem normalnym*
2. *funkcja f będzie ciągła na obszarze U, które jest obrazem obszaru Ω przy przekształceniu walcowym; $U = S(\Omega)$.*

Wtedy

$$\iiint_U f(x,y,z)\,dx\,dy\,dz = \iiint_\Omega f(\varrho cos\varphi cos\psi, \varrho sin\varphi cos\psi, \varrho sin\psi)\varrho^3\,d\varrho\,d\psi\,d\varphi$$

29 Metody dowodzenia poprawności pętli.

- **Asercja** - warunek logiczny wyrażający zależności między zmiennymi algorytmu w kontekście ich aktualnego stanu.
- **Asercja początkowa** - asercja określająca warunek wejściowy (warunek danych wejściowych) algorytmu.
- **Asercja końcowa** - asercja określająca wyniki algorytmu (warunek danych wyjściowych).
- Asercje początkowa i końcowa są znane w momencie definiowania problemu i poszukiwania algorytmu, stanowiąc **specyfikację algorytmu**
- Algorytm A ma własność **określoności obliczeń** względem warunku α, jeśli dla każdych danych spełniających warunek α działanie algorytmu A nie zostanie zerwane (oznaczenie $obl(\alpha, A)$).
- Algorytm A ma **własność stopu** względem warunku α, jeśli dla każdych danych spełniających warunek α działanie algorytmu A nie ciągnie się w nieskończoność (oznaczenie stop (α, A)).
- Algorytm A jest **częściowo poprawny** względem warunku początkowego α oraz warunku końcowego β, gdy dla każdych danych spełniających warunek początkowy α, jeżeli działanie algorytmu A dochodzi do końca, wyniki spełniają warunek końcowy β (oznaczenie cp (α, A, β)).
  $cp(\alpha, z \leftarrow w, \beta) \iff (\alpha \Rightarrow \beta|_{z \leftarrow w})$
  Przy asercji początkowej α i asercji końcowej β podstawienie z $\leftarrow$ w jest częściowo poprawne wtedy i tylko wtedy, gdy poprawna jest implikacja od asercji α do asercji β, z zastąpieniem każdego wystąpienia zmiennej z podstawianym wyrażeniem w.
- Algorytm A jest **całkowicie poprawny** (semantycznie poprawny) względem warunku początkowego α oraz warunku końcowego β, jeśli ma własność określoności obliczeń względem warunku α, ma własność stopu względem warunku α oraz jest częściowo poprawny względem warunku α i warunku β (oznaczenie sp (α, A, β)).
- Formalnie: $sp(\alpha, A, \beta) \iff obl(\alpha, A) \wedge stop(\alpha, A) \wedge cp(\alpha, A, \beta)$

30 Odwrotna Notacja Polska: definicja, własności, zalety i wady, algorytmy.

31 Modele obliczen: maszyna Turinga.

32 Modele obliczen: automat skończony, automat ze stosem.

33 Złożoność obliczeniowa - definicja notacji: O, Ω, Θ.

Definicja 33.1 *Niech* $f, g, h : \mathbb{N} \to \mathbb{R}_+ \cup \{0\}$, *wtedy:*

- $\mathbf{f(n) = O(g(n))}$ - f *jest* ***co najwyżej rzędu*** g, *gdy istnieje* $c > 0$ *i* $n_0 \in \mathbb{N}$, *takie że* $f(n) \leqslant cg(n)$ *dla każdego* $n \geqslant n_0$.
- $\mathbf{f(n) = \Omega(g(n))}$ - f *jest* ***co najmniej rzędu*** g, *gdy* $g(n) = O(f(n))$
- $\mathbf{f(n) = \Theta(g(n))}$ - f *jest* ***dokładnie rzędu*** g, *gdy* $f(n) = O(g(n))$ *i* $f(n) = \Omega(g(n))$.

