1. funksiyaning -2 ≤ *x* ≤ 2 va -2 ≤ *y* ≤ 2 sohalardagi 3D sirt grafigini tuzing. **(10 ball)**
2. Ikkita funksiya parametrik ravishda tenglamalar orqali berilgan:

Bitta oynada 0 ≤ *t* ≤ 2π. uchun y ga x va v ga u funksiyalarini chizing. Grafiklarni shunday

formatlangki, ikkala o‘q ham –2 dan 2 gacha bo'lsin. O’qlarni belgilang va grafikga sarlavha bering.  **(10 ball)**

1. uchun yillik foiz stavkasi r bo'lgan hisobvaraqqa o'tkazilishi kerak bo'lgan depozit hisobvarag'iga oylik P omonatini quyidagi formula yordamida hisoblash mumkin:

Yillik foiz stavkasi 4,85% bo'lsa, 10, 11, 12, 13, 14 va 15 yil davomida 200 000 dollarni to'plash uchun oylik omonat miqdorini hisoblang. Natijalarni ikkita ustunli jadvalda ko'rsating, bu erda birinchi ustun yillar soni, ikkinchi ustun oylik omonat hisoblanadi. **(10 ball)**

1. O'rtacha baho (GPA) ni 0 dan 4 gacha bo'lgan shakl bo'yicha hisoblaydigan maxsus funksiyani yozing, bunda A = 4, B = 3, C = 2, D = 1, va E = 0. Funksiya nomi va argumentlari uchun **av = GPA(g,h)** dan foydalaning. Kirish argumenti **g** – elementlari fanlardagi baholarning raqamli qiymatlari bo'lgan vektor. Kirish argumenti **h** – mos ravishda fanlar bo'yicha tegishli kredit soatlari bo'lgan vektor. Chiqish argumenti **av** esa hisoblangan GPA qiymati (*GPA = (Kredit soatlariga ko'paytirilgan ballar yig'indisi) / (kredit soatlari yig'indisi)*). Quyidagi natijalar bilan talabaning GPA sini hisoblash uchun funksiyadan foydalaning: **(15 ball)**:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *Baholar* | *B* | *A* | *C* | *E* | *A* | *B* | *D* | *B* |
| *Kredit soatlari* | 3 | 4 | 3 | 4 | 3 | 4 | 3 | 2 |

1. Imtihonda olingan 30 ta baholar ro'yxati:31, 70, 92, 5, 47, 88, 81, 73, 51, 76, 80, 90, 55, 23, 43, 98, 36, 87, 22, 61, 19, 69, 26, 82, 89, 99, 71, 59, 49, 64.

0 dan 19 gacha, 20 dan 39 gacha, 40 dan 59 gacha, 60 dan 79 gacha va 80 dan 100 gacha nechta baholar borligini aniqlaydigan kompyuter dasturini yozing. Natijalarni quyidagi shaklda ko'rsating:

0 dan 19 gacha bo'lgan baholar 2 ta talaba

20 dan 39 gacha bo'lgan baholar 4 ta talaba

40 dan 59 gacha bo'lgan baholar 6 ta talaba

va b.q.

(*Maslahat: natijalarni ko'rsatish uchun* ***fprintf*** *buyrug'idan foydalaning.*) **(25 ball)**

