Semestrálna práca

Ľubomír Samek

3ZIE10

MATLAB Terminal app COVID-19

Obsah

[Cieľ práce 3](#_Toc121862686)

[Príprava 4](#_Toc121862687)

[Náčrt 5](#_Toc121862688)

[Programovanie 6](#_Toc121862689)

[Testovanie 7](#_Toc121862690)

[Dokončovanie 8](#_Toc121862691)

[Stručný návod na obsluhu 9](#_Toc121862692)

[Záver 10](#_Toc121862693)

[Prílohy 11](#_Toc121862694)

[Zdroje a zoznam použitej literatúry 27](#_Toc121862695)

# Cieľ práce

Cieľom našej práce, bolo vyhotoviť, plne funkčnú terminálovú aplikáciu s použitím Matlab-u. Vyhotovená aplikácia mala vyzerať ako terminálová aplikácia ktorá čerpá dáta z CSV dokumentu (databázy), pracuje s nimi, vytvára požadované grafy a štatistiky, počíta matematické úkony (medián, počty dní,...).

# Príprava

Príprava pozostávala z nasledujúcich bodov.

1. Obdržanie potrebných súborov a naštudovanie (konkrétne PDF zadanie a CSV súbor s databázou)
2. Inštalácie samotného Matlab-u (konkrétne desktopová verzia pre Windows).
3. Inštalácia programovacieho prostredia Visual Studio Code a potrebných matlab rozšírení a témy ktorá nám pasovali na programovanie s Matlab-om (.m). Prečo VS Code? Je to veľmi intuitívne prostredie na ktoré sme zvyknutý.
4. Ako posledný bod našej prípravy bolo nájsť si vhodný playlist aby nám to išlo lepšie od ruky.

# Náčrt

Najprv sme sa rozhodovali medzi GUI a terminálom. Rozhodli sme sa pre GUI a vytvorili sme jednoduchý náčrt a začali podľa neho pracovať v Matlab-e (viď. Príloha č.: 15). V tom sme narazili na niekoľko problémov.

1. Keďže sme pracovali s GUI programovať sa dalo iba v Matlab-e pretože VS Code nepodporoval súbor .mlapp.
2. Po poukladaní jednotlivých GUI elementov som sa musel naučiť ako sa medzi sebou dajú prepojiť, čo bolo zdĺhavé a dosť pravdepodobné že to nestihneme.
3. Ako posledný a najhlavnejší problém bolo problém pri importe databázy, a prinútenie aby spolupracovala, (keďže databáza neobsahovala iba čísla ale aj nečíselné hodnoty – mimo názvov).

Pre to sme sa nakoniec rozhodli programovať terminálovú aplikáciu. Pre terminálovú aplikáciu neexistuje žiadny náčrt, môžeme povedať že sme to rozdelili na niekoľko malých segmentov ktoré si priblížime v kapitole „Programovanie“.

# Programovanie

Programovanie sme si rozdelili na niekoľko malých segmentov. Ako prvé bolo potrebné načítať dáta z databázy. Bolo potrebné prekonvertovať 2 stĺpce databázy z „string“ na „double“. Po načítaní dát bolo potrebné z nich vykresliť grafy.

Naprogramovali sme funkciu showBar pre zobrazenie jednotlivých grafov, k tejto funkcii bolo potrebné vytvoriť funkciu inputHandler ktorá sa starala o voľbu ktorý graf sa má vykresliť. K funkcii inputHandler bolo potrebné vytvoriť opatrenie pred chybami v kóde a zároveň sprava pre užívateľa, ak zadal zlý znak. Keď tieto funkcie fungovali bolo na čase vytvoriť stručné inštrukcie pre užívateľa aby vedel ako má aplikáciu použiť.

Neskôr prišiel čas na zakomponovanie časového okna a mediánu. Obe funkcie sme zakomponovali vo funkcii showBar a samozrejme inputHandler. Naprogramovali sme aj funkciu checkSwap ktorá uľahčila užívateľovi zadávanie dátumov, aj keď sa užívateľ pomýlil a napísal dátumy naopak, teda ak napísal najprv väčší a potom menší dátum, tak sa dátumy vymenili.