34 Złożoność obliczeniowa - pesymistyczna i średnia.

Definicja 34.1 *Niech:*

- D_n *- zbiór danych rozmiaru* n*,*
- $t(d)$ *- liczba operacji dominujących,*
- X_n *- zmienna losowa dla* $t(d) \in D_n$*,*
- p_{kn} *- rozkład prawdopodbieństwa zmiennej* X_n*.*

Optymistyczna złożoność czasowa*:*

$$Opt(n) = inf\{t(d) : d \in D_n\}$$

Średnia złożoność czasowa*:*

$$A(n) = ave(X_n) = \sum_{k \geqslant 0} kp_{nk}$$

Pesymistyczna złożoność czasowa*:*

$$W(n) = sup\{t(d) : d \in D_n\}$$

35 Metoda ”dziel i zwyciężaj”; zalety i wady.

36 Lista: ujęcie abstrakcyjne, możliwe implementacje i ich złożoności.

Definicja 36.1 *Lista jest abstrakcyjną strukturą danych (ADT), która posiada następujące właściwości*

- *Przechowuje elementy* ***tego samego typu***
- *Wykorzystuje pamięć w sposób* ***dynamiczny lub statyczny***
- *Dostęp do danych jest* ***sekwencyjny***
- *Elementy w liście sa* ***liniowo uporządkowane*** *zgodnie z ich pozycją na liście*
  - *element ai znajduje się na pozycji ‘i’*
  - *element ai poprzedza ai+1 dla i=1,2,...,n-1*
  - *element ai następuje po ai-1 dla i=2,...,n*
- *Wyróżnia się dwa podstawowe sposoby implementacji listy jako ADT*
  - *Tablica*
  - *Lista wiązana*

Definicja 36.2 *W sensie matematycznym lista jest skończonym ciągiem elementów ustalonego typu Node:* $a1, a2, ..., an, , n \geqslant 0$ *przy czym Node, zwany typem bazowym, może być np. typem* ***klasy, struktury, int, string*** *itp.*

36.1 Implementacja za pomocą tablic (array)

Tablica wykorzystuje pamięć w sposób **statyczny**. Oznacza to, że ma z góry określony rozmiar oraz pewna jej część może być niewykorzystana. Tablica posiada zmienne wskazujące na jej maksymalny rozmiar, ilość zajętych miejsc. Indeksowanie elementów zazwyczaj zaczyna się od 0 i kończy na maxsize-1.

36.2 Implementacja za pomocą Listy Wiązanej (Linked List)

Rozróżnia się dwa podstawowe rodzaje:

1. Lista wiązana **jednokierunkowa**
   Z każdego elementu możliwe jest przejście do jego następnika NEXT

2. Lista wiązana **dwukierunkowa**
   Z każdego elementu możliwe jest przejscie do jego następnika NEXT i poprzednika PREV

W liście wiązanej wyróżnia się dwie podstawowe zmienne typu Node potrzebne do jej obsługi

1. HEAD

2. TAIL

Dodatkowo, lista wiązana jest **cykliczna** jeżeli

1. $next(TAIL) \rightarrow HEAD$

2. $prev(HEAD) \rightarrow TAIL$

Pojedynczy element listy jest klasą lub strukturą, zwyczajowo nazywa się Node (węzeł). Przechowuje pole data o pożądanym typie danych oraz wskaźnik (wskaźniki) na element następny (i poprzedni).
Przykład implementacji struktury Node w C++

```
struct Node {
    int data;
    struct Node* next;
    struct Node* prev;
};
```

Przykład implementacji klasy Node w Java

```
class Node<E> {
    E data;
    Node<E> next;
    Node<E> prev;
};
```

Przykład implementacji klasy LinkedList w Java

```
class LinkedList<E> {
    Node<E> head; // head of the list
    Node<E> tail; // tail of the list
    int counter = 0;
    ...
}
```

Operacje i złożoności

1. `CREATE(l: List)`

   Tworzy nową listę
   O(1) - czas stały

2. `APPEND(l: List; d: Data)`

   Dodaje na końcu listy l rekord d
   O(1) - czas stały

3. `INSERT(l: List; d: Data; i: Integer)`

   Dodaje rekord d do określonego miejsce i w liście l
   O(n) - czas liniowy

4. `FIND(l: List; i: Integer)`

   Znajduje i-ty w kolejności rekord z listy
   O(min(i, n)) - czas liniowy

5. `DELETE(l: List; i: Integer)`

   Usuwa -ity w kolejności rekord z listy
   O(min(i, n)) - czas liniowy

6. `MAKENULL(l: List)`

   Czyści listę, usuwając wszystkie rekordy
   O(1) - czas stały

37 Kolejka i kolejka priorytetowa: ujęcie abstrakcyjne, możliwe implementacje i ich złożoności.

37.1 Kolejka

Definicja 37.1 *Kolejka FIFO (First In First Out).*
Jest to lista, w której

- *Wstawianie nowego elementu odbyca się na końcu kolejki* ***rear***
- *Usuwanie elementu odbywa się na początku kolejki* ***front***
- *Wyróżnia się dwa podstawowe sposoby implementacji kolejki*
  - *Tablica cykliczna*
  - *Lista wiązana*