**1-savol**

**[x, y] = meshgrid(-2:0.1:2, -2:0.1:2);**

**z = 0.5\*x.^2 + 0.5\*y.^2;**

**surf(x, y, z);**

**xlabel('x'); ylabel('y'); zlabel('z');**

**title('z = 0.5x^2 + 0.5y^2');**

**2savol**

**% Define the range for the parameter t**

**t = linspace(0, 2\*pi, 200); % Use 200 points for a smooth curve**

**% Define the first parametric function: x = cos^3(t), y = sin^3(t)**

**x = cos(t).^3;**

**y = sin(t).^3;**

**% Define the second parametric function: u = sin(t), v = cos(t)**

**u = sin(t);**

**v = cos(t);**

**% Create a new figure**

**figure;**

**% Plot the first parametric function (y vs x)**

**plot(x, y, 'b-', 'LineWidth', 2); % 'b-' for blue solid line, set line width**

**hold on; % Keep the current plot so we can add the second one**

**% Plot the second parametric function (v vs u)**

**plot(u, v, 'r--', 'LineWidth', 2); % 'r--' for red dashed line, set line width**

**% Format the plot**

**xlim([-2 2]); % Set the x-axis limits from -2 to 2**

**ylim([-2 2]); % Set the y-axis limits from -2 to 2**

**% Label the axes**

**xlabel('x and u');**

**ylabel('y and v');**

**% Add a title to the plot**

**title('Parametric Plots of x=cos^3(t), y=sin^3(t) and u=sin(t), v=cos(t)');**

**% Add a legend to distinguish the two plots**

**legend('y vs x (x=cos^3(t), y=sin^3(t))', 'v vs u (u=sin(t), v=cos(t))');**

**% Turn on the grid for better visualization (optional)**

**grid on;**

**% Second Code option:**

**t = linspace(0, 2\*pi, 100);**

**x = cos(t).^3;**

**y = sin(t).^3;**

**u = sin(t);**

**v = cos(t);**

**plot(x, y, 'b', u, v, 'r--');**

**xlim([-2 2]); ylim([-2 2]);**

**xlabel('x and u'); ylabel('y and v');**

**title('Parametric Plots');**

**legend('y vs x', 'v vs u');**

**grid on;**

**3-savol**

**% MATLAB script to calculate the monthly deposit required to reach a future value.**

**% Definition of the problem:**

**% We need to calculate the monthly deposit (P) required to accumulate a future**

**% value (F) of $200,000 over different time periods (N) with an annual**

**% interest rate (r) of 4.85%. The formula provided is:**

**%**

**% P = F \* (r / 12) / ((1 + r / 12)^(12\*N) - 1)**

**%**

**% where:**

**% F = Future value**

**% r = Annual interest rate**

**% N = Number of years**

**% Given parameters:**

**F = 200000; % Future value in dollars**

**r\_annual = 4.85; % Annual interest rate in percentage**

**r = r\_annual / 100; % Annual interest rate in decimal form**

**years = 10:15; % Number of years for which to calculate the monthly deposit**

**% Initialize an empty matrix to store the results**

**results = zeros(length(years), 2);**

**% Loop through each number of years and calculate the monthly deposit**

**for i = 1:length(years)**

**N = years(i);**

**% Apply the formula to calculate the monthly deposit P**

**monthly\_interest\_rate = r / 12;**

**number\_of\_periods = 12 \* N;**

**P = F \* monthly\_interest\_rate / ((1 + monthly\_interest\_rate)^number\_of\_periods - 1);**

**% Store the results in the matrix**

**results(i, 1) = N;**

**results(i, 2) = P;**

**end**

**% Display the results in a two-column table**

**disp('-----------------------------------');**

**disp(' Number of Years | Monthly Deposit ($) ');**

**disp('-----------------------------------');**

**for i = 1:size(results, 1)**

**fprintf(' %2d | %10.2f \n', results(i, 1), results(i, 2));**

**end**

**disp('-----------------------------------');**

**% Explanation of the Code:**

**% 1. Define the problem:**

**% - The problem asks us to calculate the monthly deposit needed to reach a**

**% target amount given an interest rate and a time period. The formula**

**% for this calculation is provided.