Ako posledná časť zadania bol výpis najlepšieho a najhoršieho segment, na to sme naprogramovali funkciu bestWorstTimeSegment.

V prílohách nájdete aj zdrojový kód (viď. Príloha č.: 1).

# Testovanie

Testovanie pozostávalo z otestovania celého program či funguje všetko správne, či sa niečo nezaseklo, či niečo nezlyhalo. Tak tiež v rámci testovania sme sa venovali negatívnemu testovaniu.

Snažili sme sa cielene robiť chyby ako užívateľ a testovali či niekde nevznikne chyba. Pri negatívnom testovaní sa vyskytli drobné chyby ktorými sme predišli vo funkcii showBar, inputHandler a funkcii checkSwap.

# Dokončovanie

Ako predposledný bod v semestrálnej práci sme sa zaoberali správnym syntaxom, jednoduchosťou a prehľadnosťou kódu, vykomentovaním kódu.

Aby aplikácia vyzerala krajšie v termináli vytvorili sme peknú hlavičku programu. Program sme okrem využívania lokálneho zdroja databázy (off-line version) pripravili na prácu online, pri čom nie je potrebná lokálna databáza ale databáza je importovaná priamo z internetu, stačí internetové pripojenie. (on-line version).

V kapitole “Stručný návod na obsluhu” sa nachádza návod ako používať MATLAB Terminál app COVID-19. (viď. Príloha č.: 14)

# Stručný návod na obsluhu

Po spustení má užívateľ možnosť vypísať:

* 7 rôznych grafov (viď. Príloha č.: 2-8).
* Časové okno v ktorom sa majú dané grafy vypísať (viď. Príloha č.: 9)
* Možnosť vypísať len medián (viď. Príloha č.: 10)
* Okrem samotného grafu sa zobrazí medián a pre koľko dní sa graf zobrazuje. (viď. Príloha č.: 11)
* Možnosť vypísať najhorší/najlepší časový segment (viď. Príloha č.: 12)
* Užívateľ má tiež možnosť ukončiť aplikáciu (viď. Príloha č.: 13)

# Záver

V závere sa nám podarila plne funkčná terminálová aplikácia ktorá dokáže pracovať s databázou, vytvárať rôzne grafy a vykonávať matematické úkony. Ako malú funkciu je možné používať terminálovú aplikáciu aj pri absencii databázy, avšak je potrebné internetové pripojenie.

# Prílohy

%% nadpis terminalovej aplikacie

fprintf('+-+-+-+-+-+-+ +-+-+-+-+-+-+-+-+ +-+-+-+ +-+-+-+-+-+ +-+ +-+-+ \n');

fprintf('|M|A|T|L|A|B| |t|e|r|m|i|n|a|l| |a|p|p| |C|O|V|I|D| |-| |1|9| \n');

fprintf('+-+-+-+-+-+-+ +-+-+-+-+-+-+-+-+ +-+-+-+ +-+-+-+-+-+ +-+ +-+-+ \n');

%% import dat zo suboru

M = readtable('OpenData\_Slovakia\_Covid\_DailyStats.csv','PreserveVariableNames',true);

%% import dat z githubu

% M = readtable('https://raw.githubusercontent.com/Institut-Zdravotnych-Analyz/covid19-data/main/DailyStats/OpenData\_Slovakia\_Covid\_DailyStats.csv');

%% pre "AgTests"a "AgPosit" zmena stringu "NA" na double "0"

M.AgTests = str2double(M.AgTests);

M.AgPosit = str2double(M.AgPosit);

%% vypis instrukcii pre uzivatela

fprintf('Voľby: \n')

fprintf('"1" pre pozitívne PCR testy (celé obdobie) \n')

fprintf('"2" pre PCR testy (denne) \n')

fprintf('"3" pre pozitívne PCR testy (denne) \n')

fprintf('"4" pre počet úmrtí (celé obdobie) \n')

fprintf('"5" pre Ag testy (denne) \n')

fprintf('"6" pre pozitívne Ag testy (denne) \n')

fprintf('"7" pre hospitalizovaných \n')

fprintf('"8" pre najlepší / najhorší časový segment \n')

fprintf('"0" pre ukončenie \n')

