Technologie informacyjne

tydzień 2.

Michał Kasprowicz
mkasprowicz@atins.pl

Podstawy arkuszy kalkulacyjnych

Podstawy arkuszy kalkulacyjnych

- Wprowadzenie do pracy z danymi tabelarycznymi
- Analiza i wizualizacja informacji
- Narzędzie niezbędne w pracy akademickiej i zawodowej

- Definicja i charakterystyka
  - Tabela do przechowywania i przetwarzania danych
  - Organizacja danych w wierszach i kolumnach
  - Możliwość wykonywania operacji logicznych, obliczeń, analiz i tworzenia wykresów
  - Automatyzacja powtarzalnych operacji
-

Struktura arkusza kalkulacyjnego

- Struktura arkusza kalkulacyjnego
  - Komórki
  - Wiersze
  - Kolumny
  - Arkusze robocze
- Adresowanie komórek

Typy danych w arkuszu

- Liczby (integers, floats): 42, 3.14, -15.5
- Tekst (string): „Jan Kowalski“, „Hello world“
- Daty: 2026-02-21, 21/02/2026
- Wartości logiczne: TRUE, FALSE
- Formuły (wzory): =A1+B1

Formatowanie komórek

- Czcionka: rodzaj, rozmiar, styl (pogrubienie, kursywa)
- Wyrównanie: w poziomie, w pionie
- Szerokość kolumn, wysokość wierszy
- Obramowanie i wypełnienie kolorem
- Format liczb: waluta, procenty, liczby z wybraną ilością miejsc po przecinku.

Formuły i operatory arytmetyczne

- Formuła rozpoczyna się od znaku =
- Odwołania do komórek, zamiast stałych wartości
- Wykonywanie obliczeń matematycznych
- Dodawanie: =A1+B1
- Odejmowanie: =A1-B1
- Mnożenie: =A1*B1
- Dzielenie: =A1/B1
- Potęgowanie: =A1^2 (A1 do kwadratu)

Funkcje

- Sumowanie zakresów danych
  - Składnia: =SUM(zakres)
  - Przykłady:
    - =SUM(A1:A10)
    - =SUM(A1:B10)
    - =SUM(A1,A5,A9)
- Analiza zestawu danych
  - AVEREAGE(zakres) – średnia arytmetyczna
  - MIN(zakres) – wartość minimalna
  - MAX(zakres) – wartość maksymalna
  - COUNT(zakres) – liczba komórek z liczbami
  - COUNTA(zakres) – liczba niepustych komórek

Adresy względne vs. bezwzględne

- Typy odniesień do komórek:
  - Względne: A1 – zmienia się przy kopiowaniu
  - Bezwzględne: A1 – nie zmienia się przy kopiowaniu
  - Mieszane: $A1 (zablokowana kolumna) lub A$1 (zablokowany wiersz)
  - Klawisz F4 przełącza między typami
-

Logika warunkowa

- Podejmowanie decyzji w arkuszu:
  - Funkcja IF
  - Składnia: =IF(warunek, wartość_jeśli_prawda, wartość_jeśli_fałsz)
  - Operatory porównania: >, <, >=, <=, =, <>
  - Zagnieżdżenie IF dla wielu warunków: staje się nieczytelne, czasem lepiej jest użyć innych funkcji (VLOOKUP), oraz funkcji logicznych AND, OR.

Funkcje logiczne AND i OR

- AND(warunek1, warunek2, ...) - wszystkie muszą być prawdziwe
- OR(warunek1, warunek2, ...) - przynajmniej jeden prawdziwy
- Przykład użycia: =IF(AND(A1>50, B1<100), „OK“, „Błąd“)
- Sprawdzenie wielu kryteriów jednocześnia

Funkcje tekstowe

- CONCATENATE(tekst1, tekst2, ….) lub operator & - łączenie tekstów
- LEFT(tekst, liczba_znaków) – pobierz daną liczbę znaków w lewej
- RIGHT(tekst, liczba_znaków) – pobierz daną liczbę znaków z prawej
- LEN(tekst) – dugość tekstu (liczba znaków)
- UPPER/LOWER – zmiana na wielkie/małe litery

Funkcje daty i czasu

- TODAY() - zwraca dzisiejszą datę
- NOW() - zwraca aktualną datę i godzinę
- DATE(rok, miesiąc, dzień)
- YEAR(data), MONTH(data), DAY(data)
- Obliczenia: różnice dat, dodawanie dni

Sortowanie danych

- Sortowanie rosnące (A → Z, 0 → 9)
- Sortowanie malejące (Z → A, 9 → 0)
- Sortowanie po jednej lub wielu kolumnach
- Opcja „moje dane zawierają nagłówki“
- Przykład użycia: sortowanie listy publikacji po roku, potem alfabetycznie po autorze.

Filtrowanie danych

- Autofiltr – rozwijanie listy w nagłówkach
- Wybór konkretnych wartości do wyświetlenia
- Filtrowanie liczb: większe niż, mniejsze niż, zakres
- Filtrowanie tekstu: zawiera, zaczyna się od
- Filtrowanie dat: dzisiaj, ten tydzień, zakres niestandardowy
- Ctrl+Shift+L – włącza/wyłącza filtr

Tworzenie wykresów

- *„Jeden obraz wart tysiąc słów“*
- Wykres jako graficzna reprezentacja danych liczbowych
- Ułatwia znalezienie trendów i zależności
- Typy wykresów: kolumnowy, liniowy, kołowy, punktowy
- Proces: zaznacz dane → wstaw wykres → dostosuj

Typy wykresów i ich zastosowania

- Kolumnowy/słupkowy – porównywanie kategorii
- Liniowy – trendy w czasie, dane ciągłe
- Kołowy – proporcje części do całości
- Punktowy (rozproszenia) – zależności między dwoma zmiennymi
- Obszarowy – suma składowych w czasie

Elementy wykresu

- Tytuł wykresu – zwięzły opis co przedstawia
- Osie: X (pozioma) i Y (pionowa) z opisami
- Legenda – objaśnienie serii danych
- Siatka – ułatwia odczytywanie wartości
- Etykiety danych
- **Główna zasada**: wykres powinien być zrozumiały bez czytania towarzyszącego tekstu.