Operacje i złożoności

1. `Create()`

   Tworzy pustą kolejkę

2. `isEmpty()`

   Sprawdza czy kolejka pusta

3. `Enqueue(x)`

   Wstawia nowy element x

4. `Front()`

   Odczytuje pierwszy element w kolejce

5. `Dequeue()`

   Usuwa pierwszy element w kolejce i go zwraca

37.2 Sposoby implementacji kolejki

Na początku zauważmy, że **nieefektywną** realizacją kolejki w tablicy jest przyjęcie, że początek kolejki **front** będzie zawsze równy 0, a **rear** będzie indeksem do pierwszego wolnego elementu tablicy.

Najbardziej **efektywnym** rozwiązaniem jest reprezentacja kolejki w **tablicy cyklicznej**. Operacje wstawiania i usuwania zwiększają pozycje końca i początku kolejki o 1 modulo rozmiar tablicy, wykorzystując metodę:

```
private int Add (int i){
    return (i+1) % maxSize; // reszta z dzielenia
}
```

37.3 Kolejka Priorytetowa

Definicja 37.2 *Kolejka Priorytetowa*
Jest to lista, w której

- *Usuwany jest zawsze element o największej (najmniejszej) wartości*

Operacje

1. `Insert(x, q)`

   Wstawienie elementu 'x' do kolejki q

2. `Max(Q)`

   Odczyt największego elementu w kolejce (element ten jest zwracany jako wynik operacji, kolejka się nie zmienia)

3. `Min(Q)`

   Odczyt największego elementu w kolejce (element ten jest zwracany jako wynik operacji, kolejka się nie zmienia)

4. `Delete_Max(Q)`

   Usunięcie największego elementu z kolejki q

5. `Delete_Min(Q)`

   Usunięcie największego elementu z kolejki q

37.4 Sposoby implementacji kolejki priorytetowej

- Lista - Tablica Nieuporządkowana NIEEFEKTYWNA!
- Lista - Tablica Uporządkowana Rosnąco (Malejąco) NIEEFEKTYWNA!
- Kopiec Max (Min)

38 Algorytmy sortowania QuickSort i MergeSort: metody wyboru pivota w QS; złożoności.

38.1 QuickSort.

```
def quickSort(arr, start, end):
if (start < end)
pivot = partition(arr, start, end)

quickSort(arr, start, pivot - 1)
quickSort(arr, pivot + 1, end)


def partition (arr, start, end):
pivot = arr[end]
i = (start - 1)

for j in range [start,end- 1]:
if (arr[j] < pivot):
i++;
swap arr[i] and arr[j]

swap arr[i + 1] and arr[end])
return (i + 1)
```

Złożoność: pesymistyczna - $O(n^2)$, średnia i optymistyczna - $O(nlog_2n)$.

Sposoby wybrania pivota

1. Pierwszy element
2. Ostatni element
3. Mediana z pierwszego, środkowego i ostatniego
4. Losowy element

38.1.1 MergeSort.

```
def mergeSort (arr, start, end):
if end > start:
middle = (start+end)/2

mergeSort (arr, start, middle)
mergeSort (arr, middle+1, end)

merge (arr, start, middle, right)
```

Złożoność: pesymistyczna, średnia, optymistyczna - $O(nlogn)$.

39 Algorytm sortowania bez porównań (sortowanie przez zliczanie, sortowanie kubełkowe oraz sortowanie pozycyjne).

39.1 CountSort.

```
def countSort (arr):
count = []
for a in arr:
count[a] += 1

i = 0;
for j in range [0, arr.len]:
while (count[i] == 0):
i++
arr[j] = i
count[i]--
```

Złożoność: $O(n+k)$.