%% volba grafu, casoveho okna, medianu a casoveho segmentu ktory sa ma zobrazit, ukoncenie programu

while 1

    chosenBar = inputHandler("Voľba: ", 'i');

    if chosenBar >= 1 && chosenBar <= 7

        dates = inputHandler("Zvoľte časové okno [rrrr-mm-dd/rrrr-mm-dd], [v]: ", 's');

        medianInput = inputHandler("Chcete vypísať len median? [a/N]: ", 's');

        showBar(M, chosenBar, dates, medianInput);

    elseif chosenBar == 8

        segmentInput = inputHandler("Zvoľte časový segment [dd]: ", 'i');

        bestWorstTimeSegment(M, segmentInput);

    elseif chosenBar == 0

        fprintf('Ukončovanie...');

        break

    else

        fprintf('Nesprávny znak, prosím skúste to znova! \n');

        continue

    end

end

%%

*function* showBar(*M*, *x*, *dates*, *medianInput*)

    %% graf vypise vsetky dni + median

    if dates == "v"

        fprintf('Graf zobrazuje všetky zaznamenané dni \n');

        fprintf('Median %.3f \n', median(M{:,x+1}));

        xAxis = M{:,1};

        yAxis = M{:,x+1};

    else

        %% rozdelenie datumov na dva stringy

        dates = split(dates, "/");

        %% najdenie datumov

        try

            dates{1} = find(M.Datum == dates{1});

            dates{2} = find(M.Datum == dates{2});

            dates = checkSwap(dates);

        catch

            fprintf('Nesprávny dátum, prosím skúste to znova! \n');

            return

        end

        fprintf('Graf zobrazuje %.0f deň/dní)\n', dates(2)-dates(1)+1);

        fprintf('Median %.3f \n', median(M{dates(1):dates(2),x+1}));

        %graf

        xAxis = M{dates(1):dates(2),1};

        yAxis = M{dates(1):dates(2),x+1};

    end

    if lower(medianInput) ~= 'a'

        bar(xAxis, yAxis);

        %% nastavenie spravneho formatu cisel pre os Y

        ytickformat('%,.0f')

        ax = gca;

        ax.YAxis.Exponent = 0;

    end

*end*

%% automaticka zmena nespravnej postupnosti datumu, na spravny

*function* *swapped* = checkSwap(*x*)

    if x{1} > x{2}

        swapped = [x{2}, x{1}];

    else

        swapped = [x{1}, x{2}];

    end

*end*

%% rozdelovanie na string (pismena) a int (cisla), opatrenie pred errormi, vypis pre uzivatela

*function* *output* = inputHandler(*prompt*, *datatype*)

    if datatype == 's'

        output = *input*(prompt, 's');

        if isempty(output)

            fprintf("Nesprávny znak, prosím skúste to znova! \n");

            output = "";

        end

    elseif datatype == 'i'

        output = *input*(prompt);

        if isempty(output)

            fprintf("Nesprávny znak, prosím skúste to znova! \n");

            output = 727; %hocico okrem 0-8

        end

    end

*end*

%% najlepsi / najhorsi casovy segment, kedy sa pocet potvrdenych PCR testov, hospitalizovanych alebo umrti zvysil / znizil najviac, v zadanom casovom intervale

*function* bestWorstTimeSegment(*M*, *n*)

    columns = ["Pocet.potvrdenych.PCR.testami" "Pocet.hospitalizovanych" "Pocet.umrti"];

    for column = columns

        loser = [0 0 0];

        winner = [0 0 0];

        last = 0;

        %% zacina na n-tom riadku, prechadza kazdy n-ty riadok, porovnava s poslednym, a ulozi ho,

        for i = n:n:length(M{:, 1})

            total = M{i, column};

            if loser(1) < total-last

                loser(1) = total-last;

                loser(2) = i;

                loser(3) = i+n;

            end

            if winner(1) > total-last || (winner(1) == 0 && winner(2) == 0) % (opatrenie pri prvom tyzdni)

                winner(1) = total-last;

                winner(2) = i;

                winner(3) = i+n;

            end

            last = total;

        end

        fprintf("%s v %d dnovych segmentoch: \n", column, n);