Praktyczne zastosowania arkuszy

- Prowadzenie dziennika (pomiary, obserwacje)
- Analiza danych eksperymentalnych (statystyki, wykresy)
- Budżetowanie projektu badawczego
- Planowanie harmonogramu pracy
- Tworzenie ankiet i analiza wyników

Ćwiczenia

- Stwórz arkusz kalkulacyjny rozwiązujący dowolny problem biznesowy, np. Obliczanie ceny produktów dla klienta zgodnie z polityką cenową firmy

Ćwiczenia

- Utwórz arkusz z wynikami egzaminu dla 15 studentów
  - Kolumny: Nr indeksu, imię, nazwisko, punkty, ocena, status
- W kolumnie ocena użyj zagnieżdżonych funkcji if do przypisania ocen, np:
  - 91-100: 5.0
  - 81-90: 4.5
  - 71-80: 4.0
  - .....
- W kolumnie status użyj formuły IF: jeżeli punkty >= 51, „zaliczony“, w przeciwnym razie „niezaliczony“

Ćwiczenia

- Stwórz harmonogram 10-dniowego projektu badawczego
  - Kolumny: Zadanie, Data rozpoczęcia, Czas trwania (dni), Data zakończenia, Dni do końca, Status
- Wprowadź 10 zadań i wprowadź różne daty rozpoczęcia i czasy trwania
- Oblicz datę zakończenia oraz oblicz „dni do końca“
- W kolumnie status użyj IF
  - jeśli dni do końca <0 to status Opóźnione
  - Jeśli dni do końca <=2 to status Pilne
- Sformatuj komórki koloru (formatowanie warunkowe) – Opóźnione na czerwono, pilne na pomarańczowo, itp

Wprowadzenie do grafiki komputerowej

Wprowadzenie do grafiki komputerowej

- Podstawy reprezentacji obrazów cyfrowych
- Narzędzia i techniki edycji
- Zastosowania, np. w pracy akademickiej
- Czym jest grafika komputerowa?

Wprowadzenie do grafiki komputerowej

- Podstawy reprezentacji obrazów cyfrowych
- Narzędzia i techniki edycji
- Zastosowania, np. w pracy akademickiej
- Czym jest grafika komputerowa?
  - Cyfrowa reprezentacja obrazów wizualnych
  - Tworzenie, manipulacja i wyświetlanie obrazów za pomocą komputera
  - Zastosowania: nauka, sztuka, przemysł, rozrywka
  - Dwa główne podejścia: grafika rastrowa i wektorowa

Grafika rastrowa

- Reprezentacja obrazu – siatka pikseli
- Pixel (picture element) – najmniejszy element obrazu
- Obraz to prostokątka siatka pikseli, każdy piksel, oprócz przypisanej lokalizacji, ma również przypisaną intensywność i kolor.
- Rozdzielczość: liczba pikseli w obrazie (np. 1920 x 1080)

Formaty grafiki rastrowej

- JPEG (.jpg) – zdjęcia, kompresja stratna (usuwanie części danych dla zmniejszenia rozmiaru), nie obsługuje przezroczystości
- PNG (.png) – ilustracje, przezroczystość, kompresja bezstratna
- GIF (.gif) – animacje, 256 kolorów
- BMP (.bmp) – bez kompresji, duże pliki
- TIFF (.tiff) – fotografia, druk

Grafika wektorowa

- Obiekty geometryczne: punkty, linie, krzywe, wielokąty
- Każdy obiekt jest opisany równaniami matematycznymi
- Skalowanie bez utraty jakości
- Idealna dla logo, ikon, diagramów, tekstu

Formaty grafiki wektorowej

- SVG (.svg) – Scalable Vector Graphics
- PDF (.pdf) – dokumenty, zawiera tekst i obrazy
- EPS (.eps) – stary standard drukarniczy
- AI (.ai) – format Adobe Illustrator
- Istnieje możliwość eksportu do formatów rastrowych

Kiedy używać rastra, a kiedy wektora?

- Grafika rastrowa:
  - Fotografie, realistyczne obrazy
  - Złożone efekty wizualne (cienie, gradienty)
  - Obrazy z dużą ilością kolorów i szczegółów
- Grafika wektorowa:
  - Logo i identyfikatory
  - Diagramy, schematy, infografiki
  - Tekst, typografia
  - Obrazy wymagające skalowania

Rozdzielczość obrazu

- Mierzona w pikselach: szerokość x wysokość
- 640x480 (VGA), 1920x1080 (Full HD), 3840x2160 (4K)
- Megapiksle = szerokość x wysokość / 1 000 000
- DPI/PPI – Dots Per Inch / Pixels Per Inch – gęstość pikseli w druku/na ekranie

Głębia koloru

- Bitowa głębia określa liczbę możliwych kolorów
- **1-bit**: czarno-biały (2 kolory)
- **8-bit** (grayscale): 256 odcieni szarości
- **24-bit** (True Color): 16.7 miliona kolorów (8 bitów na kanał RGB)
- **32-bit**: 24-bit kolor + 8-bit przezroczystość

Model reprezentacji kolorów

- **RGB** – Red, Green, Blue (ekrany, monitory) – *model addytywny*
- **CMYK** – Cyan Magenta, Yellow, Black (druk) – *model subtraktywny*
- **HSB/HSV** – Hue, Saturation, Brightness
- **Grayscale** – odcienie szarości
- Konwersja między przestrzeniami barw może zmieniać kolory.