**

**% 2. Given parameters:**

**% - `F = 200000;`: Sets the future value (target amount) to $200,000.**

**% - `r\_annual = 4.85;`: Sets the annual interest rate to 4.85%.**

**% - `r = r\_annual / 100;`: Converts the annual interest rate from percentage**

**% to decimal form (4.85% becomes 0.0485).**

**% - `years = 10:15;`: Creates a vector containing the number of years for**

**% which we need to perform the calculation (10, 11, 12, 13, 14, and 15).**

**% 3. Initialize results matrix:**

**% - `results = zeros(length(years), 2);`: Creates a matrix named 'results'**

**% filled with zeros. It will have a number of rows equal to the number of**

**% years we are considering and 2 columns (one for the number of years and**

**% one for the calculated monthly deposit).**

**% 4. Loop through each number of years:**

**% - `for i = 1:length(years)`: This loop iterates through each value in the**

**% 'years' vector.**

**% 5. Calculate the monthly deposit (P):**

**% - `N = years(i);`: Gets the current number of years from the 'years' vector.**

**% - `monthly\_interest\_rate = r / 12;`: Calculates the monthly interest rate**

**% by dividing the annual interest rate by 12.**

**% - `number\_of\_periods = 12 \* N;`: Calculates the total number of periods**

**% (months) by multiplying the number of years by 12.**

**% - `P = F \* monthly\_interest\_rate / ((1 + monthly\_interest\_rate)^number\_of\_periods - 1);`:**

**% This line implements the provided formula to calculate the monthly deposit 'P'.**

**% - `(1 + monthly\_interest\_rate)^number\_of\_periods` calculates the future**

**% value factor.**

**% - Subtracting 1 gives the total interest earned factor.**

**% - The formula then calculates the periodic payment needed to reach the**

**% future value.**

**% 6. Store the results:**

**% - `results(i, 1) = N;`: Stores the current number of years in the first**

**% column of the 'results' matrix.**

**% - `results(i, 2) = P;`: Stores the calculated monthly deposit in the**

**% second column of the 'results' matrix.**

**% 7. Display the results in a table:**

**% - `disp('-----------------------------------');` and subsequent `disp`**

**% statements print a header for the table.**

**% - `for i = 1:size(results, 1)`: This loop iterates through each row of**

**% the 'results' matrix.**

**% - `fprintf(' %2d | %10.2f \n', results(i, 1), results(i, 2));`:**

**% This line prints each row of the 'results' matrix in a formatted way.**

**% - `%2d` formats the number of years as an integer with a width of 2.**

**% - `%10.2f` formats the monthly deposit as a floating-point number with**

**% a width of 10 and 2 decimal places.**

**% How to run this code:**

**% 1. Save the code as a .m file (e.g., monthly\_deposit.m).**

**% 2. Open MATLAB.**

**% 3. Navigate to the directory where you saved the file.**

**% 4. Type `monthly\_deposit` in the MATLAB command window and press Enter.**

**% This will execute the script and display the table of results.**

**4-savol**

**function averageGPA = GPA(grades, creditHours)**

**% GPA Calculates the Grade Point Average.**

**% averageGPA = GPA(grades, creditHours) calculates the GPA based on**

**% letter grades and corresponding credit hours.**

**%**

**% Input Arguments:**

**% grades - A cell array or string array of letter grades (e.g., {'A', 'B', 'C'}).**

**% creditHours - A numerical array of credit hours corresponding to the grades (e.g., [3, 4, 3]).**

**%**

**% Output Argument:**

**% averageGPA - The calculated GPA value.