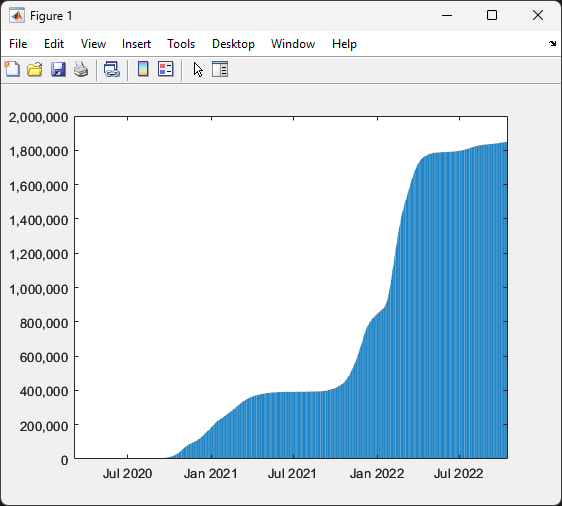
        fprintf("\tNajmenej pripadov: %0.f  medzi %s a %s \n", winner(1), M{winner(2), 1}, M{winner(3), 1});

        fprintf("\tNajviac pripadov: %0.f  medzi %s a %s \n", loser(1), M{loser(2), 1}, M{loser(3), 1});

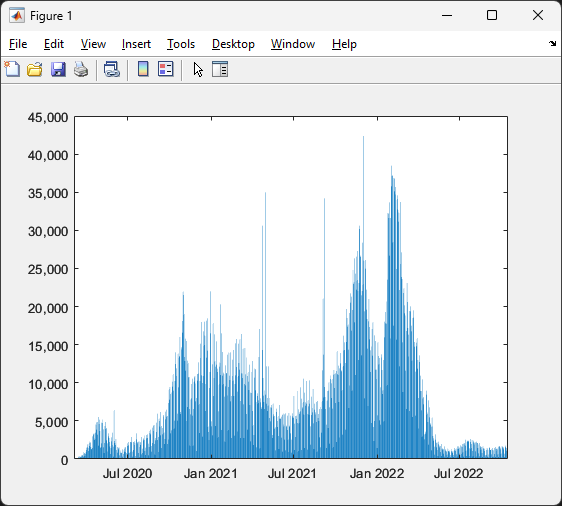
    end

*end*

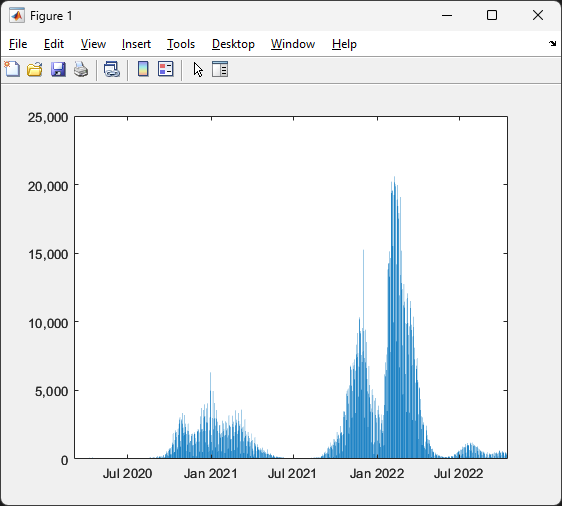
Príloha č.: 1 (Zdrojový kód)



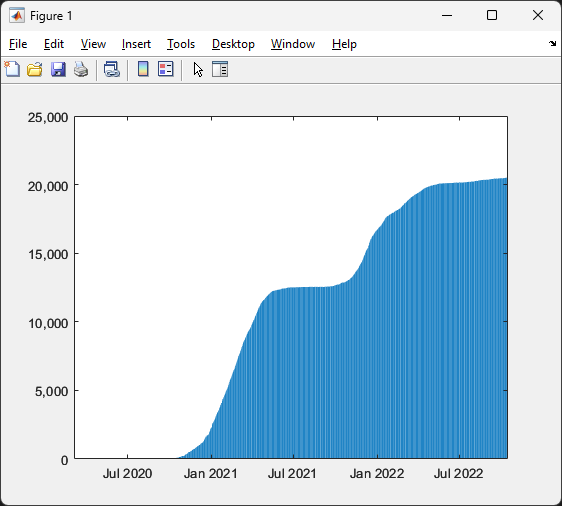
Príloha č.: 2 (Graf č.: 1 - pre pozitívne PCR testy (celé obdobie))