Kompresja obrazów

- Kompresja bezstratna (lossless):
  - Zachowuje wszystkie informacje
  - Algorytmy typu ZIP
  - Formaty: PNG, GIF, BMP (z kompresją)
  - Większe pliki
- Kompresja stratna (lossy)
  - Usuwa część informacji
  - Drastyczna redukcja rozmiaru
  - Formaty: JPEG
  - Artefakty przy niskiej jakości lub wielokrotnej kompresji

Narzędzia – oprogramowanie graficzne

- Grafika rastrowa:
  - Adobe Photoshop
  - GIMP (darmowy, open-source, wiele platform)
  - Paint (Windows)
  - Photopea (w przeglądarce)
- Grafika wektorowa:
  - Adobe Illustrator
  - Inkscape
  - Figma

Podstawowe operacje na obrazie

- Kadrowanie
  - Crop, ustalenie proporcji, usuwanie zbędnych elementów, skupienie się na głównym obiekcie
- Skalowanie
  - Resize, zmiana wymiarów w pikselach, Resampling – przeliczanie pikseli przy zmianie rozmiaru, Upsizing/Downsizing
- Regulacja jasności i kontrastu
  - Brightness, Contrast – do niedoświetlonych/prześwietlonych zdjęć
- Korekcja i modyfikacja barw
  - Hue (odcień), Saturation (nasycenie), Color balance, Desaturation

Warstwy – niedestrukcyjna edycja obrazów

- Warstwa – niezależny poziom obrazu
- Nakładanie warstw jedna na drugiej
- Kolejność warstw: górne przykrywają dolne
- Przezroczystość warstw (opacity)
- Tryby mieszania
- Zastosowanie – warstwy pozwalają na dodawanie adnotacji do obrazów (np. Mikroskopowych) bez modyfikowania oryginału

Zaznaczenia i maski

- Selection tool – zaznaczanie obszarów obrazu
- Prostokątne, eliptyczne, lasso (ręczne), magic wang (np. Wg koloru)
- Operacje na zaznaczeniu: kopiuj, usuń, wypełnij, filtruj
- Maski – zaawansowana kontrola przezroczystości
- Feathering – rozmycie krawędzi zaznaczenia

Filtry i efekty

- Popularne filtry:
  - Blur (rozmycie) – Gaussian, motion blur
  - Sharpen (wyostrzenie) – zwiększa ostrość krawędzi
  - Noise reduction (usuwanie szumu/ziarna)
  - Edge detection – wykrywanie konturów
  - Artistic filters – efekty malarskie, szkicu, itp.

Dodawanie tekstu i kształtów

- Text tool – dodawanie tekstu
- Definiujemy czcionkę, rozmiar, kolor, wyrównanie
- Kształty podstawowe: prostokąt, okrąg, linia, strzałka
- Grupowanie elementów
- Przydatne dla etykiet, diagramów, wskazywania szczegółów

Eksport i zapisywanie

- Save vs. Export
- Wybór formatu: PNG dla grafiki, JPEG dla zdjęć
- Kompresja: balans między rozmyciem a jakością
- Embedding metadata (EXIF, IPTC) – metadane, informacje o autorze, prawach, itd
- Wersjonowanie: zapis pliku roboczego (PSD/XCF) + eksport fo publikacji

Standardy obrazów w publikacjach naukowych

- Wymagania czasopism: rozdzielczość, format, rozmiar
- Typowe: min. 300 DPI, TIFF lub high-quality JPEG
- Skale wielkości i paski skali
- Legendy i opisy: zwięzłe, informacyjne
- Spójność stylu między figurami w jednej publikacji (np. „Figure 1. [tytuł][opis]“

Prawa autorskie i licencje

- Copyright – domyślna ochrona twórczości
- Public Domain – bez praw autorskich, swobodne użycie
- Creatvie Commons – różne poziomy ograniczeń
- Stock photos: płatne i darmowe
- Zawsze podaj źródło obrazu w publikacjach

Ćwiczenia

- Wejdź na photopea.com
- Stwórz plakat filmowy dowolnego filmu
- Wykorzystaj warstwy, zdjęcia, tekst
- Dodaj autora grafiki
- Wyeksportuj do PNG lub JPEG

Ćwiczenia

- Pobierz dowolne zdjęcie mikroskopowe
- W photopea wykonaj edycję zdjęcia, dokonaj anotacji obserwacji. Zaznacz, powiększ, zilustruj, opisz
- Wyeksportuj do PNG
- Za pomocą chataGPT stwórz opis naukowy obserwacji, wykorzystując stworzoną grafikę naukową, stwórz notatkę w wordzie.

Ćwiczenia

- W photopea stwórz logo/ikonę:
  - Koła naukowego
  - Projektu akademickiego
  - Zespołu heavy metalowego
- Wymagania: 500x500 pikseli
- Tylko kształty geometryczne + tekst
- Maksymalnie 3 kolory
- Minimalistyczny design

Wprowadzenie do baz danych

Wprowadzenie do baz danych

- Podstawy organizacji i przechowywania danych
- Model relacyjny i język SQL
  - Dane grupowane są w relacje, które reprezentowane są za pomocą tabel
  - SQL – Structured Query Language – uniwersalny standard
- Praktyczne zastosowania w nauce i biznesie

Czym jest baza danych?