**

**%**

**% Grade Definitions:**

**% A = 4**

**% B = 3**

**% C = 2**

**% D = 1**

**% E = 0**

**%**

**% GPA Calculation Formula:**

**% GPA = (Sum of (Grade Points \* Credit Hours)) / (Sum of Credit Hours)**

**% Define the mapping between letter grades and grade points**

**gradeMap = containers.Map({**

**'A', 'B', 'C', 'D', 'E'**

**}, [**

**4, 3, 2, 1, 0**

**]);**

**numCourses = length(grades);**

**if numCourses ~= length(creditHours)**

**error('The number of grades and credit hours must be the same.');**

**end**

**totalGradePoints = 0;**

**totalCreditHours = sum(creditHours);**

**for i = 1:numCourses**

**grade = upper(grades{i}); % Convert grade to uppercase for case-insensitivity**

**if isKey(gradeMap, grade)**

**gradePoint = gradeMap(grade);**

**totalGradePoints = totalGradePoints + (gradePoint \* creditHours(i));**

**else**

**error('Invalid grade: %s. Please use valid letter grades (A, B, C, D, E).', grade);**

**end**

**end**

**if totalCreditHours == 0**

**averageGPA = 0; % Avoid division by zero if there are no credit hours**

**else**

**averageGPA = totalGradePoints / totalCreditHours;**

**end**

**end**

**%% Example Usage:**

**% Define the student's grades and credit hours as provided in the image**

**studentGrades = {'B', 'A', 'C', 'E', 'A', 'B', 'D', 'B'};**

**studentCreditHours = [3, 4, 3, 4, 3, 4, 3, 2];**

**% Calculate the GPA using the GPA function**

**gpaResult = GPA(studentGrades, studentCreditHours);**

**% Display the result**

**disp(['The calculated GPA is: ', num2str(gpaResult)]);**

**5-savol**

**% List of exam scores**

**scores = [31, 70, 92, 5, 47, 88, 81, 73, 51, 76, 80, 90, 55, 23, 43, 98, 36, 87, 22, 61, 19, 69, 26, 82, 89, 99, 71, 59, 49, 64];**

**% Define the score ranges**

**ranges = [0, 20, 40, 60, 80, 101]; % The upper bound is exclusive, so use 101 for 80-100**

**% Initialize counters for each range**

**rangeCounts = zeros(1, length(ranges) - 1);**

**% Iterate through the scores and count how many fall into each range**

**for i = 1:length(scores)**

**score = scores(i);**

**if score >= ranges(1) && score < ranges(2)**

**rangeCounts(1) = rangeCounts(1) + 1;**

**elseif score >= ranges(2) && score < ranges(3)**

**rangeCounts(2) = rangeCounts(2) + 1;**

**elseif score >= ranges(3) && score < ranges(4)**

**rangeCounts(3) = rangeCounts(3) + 1;**

**elseif score >= ranges(4) && score < ranges(5)**

**rangeCounts(4) = rangeCounts(4) + 1;**

**elseif score >= ranges(5) && score < ranges(6)**

**rangeCounts(5) = rangeCounts(5) + 1;**

**end**

**end**

**% Display the results using fprintf in the specified format**

**fprintf('0 dan 19 gacha bo''lgan baholar \t\t %d ta talaba\n', rangeCounts(1));**

**fprintf('20 dan 39 gacha bo''lgan baholar \t\t %d ta talaba\n', rangeCounts(2));**

**fprintf('40 dan 59 gacha bo''lgan baholar \t\t %d ta talaba\n', rangeCounts(3));**

**fprintf('60 dan 79 gacha bo''lgan baholar \t\t %d ta talaba\n', rangeCounts(4));**

**fprintf('80 dan 100 gacha bo''lgan baholar \t\t %d ta talaba\n', rangeCounts(5));**

**% second code option:**

**scores = [31, 70, 92, 5, 47, 88, 81, 73, 51, 76, 80, 90, 55, 23, 43, 98, 36, 87, 22, 61, 19, 69, 26, 82, 89, 99, 71, 59, 49, 64];**

**ranges = [0 20 40 60 80 101];**

**counts = histcounts(scores, ranges);**

**fprintf('0 dan 19 gacha bo''lgan baholar \t %d ta talaba\n', counts(1));**

**fprintf('20 dan 39 gacha bo''lgan baholar \t %d ta talaba\n', counts(2));**

**fprintf('40 dan 59 gacha bo''lgan baholar \t %d ta talaba\n', counts(3));**

**fprintf('60 dan 79 gacha bo''lgan baholar \t %d ta talaba\n', counts(4));**

**fprintf('80 dan 100 gacha bo''lgan baholar \t %d ta talaba\n', counts(5));**