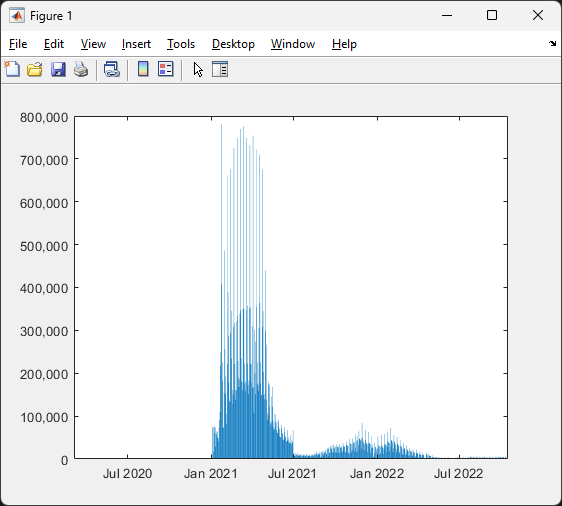
Príloha č.: 3 (Graf č.: 2 - pre PCR testy (denne))



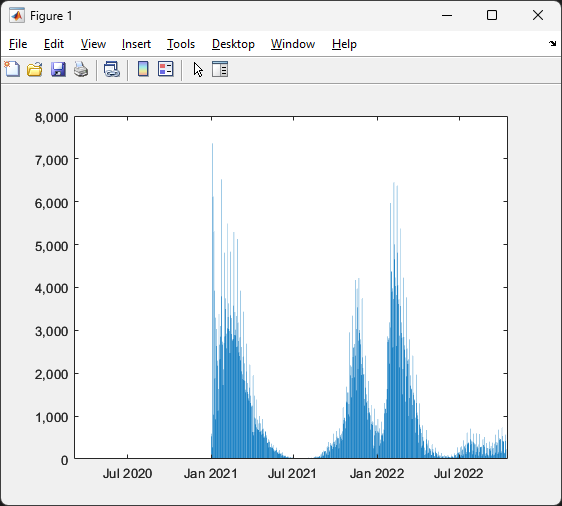
Príloha č.: 4 (Graf č.: 3 - pre pozitívne PCR testy (denne))



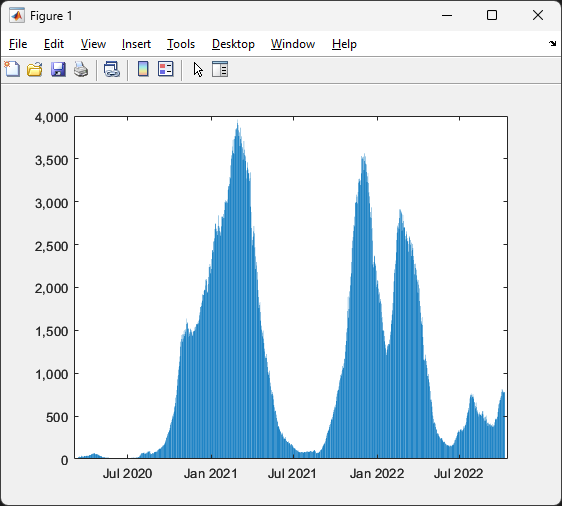
Príloha č.: 5 (Graf č.: 4 - pre počet úmrtí (celé obdobie))



Príloha č.: 6 (Graf č.: 5 - pre Ag testy (denne))



Príloha č.: 7 (Graf č.: 6 - pre pozitívne Ag testy (denne))



Príloha č.: 8 (Graf č.: 7 pre hospitalizovaných)