- Uporządkowany zbiór powiązanych danych
- Zarządzana przez system DBMS (Database Management System)
- Zapewnia: przechowywanie, bezpieczeństwo, spójność (np. data urodzenia studenta nie może być w przyszłości), współbieżny dostęp
- Od prostych plików po rozproszone systemy

Baza danych a arkusz kalkulacyjny

- Arkusz kalkulacyjny:
  - Małe zbiory danych
  - Jednorazowe analizy
  - Jeden użytkownik
  - Dużo obliczeń, wizualizacji
- Baza danych:
  - Duże zbiory danych
  - Dane strukturalne z relacjami
  - Wielu użytkowników jednocześnie
  - Integralność i bezpieczeństwo danych

Model relacyjny

- Fundamenty współczesnych baz danych:
  - Zaproponowany przez Edgara Codda (lata 1970)
  - Dane organizowane są w tabele (z relacjami pomiędzy nimi)
  - Tabela – zbiór rekordów o tej samej strukturze
  - Relacje między tabelami przez klucze
  - Standard w przemyśle i biznesie

Struktura tabeli

- Tabela (table) – kolekcja powiązanych danych
- Rekord/Wiersz (Record/Row) – pojedynczy wpis
- Pole/Kolumna (Field/Column) – atrybut rekordu
- Komórka (Cell) – wartość w konkretnym miejscu
- Schemat tabeli – definicja struktury

Typy danych w bazach

- INTEGER – liczby całkowite
- FLOAT – liczby zmiennoprzecinkowe
- VARCHAR(n) – tekst o zmiennej długości (max n znaków)
- TEXT – długie teksty
- DATE, TIME, DATETIME – daty i czas
- BOOLEAN - ?

Klucz główny (primary key)

- Kolumna jednoznacznie identyfikująca rekord
- Każda wartość klucza musi być unikalna
- Nie może być NULL (pusta)
- Często automatycznie zwiększana wartość (1, 2, 3, 4....)
- Używany do tworzenia relacji między tabelami

Klucz obcy (foreign key)

- Kolumna w jednej tabeli odwolująca się do PRIMARY KEY innej tabeli
- Tworzy relację rodzic-dziecko
- Zapewnia integralność referencyjną
- Przykład: tabela WYPOŻYCZENIA ma kolumnę id_studenta
- Baza nie pozwoli usunąć studenta mającego wypożyczenia

Rodzaje relacji między tabelami

- Jeden-do-jednego – rzadko używane (np. osoba - paszport)
- Jeden-do-wielu – najczęstsze (np. Autor-książki)
- Wiele-do-wielu – wymaga tabeli pośredniczącej, tzw. Junction table zawierająca dwa klucze obce (np. studenci-kursy)

Język SQL - wprowadzenie

- SQL to standardowy język komunikacji z bazami relacyjnymi
- Składnia podobna do jęz. angielskiego
- Deklaratywny – opisujemy co chcemy uzyskać a nie jak („*chcę imiona studentów starszych niż 25 lat*“ a nie „*otwórz tabelę, iteruj po wierszach, sprawdź każdy wiek*“)
- Kategorie poleceń:
  - DDL (Data Definition Language) – tworzenie struktury (CREATE, DROP)
  - DML (Data Manipulation Language) – operacje na danych (SELECT, INSERT)
  - DCL (Data Contol Language) – kontrola dostępu (GRANT, REVOKE)

SELECT

- Pobieranie danych z tabeli:
  - SELECT kolumna1, kolumna2 FROM nazwa_tabeli;
  - SELECT * FROM studenci;
  - SELECT imię, nazwisko FROM studenci;
- SELECT – co chcemy pobrać
- FROM – z której tabeli
- * wszystkie kolumny
- Średnik na końcu polecenia

WHERE

- Wybieranie rekordów spełniających warunek:
  - SELECT * FROM studenci WHERE wiek > 20;
  - SELECT imie FROM studenci WHERE kierunek = 'Bioinformatyka';
  - SELECT * FROM pomiary WHERE temperatura < 37.0;
- Operatory porównania (`=`, `>`, `<`, `>=`, `<=`, `<>`)
- Teksty w apostrofach

Operatory logiczne

- Łączenie wielu warunków:
  - SELECT * FROM studenci WHERE wiek > 20 AND kierunek = 'Biologia';
  - SELECT * FROM studenci WHERE rok = 1 OR rok = 2;
  - SELECT * FROM studenci WHERE NOT (kierunek = 'Informatyka');
- AND – oba warunki muszą być spełnione
- OR – przynajmniej jeden warunek musi być spełniony
- NOT – negacja warunku
- Nawiasy dla złożonych wyrażeń

ORDER BY

- Uporządkowanie rekordów:
  - SELECT * FROM studenci ORDER BY nazwisko;
  - SELECT * FROM studenci ORDER BY wiek DESC;
  - SELECT imie, ocena FROM studenci ORDER BY ocena DESC, nazwisko ASC;
- ORDER BY kolumna – sortowanie rosnące
- DESC – malejące
- ASC – rosnące
- Sortowanie po wielu kolumnach

LIMIT

- Pobieranie tylko pierwszych N rekordów:
  - SELECT * FROM studenci LIMIT 10;
  - sSELECT * FROM produkty ORDER BY cena DESC LIMIT 5;
- Zwraca tylko określoną liczbę pierwszych rekordów
- Często używane z ORDER BY
- Przydatne do: podglądu danych, top N, paginacji

Funkcje agregujące

- Obliczenoa na zbiorach rekordów:
  - SELECT COUNT(*) FROM studenci;
  - SELECT AVG(ocena) FROM studenci WHERE kierunek = 'Biologia';
- COUNT(*) - liczba rekordów
- SUM(kolumna) – suma wartości
- AVG(kolumna) – średnia
- MIN(kolumna) – wartość minimalna
- MAX(kolumna) – wartość maksymalna

