# Dokumentácia k úlohám 1 - 4

*Jozef Barčák, API_2*

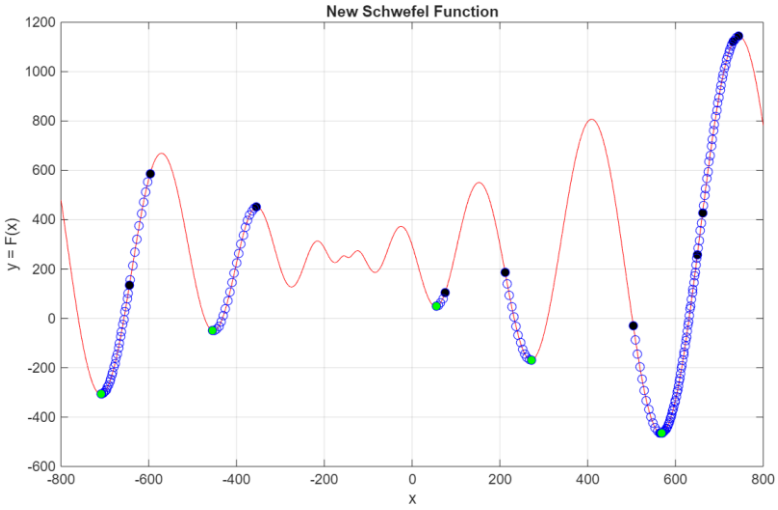
## Úloha č. 1

Použité funkcie:

- testfn3b(Pop) - schwefelova funkcia na nájdenie globálneho minima

**funkcia z toolboxu genetic*

Výsledky:




*Graf dosiahnutých výsledkov (čierny bod – začiatok, zelený bod – koniec, modrý bod - posun)*

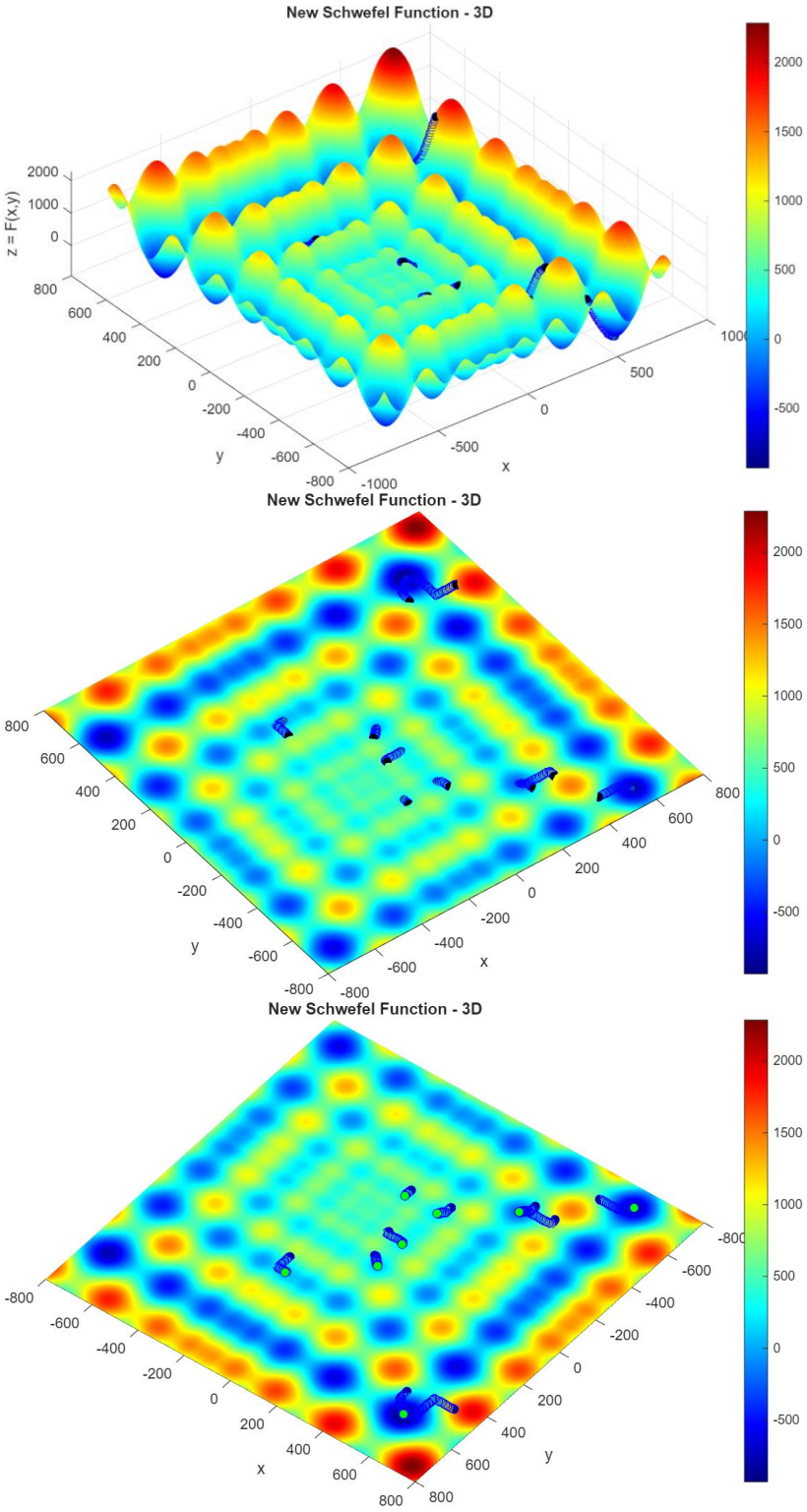
```
Nájdené minimum: x=569  F(x)=-464.603947,
Nájdené minimum: x=565  F(x)=-464.538109,
Nájdené minimum: x=-707  F(x)=-306.578569,
Nájdené minimum: x=567  F(x)=-465.073729,
Nájdené minimum: x=272  F(x)=-168.848647,
Nájdené minimum: x=-709  F(x)=-307.156516,
Nájdené minimum: x=-455  F(x)=-49.768052,
Nájdené minimum: x=55  F(x)=48.336093,
Nájdené minimum: x=567  F(x)=-465.073729,
Nájdené minimum: x=569  F(x)=-464.603947,

Najmenšie minumum: x=567 F(x)=-465.073729
```