Obrázok, na ktorom je text

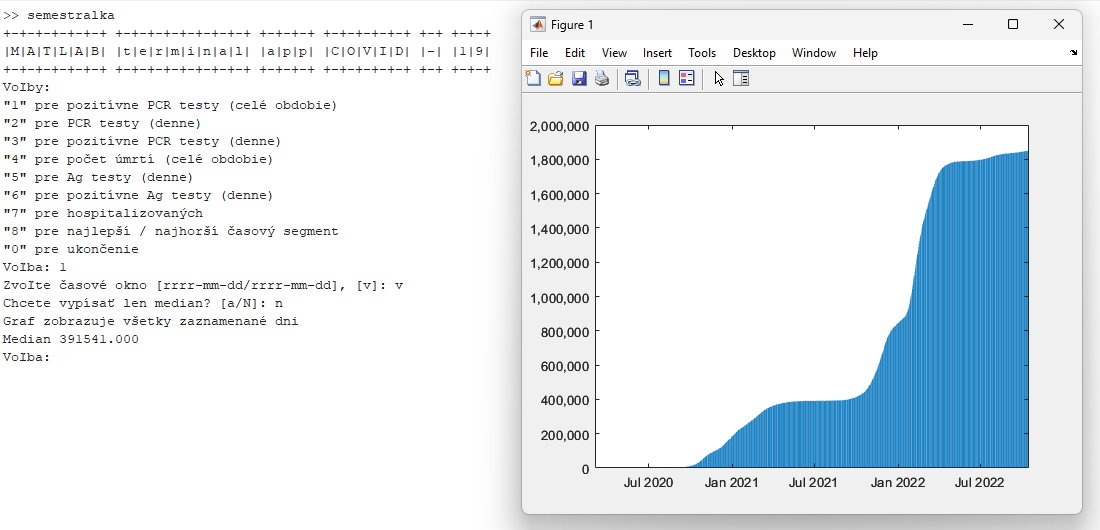
Automaticky generovaný popis

Príloha č.: 9 (Zadávanie časového okna)

Obrázok, na ktorom je text

Automaticky generovaný popis

Príloha č.: 10 (Medián)



Príloha č.: 11 (Náhľad kompletnej terminálovej aplikácie)

Obrázok, na ktorom je text

Automaticky generovaný popis

Príloha č.: 12 (Najhorší/Najlepší časový segment)

Obrázok, na ktorom je text

Automaticky generovaný popis

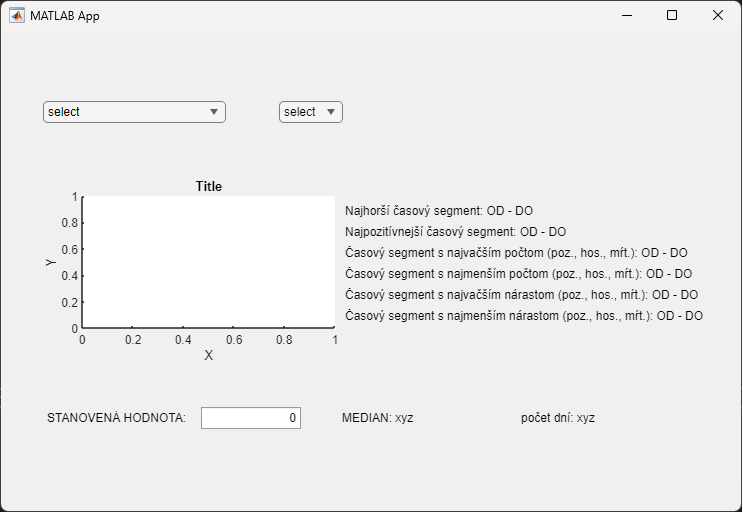
Príloha č.: 13 (Ukončenie)

+-+-+-+-+-+-+ +-+-+-+-+-+-+-+-+ +-+-+-+ +-+-+-+-+-+ +-+ +-+-+

|M|A|T|L|A|B| |t|e|r|m|i|n|a|l| |a|p|p| |C|O|V|I|D| |-| |1|9|

+-+-+-+-+-+-+ +-+-+-+-+-+-+-+-+ +-+-+-+ +-+-+-+-+-+ +-+ +-+-+

Príloha č.: 14 (Nadpis)



Príloha č.: 15 (Matlab GUI)

# Zdroje a zoznam použitej literatúry

<https://www.mathworks.com/products/matlab.html>

<https://code.visualstudio.com/>

<https://www.mathworks.com/help/matlab/ref/.html>

<https://www.mathworks.com/matlabcentral/answers/>