GROUP BY

- Agregacja w podgrupach
  - SELECT kierunek, COUNT(*) FROM studenci GROUP BY kierunek;
  - SELECT rok, AVG(ocena) FROM studenci GROUP BY rok;
  - SELECT kategoria, COUNT(*), AVG(cena) FROM produkty GROUP BY kategoria;
- Dzieli na drupy według wartości w kolumnach
- Funkcje agregujące obliczane osobno dla każdej grupy

HAVING

- Warunki na wynikach agregacji:
  - SELECT kierunek, COUNT(*) FROM studenci
    GROUP BY kierunek
    HAVING COUNT(*) > 30;

  - SELECT kategoria, AVG(cena) FROM produkty
    GROUP BY kategoria
    HAVING AVG(cena) < 100;
- WHERE - filtruje rekordy przed grupowaniem
- HAVING – filtruje grupy po agregacji

INSERT

- Wstawianie nowych rekordów:
  - INSERT INTO studenci (imie, nazwisko, wiek, kierunek)
    VALUES ('Jan', 'Kowalski', 21, 'Biologia');
  - INSERT INTO studenci (imie, nazwisko)
    VALUES ('Anna', 'Nowak');
- Określamy tabelę i kolumny
- VALUES – wartości do wstawienia
- Liczba i kolejność wartości musi odpowiadać kolumnom
- Kolumny z domylnymi wartościami można pominąć

UPDATE

- Zmiana istniejących rekordów:
  - UPDATE studenci SET wiek = 22 WHERE id = 5;
  - UPDATE studenci SET kierunek = 'Bioinformatyka' WHERE kierunek = 'Biologia' AND rok = 1;
  - UPDATE produkty SET cena = cena * 1.1;
- SET – nowa wartość
- WHERE określa które rekordy zmienić
- UWAGA! Bez WHERE zmienią się WSZYSTKIE rekordy

DELETE

- Usuwanie rekordów z tabeli:
  - DELETE FROM studenci WHERE id = 10;
  - DELETE FROM pomiary WHERE data < '2020-01-01';
  - DELETE FROM studenci; - **usuwa WSZYSTKIE rekordy!**
- WHERE określa które rekordy usunąć
- Bez WHERE usuniesz wszystkie rekordy
- Usunięcie jest nieodwracalne (jedynie przez backup)

JOIN

- Pobieranie danych z wielu tabel:
  - SELECT studenci.imie, kursy.nazwa

    FROM studenci

    JOIN zapisy ON studenci.id = zapisy.id_studenta

    JOIN kursy ON zapisy.id_kursu = kursy.id;
- INNER JOIN – tylko rekordy mające odpowiedniki w obu tabelach
- LEFT JOIN – wszystkie z lewej + dopasowanie z prawej
- Warunek łączenia: ON kolumna1=kolumna2

Narzędzia do pracy z bazami

- Zarządzanie:
  - SQLite – lekka,idealna do nauki
  - MySQL/MariaDB – popularna open-sourse, web
  - PodtgreSQL – zaawansowana open-source
  - Oracle, MS SQL Server – komercyjne, enterprise
- Interfejsy:
  - DB Browser
  - PhpMyAdmin webowy dla MySQL

Projektowanie bazy danych

- Identyfikacja encji (rzeczy do przechowania)
- Określenie atrybutów każdej encji
- Ustalenie relacji między encjami
- Definicja kluczy głównych i obcych
- Normalizacja (unikanie redundancji)

Ćwiczenia

- Stwórz prostą relacyjną bazę danych w excelu:
- Zakładki:
  - Książki (ID, tytuł, rok wydania, ID autora, Kategoria), stwórz 15 książek
  - Autorzy (ID autora, Imię, Nazwisko, Narodowość), stwórz 8-10 autorów
  - Wypożyczenia (ID wypożyczenia, ID książki, Data wypożyczenia, Data zwrotu, Wypożyczający), stwórz 10 rekordów
  - Analizy
- Do Analizy stwórz listę książek (VLOOKUP „=VLOOKUP(ID_autora, AUTORZY!A:D, 2, FALSE)“ dla imienia), stwórz tabel
- Obok stwórz tabelę ze statystykami: liczba książek ogółem, liczba wg kategorii (COUNTIF), najstarszy rok wydania (MIN), najnowszy rok wydania (MAX)
- Filtruj (pokaż tylko książki naukowe, tylko wydane po 2010 roku)
- Posortuj książki alfabetycznie

Ćwiczenia

- Zaprojektuj schemat bazy danych dla systemu zarządzania laboratorium.
- Wymagane tabele:
  - PRACOWNICY (naukowcy, technicy), PROJEKTY_BADAWCZE, SPRZĘT_LABORATORYJNY, REAGENTY, EKSPERYMENTY, PUBLIKACJE
- Dla każdej tabeli określ:
  - Nazwę tabeli
  - Listę kolumn z typami danych
  - Klucz główny
  - Klucze obce – jeśi są
  - Krótki opis co tabela przechowuje
- Opisz zadania (3 – 5) jakie można wykonać na tej bazie danych (przykładowe operacje biznesowe)

Wprowadzenie do metod programowania

Prezentacja jest dostępna na:

https://git.michalkasprowicz.com/administrator/WSKZ

Wprowadzenie do metod programowania

- Podstawy myślenia algorytmicznego
- Fundamenty języków programowania
- Python jako pierwszy język
- Komunikacja z komputerem:
  - Program to zestaw instrukcji dla komputera
  - Algorytm to przepis na rozwiązanie problemu
  - Język programowania to formalny sposób wyrażania algorytmów
  - Kompilacja / interpretacja to tłumaczenie na język maszynowy

Paradygmaty programowania

- Paradygmat – styl/filozofia programowania
- Określa jak organizujemy i strukturyzujemy kod
- Główne paradygmaty:
  - Imperatywny – sekwencja poleceń
  - Proceduralny – funkcje/procedury
  - Obiektowy – obiekty i klasy
  - Funkcyjny – funkcje matematyczne
  - Deklaratywny – co osiągnąć, nie jak