39.2 BucketSort.

```
bucketSort (arr):
n = arr.len
buckets = [{} for i in range [1,n]]

for a in arr:
buckets[n*arr[i]].add(arr[i])

for b in buckets:
sort(b)

i = 0
for b in buckets:
for a in b:
arr[i++] = a
```

Złożoność: pesymistyczna - $O(n^2)$, średnia - $O(n+\frac{n^2}{k}+k)$.

39.3 RadixSort.

```
def radixSort (arr):
for all digits ascending:
countSort(arr, digit)
```

Złożoność: $O(d(n+k))$, gdzie d jest liczbą cyfr.

40 Reprezentacja drzewa binarnego za pomocą porządków (preorder, inorder, postorder).

```
def preorder (v):
func(v)
preorder(v.left)
preorder(v.right)

def inorder (v):
inorder(v.left)
func(v)
inorder(v.right)

def postorder (v):
postorder(v.left)
postorder(v.right)
func(v)
```

Możemy odtworzyć wyjściowe drzewo, jeśli mamy inorder i post- lub pre-order.

41 Algorytmy wyszukiwania następnika i poprzednika w drzewach BST; usuwanie węzła.

42 B-drzewa: operacje i ich złożoność.

43 Drzewa AVL: rotacje, operacje z wykorzystaniem rotacji i ich złożoność.

44 Algorytmy przeszukiwania wszerz i w głąb w grafach.

45 Algorytmy wyszukiwania najkrótszej ścieżki (Dijkstry oraz Bellmana-Forda).

46 Programowanie dynamiczne: podział na podproblemy, porównanie z metodą "dziel i zwyciężaj".

47 Algorytm zachłanny: przykład optymalnego i nieoptymalnego wykorzystania.

48 Kolorowania wierzchołkowe (grafów planarnych) i krawędziowe grafów, algorytmy i ich złożoności.

49 Algorytmy wyszukiwania minimalnego drzewa rozpinającego: Boruvki, Prima i Kruskala.

50 Najważniejsze algorytmy wyznaczania otoczki wypukłej zbioru punktów w układzie współrzędnych (Grahama, Jarvisa, algorytm przyrostowy (quickhull)).

51 Problemy P, NP, NP-zupełne i zależności między nimi. Hipoteza P vs. NP.

52 Automat minimalny, wybrany algorytm mini-

58 Reprezentacja liczb całkowitych; arytmetyka.

59 Reprezentacja liczb rzeczywistych; arytmetyka zmiennopozycyjna.

60 Różnice w wywołaniu funkcji statycznych, niestatycznych i wirtualnych w C++.

61 Sposoby przekazywania parametrów do funkcji (przez wartość, przez referencję). Zalety i wady.

62 Wskaźniki, arytmetyka wskaźników, różnica między wskaźnikiem a referencją w C++.

63 Podstawowe założenia paradygmatu obiektowego: dziedziczenie, abstrakcja, enkapsulacja, polimorfizm.

64 Funkcje zaprzyjaźnione i ich związek z przeładowaniem operatorów w C++.

65 Programowanie generyczne na podstawie szablonów w języku C++.

66 Podstawowe kontenery w STL z szerszym omówieniem jednego z nich.

67 Obsługa sytuacji wyjątkowych w C++.

68 Obsługa plików w języku C.

69 Model wodospadu a model spiralny wytwarzania oprogramowania.

70 Diagram sekwencji i diagram przypadków użycia w języku UML.

76 Relacyjny model danych, normalizacja relacji (w szczególności algorytm doprowadzenia relacji do postaci Boyce'a-Codda), przykłady.

77 Indeksowanie w bazach danych: drzewa B+, tablice o organizacji indeksowej, indeksy haszowe, mapy binarne.

78 Podstawowe cechy transakcji (ACID). Metody sterowania współbieżnością transakcji, poziomy izolacji transakcji, przykłady.

79 Złączenia, grupowanie, podzapytania w języku SQL.

80 Szeregowalność harmonogramów w bazach danych.

81 Definicja cyfrowego układu kombinacyjnego - przykłady układów kombinacyjnych i ich implementacje.

82 Definicja cyfrowego układu sekwencyjnego - przykłady układów sekwencyjnych i ich implementacje.

83 Minimalizacja funkcji logicznych.

84 Programowalne układy logiczne PLD (ROM, PAL, PLA).

85 Schemat blokowy komputera (maszyna von Neumanna).

86 Zarządzanie procesami: stany procesu, algorytmy szeregowania z wywłaszczaniem.