Geneza programowania obiektowego

- Od chaosu do organizacji
- Problem w dużych programach:
  - Tysiące linii kodu w jednym pliku
  - Funkcje operujące na globalnych zmiennych
  - „Spaghetti code“ – niemożliwy do utrzymania
  - Trudna współpraca zespołowa
  - Błąd w jednym miejscu psuje cały kod
- Rozwiązanie:
  - **Enkapsulacja** – grupowanie danych i funkcji
  - **Modularność** – niezależne komponenty
  - **Reużywalność** – raz napisane, wielokrotnie użyte
  - **Łatwe utrzymanie** – zmiany lokalne, nie globalne

Podstawowe koncepcje OOP

- Klasa - szablon/przepis
  - Definicja struktury danych (atrybuty)
  - Definicja zachowań (metody)
  - Nie zajmuje pamięci (tylko kod)
- Obiekt – konkretna instancja klasy
  - Rzeczywista „rzecz“ w pamięci
  - Posiada konkretne wartości atrybutów
  - Można tworzyć wiele obiektów z jednej klasy

Przykład

```
class Samochod:          # Klasa - szablon
    def __init__(self, marka, kolor):
        self.marka = marka
        self.kolor = kolor



auto1 = Samochod("Toyota", "czerwony")  # Obiekt 1
auto2 = Samochod("BMW", "czarny")       # Obiekt 2
```

Enkapsulacja

- Enkapsulacja:
  - Grupowanie danych i metod w jednostkę (obiekt)
  - Ukrywanie wewnętrznych szczegółów
  - Publiczny interfejs vs prywatna implementacja
  - Modyfikacja dostępu: public, private, protected
- Abstrakcja:
  - Ukrywanie złożoności, pokazywanie tylko istoty
  - Użytkownik widzi CO robi, a nie JAK
  - Przykad: samochód → wciskasz gaz, nie wiesz jak działa silnik

Dziedziczenie

- Budowanie na istniejącym kodzie
- Tworzenie nowej klasy na bazie istniejącej
- Klasa potomna „dziedziczy“ atrybuty i metody klasy rodzica
- Rozszerzenie specjalizacja zachowań
- Przykład:

```
class Zwierze:                  # Klasa bazowa (rodzic)
      def __init__(self, nazwa):
      self.nazwa = nazwa

      def jedz(self):
      print(f"{self.nazwa} je")

class Pies(Zwierze)
       def szczekaj(self):
       print("Hau hau!")

class Kot(Zwierze):
       def miauczaj(self):
       print("Miau!")

rex = Pies("Rex")
rex.jedz()       # Odziedziczona metoda
rex.szczekaj()   # Własna metoda
```

OOP w praktyce

- OOP świetnie sprawdza się gdy:
  - Modelujesz rzeczywiste encje (Student, Książka, Pacjent)
  - Masz złożony system z wielomna powiązanymi komponenetami
  - Potrzebujesz reużywalności i rozszerzalności
  - Projekt długoterminowy, duży zespół
  - Przykłady: systemy biznesowe, gry, symulacje, GUI
- OOP może być przesadą gdy:
  - Prosty skrypt
  - Jednorazowa analiza danych
  - Problem naturalnie proceduralny/funkcyjny

Dlaczego python?

- Prosta, czytelna składnia przypominająca jęz. angielski
- Wszechstronność: web, data science, automatyzacja, AI
- Ogromna społeczność i biblioteki
- Popularny w nauce (bioinformatyka, analiza danych)
- Darmowy, open-source, dostępny na wszystkich platformach

Algorytm – myślenie krok po kroku

- Algorytm – sformalizowany przepis na rozwiązanie
- Cechy dobrego algorytmu:
  - Jednoznaczny (każdy krok jest jasny)
  - Skończony (kończy się w rozsądnym czasie)
  - Skuteczny (rzeczywiście rozwiązuje problem)
  - Uniwersalny (działa dla różnyc danych wejściowych)

Zmienne i typy danych

- Zmienna – „pojemnik“ na wartość
- Typ określa co można przechowywać i jakie operacje wykonywać
- Python określa typ automatycznie
- Przykłady:
  - wiek = 25 # integer (liczba całkowita)
  - temperatura = 36.6 # float (zmiennoprzecinkowa)
  - imie = "Anna" # string (tekst)
  - jest_studentem = True # boolean (prawda/fałsz)

Operacje arytmetyczne

```
a = 10
b = 3
suma = a + b        # 13
roznica = a - b     # 7
iloczyn = a * b     # 30
iloraz = a / b      # 3.333...
calkowity = a // b  # 3 (dzielenie całkowite)
reszta = a % b      # 1 (modulo)
potega = a ** b     # 1000 (10^3)
```

Operatory porównania i logiczne

- Porównania:

```
a == b  # równe
a != b  # różne
a > b   # większe
a < b   # mniejsze
a >= b  # większe równe
a <= b  # mniejsze równe
```

- Logiczne:

```
a and b  # oba prawdziwe
a or b   # przynajmniej jedno prawdziwe
not a    # negacja
```

Instrukcje if/elif/else

- Uwaga, w pythonie wcięcia (indentacja) są obowiązkowe!

```
if warunek:
    # wykonaj gdy prawda
    print("Warunek spełniony")
elif inny_warunek:
    # wykonaj gdy pierwszy fałsz, ten prawda
    print("Inny warunek spełniony")
else:
    # wykonaj gdy wszystkie fałsz
    print("Żaden warunek")
```

Pętla for

- Powtarzanie operacji
- Iteruje przez sekwencję (zakres, liczna, string)
- Dla każdego elementu wykonuje blok kodu

```
for i in range(5):
    print(i)  # wypisze 0, 1, 2, 3, 4
```

```
for imie in ["Anna", "Jan", "Ewa"]:
    print(f"Witaj, {imie}!")
```

```
for znak in "DNA":
    print(znak)  # D, N, A
```

Pętla while

- Powtarzanie dopóki warunek jest prawdziwy

```
licznik = 0
while licznik < 5:
    print(licznik)
    licznik += 1  # ważne! bez tego: pętla nieskończona


# Pętla aż użytkownik zgadnie
liczba = 7
while True:
    odpowiedz = int(input("Zgadnij liczbę: "))
    if odpowiedz == liczba:
        break  # wyjście z pętli
```

Funkcje - definicja

```
def powitanie(imie):
    print(f"Witaj, {imie}!")


powitanie("Anna")  # wywołanie


def pole_prostokata(a, b):
    wynik = a * b
    return wynik  # zwraca wartość


powierzchnia = pole_prostokata(5, 3)  # 15
```

Przekazywanie danych do funkcji

```
def oblicz_bmi(waga, wzrost):
    return waga / (wzrost ** 2)

# Argumenty pozycyjne
bmi = oblicz_bmi(70, 1.75)

# Argumenty nazwane
bmi = oblicz_bmi(waga=70, wzrost=1.75)

# Wartości domyślne
def powitanie(imie, formalnie=False):
    if formalnie:
        print(f"Dzień dobry, {imie}")
    else:
        print(f"Cześć, {imie}")
```

Listy – kolekcje danych

```
Liczby = [1, 2, 3, 4, 5]
imiona = ["Anna", "Jan", "Ewa"]
mieszana = [1, "tekst", 3.14, True]


# Dostęp przez indeks (od 0!)
print(liczby[0])  # 1 (pierwszy)
print(liczby[-1]) # 5 (ostatni)


# Modyfikacja
liczby[0] = 10
liczby.append(6)  # dodaj na końcu
liczby.remove(10) # usuń wartość
```

Słowniki – pary klucz-wartość

```
Student = {
    "imie": "Anna",
    "wiek": 21,
    "kierunek": "Biologia"
}

# Dostęp przez klucz
print(student["imie"])  # Anna
student["email"] = "anna@uni.pl"  # dodanie


# Iteracja
for klucz, wartosc in student.items():
    print(f"{klucz}: {wartosc}")
```

Operacje na tekście

```
tekst = "Bioinformatyka"
print(len(tekst))       # 15
print(tekst.upper())    # BIOINFORMATYKA
print(tekst.lower())    # bioinformatyka
print(tekst[0:3])       # Bio (slicing)
print("Bio" in tekst)   # True (sprawdzanie obecności)


# Łączenie
imie = "Anna"
nazwisko = "Kowalska"
pelne = imie + " " + nazwisko

# lub
pelne = f"{imie} {nazwisko}"
```

Obsługa błędów

```
Try:
    wiek = int(input("Podaj wiek: "))
    print(f"Za rok będziesz mieć {wiek+1} lat")
except ValueError:
    print("To nie jest liczba!")


try:
    wynik = 10 / 0
except ZeroDivisionError:
    print("Nie dziel przez zero!")
except Exception as e:
    print(f"Nieoczekiwany błąd: {e}")
```

Biblioteki i moduły

- Moduł – plik .py z kodem
- Biblioteka – zbiór modułów

```
import math
print(math.sqrt(16))  # 4.0
print(math.pi)        # 3.14159...
```

```
from random import randint
losowa = randint(1, 10)  # losowa liczba 1-10
```

```
import numpy as np
tablica = np.array([1, 2, 3])
```

Dobre praktyki programistyczne

- Pisanie czytelnego i utrzymywalnego kodu
  - Nazewnictwo – znaczące nazwy zmiennych/funkcji
  - Komentarze – wyjaśniaj „dlaczego“, a nie „co“
  - DRY – Don’t repeat youself – unikaj diplikacji, wykorzystuj funkcje
  - Funkcje – małe, jednozadaniowe
  - Formatowanie – konsekwentne wcięcia
  - Testowanie – sprawdzaj czy kod działa dla różnych danych

Języki programowania

- Python to tylko początek
- C/C++ - wydajność, systemy, gry
- Java - Android
- JavaScript - Frontend webowy
- R - statystyka, analiza danych (konkurent Pythona)
- MATLAB - obliczenia naukowe, inżynieria
- SQL - bazy danych

Techniki stosowane w sieciach komputerowych

Techniki stosowane w sieciach komputerowych

- Jak komputery komunikują się ze sobą?
- Podstawy internetu i protokołów sieciowych
- Bezpieczeństwo w sieci
- Czym jest sieć komputerowa?
  - Zbiór połączonych urządzeń (komputery, smartfony, serwery)
  - Cel: wymiana danych, współdzielenie zasobów
  - Komponenty: urządzenia końcowe (hosts), media transmisyjne (kable, WiFi), urządzenia sieciowe (router, switch)

Rodzaje sieci

- PAN (Personal Area Network) – urządzenia osobiste, zasięg ~10 metrów (bluetooth)
- LAN (Local Area Network) – sieć lokalna, w mieszkaniu/biurze
- MAN (Metropolitan Area Network) – miasto/region
- WAN (Wide Area Network) – rozległy obszar, kraj, kontynent
- INTERNET

Adres IP – identyfikator w sieci

- Logiczny adres urządzenia
- Unikalny w danej sieci adres każdego urządzenia
- Dwie wersje: IPv4 oraz IPv6
- IPv4 – 32 bity, zapis: 192.168.1.1. (4 oktety 0-255)
- IPv6 – 128 bitów, zapis 2001:0db8:85a3:8a2e:0370:7334

Adresy IP prywatne i publiczne

- Publiczne
  - Unikalne globalnie
  - Przydzielane przez ISP
  - Ograniczone (IPv4)
- Prywatne
  - Używane wewnątrz LAN
  - NAT przekłada prywatne ← → publiczne

DHCP – automatyczne przydzielanie IP

- Dynamic Host Configuration Protocol
- Automatyczne przypisywanie adresów IP urządzeniom
- Serwer DHCP (w routerze) ma pulę adresów
- Urządzenie wysyła żądanie (DHCP Discover)
- Serwer odpowiada ofertą (DHCP Offer)
- Urządzenie potwierdza (DHCP Request)
- Serwer akceptuje (DHCP ACK)
- Przydzuelenie na czas, potem odnowienie

DNS – system nazw domenowych

- Domain Name System
  - Translacja nazw na adresy IP
  - Ludzie pamiętają nazwy (michalkasprowicz.com), komputery używają IP (157.180.44.123)
  - Hierarchiczny, rozproszony system
  - Serwery DNS odpowiadają na zapytania
  - Domana składa się z poziomów oddzielonych kropkami, czytane od prawej. Przykład:
    git.michalkasprowicz.com
    .com (TLD, top-level domain)
    michalkasprowicz (second level)
    git. (subdomena)

Zagrożenia w sieci

- Malware – wirusy, trojany, ransomware
- Phishing – metoda wyłudzania danych (fałszywa strona/email)
- Man-in-the-middle – podsłuchiwanie komunikacji
- DDoS – Distributed Denial of Service – przeciążenie serwera ruchem
- SQL Injection – ataki na bazy danych aplikacji webowych
- Brut force – łamanie haseł ciągłymi próbami

Jak się chronić?

- Silne hasła – długie, unikalne, losowe, menadżer haseł
- Dwuskładnikowa autentyfikacja (2FA)
- Regularne aktualizacje systemu
- Szyfrowanie
- Backup
- Świadomość (chyba najważniejsza)

Cloud computing

- Zasoby obliczeniowe przez sieć (serwery, storage, aplikacje)
- Modele:
  - IaaS (Infrastructure as a Service) – wirtualne maszyny
  - PaaS (Platform as a Service) – środowisko dla deweloperów
  - SaaS (Software as a Service) – gotowe aplikacje
- Zalety: skalowalność, elastyczność
- Wady: zależność od podłączenia, zależność od dostępnej oferty

Internet rzeczy

- Internet of Things
  - Urządzenia codziennego użytku połączone z internetem
  - Smart home (termostaty, AC, zabezpieczenia, kamery, oświetlenie), Wearables (smartwatche), Smart City
  - Komunikacja: WiFi, bluetooth
  - Wyzwania: bezpieczeństwo, prywatność, standardy

Obliczenia kwantowe

- Rewolucja w przetwarzaniu informacji
- Czym są komputery kwantowe?
  - Wykorzystują zjawiska mechaniki kwantowej do obliczeń
  - Fundamentalnie inny sposób przetwarzania informacji niż komputery klasyczne
  - **Nie zastąpią** zwykłych komputerów – rozwiązują inne problemy
- Kluczowe różnice

Komputer klasyczny	**Komputer kwantowy**
Bit: 0 lub 1	Qubit: 0 i 1 jednocześnie
Deterministyczny	Probabilistyczny
Sekwencyjne przetwarzanie	Przetwarzanie równoległe
Bramki logiczne (AND, OR)	Bramki kwantowe (Hadamard, CNOT)

Komputery kwantowe

- W fazie prototypów ("*Optimistically, now we're in the very, very early vacuum tube era of quantum computing*", Scott Aaronson)
- Max 1000 kubitów
- Wymagają ekstremalnych warunów (temperatury bliskie zera bezwzględnego)
- Zastosowania komercyjne w fazie testów

Przewaga kwantowa

- Kryptografia
  - Algorytm Shora (teoretycznie) łamie szyfrowanie RSA
  - Kryptografia kwantowa – niemożliwa do złamania
  - Przejście na kryptografię postkwantową już się zaczęło
- Symulacje molekularne
  - Projektowanie leków (symulacja białek, interakcje molekularne)
  - Natura jest z założenia kwantowa – kwantowy komputer naturalnie symuluje zjawiska
- Optymalizacja
  - Logistyka, finanse, Machine Learning
- Sztuczna inteligencja
  - QML, QAI, szybsze przeszukiwanie dużych przestrzeni (algorytm Grovera)
- Wyzwania
  - Dekoherencja i korekcja błędów
  - Skalowalność (im więcej kubitów, tym więcej problemów)
  - Brak programistów kwantowych

Ćwiczenie finałowe

- System operacyjny nowej generacji
- **Cel dydaktyczny**: synteza wszystkich poznanych i omawianych technik.
- **Zadanie**: Wykorzystując poznane podczas kursu techniki, zaprojektuj, zaimplementuj w pełni funkcjonalny system operacyjny spełniający następujące wymagania techniczne:
  - **Stabilność**: co najmniej 847% większa niż MS Windows
  - **Wydajność**: szybszy od Supermana w locie międzyplanetarnym
  - **Efektywność pamięciowa**: bardziej oszczędny niż student pierwszego roku podczas sesji
  - **Bezpieczeństwo**: odporniejszy na ataki niż Nokia 3310 na upadki
  - **Kompatybilność**: uruchamia wszystkie programy napisane w przeszłości, teraźniejszości i przyszłości
- Wymagania techniczne
  - **Kod źródłowy**: maksymalnie 150 linii (wliczając w to komentarze)
  - **Czas implementacji**: do końca zajęć, kto skończy wcześniej, może iść do domu
  - **Zależności**: to zależy od czego