*Dosiahnuté výsledky*

### Záver:

- Pri desiatich spusteniach sme dosiahli (skoro) globálne minimum 5-krát. Odchýlka vznikla kvôli posunu na grafe „d = 5“. Presun pri dosiahnutí minima bol podľa zadania určený náhodou.

### Bonus (3D priestor):




*Grafy dosiahnutých výsledkov (čierny bod – začiatok, zelený bod – koniec, modrý bod - posun)*

*Graf č.1 – 3D pohľad, Graf č.2 – pohľad zhora, Graf č.3 – pohľad zdola,*

```
Nájdené minimum: x=55 y=-276 F(x, y)=175.636853,
Nájdené minimum: x=272 y=-453 F(x, y)=-219.364611,
Nájdené minimum: x=269 y=-452 F(x, y)=-219.003936,
Nájdené minimum: x=-176 y=271 F(x, y)=56.936430,
Nájdené minimum: x=569 y=-711 F(x, y)=-771.331564,
Nájdené minimum: x=567 y=566 F(x, y)=-930.005224,
Nájdené minimum: x=54 y=-83 F(x, y)=234.806171,
Nájdené minimum: x=-85 y=-275 F(x, y)=313.532309,
Nájdené minimum: x=54 y=52 F(x, y)=96.736684,
Nájdené minimum: x=569 y=567 F(x, y)=-929.677677,
```

*Dosiahnuté výsledky*

## Záver k bonusu:

- Pri desiatich spusteniach sme dosiahli (skoro) globálne minimum 2-krát (-930,.. a -929,..). Odchýlka vznikla kvôli posunu na grafe „d = 5“. Presun pri dosiahnutí minima bol podľa zadania určený náhodou.

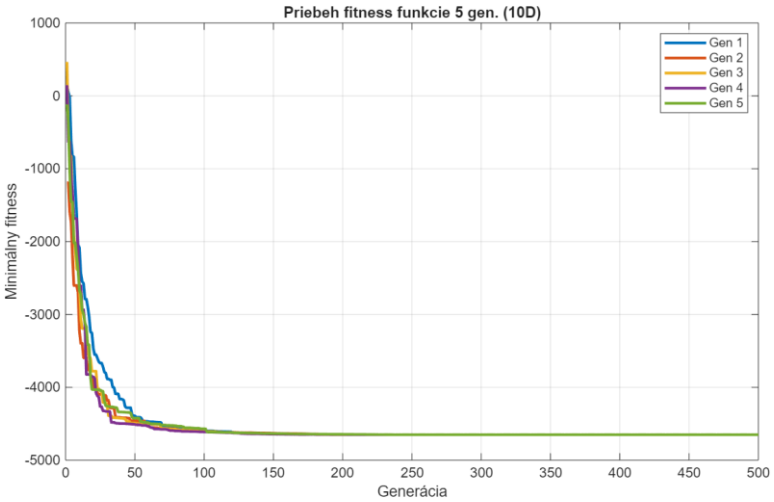
# Úloha č. 2

Použité funkcie:

- genrpop(popsize, Space) – funkcia na generovanie populácie
- testfn3b(Pop) - schwefelova funkcia na nájdenie globálneho minima
- eggholder(Pop) - eggholder funkcia na nájdenie globálneho minima – *pre bonus*
- selbest(Oldpop,Fvpop,Nums) - funkcia na výber jedincov s najwlepšou fitness hodnotou
- seltourn(Oldpop,Fit,n) - funkcia na turnajový výber jedincov
- selsus(Oldpop,Fvpop,n) – funkcia na výber jedincov pomocou váhového ruletového kolesa – *pre bonus*
- crossov(Oldpop,pts,sel) – funkcia na kríženie jedincov
- mutx(Oldpop,factor,Space) – funkcia na globálnu mutáciu populácie
- muta(Oldpop,factor,Amps,Space) – funkcia na lokálnu mutáciu (aditívnu) populácie

**všetky funkcie z toolboxu genetic*

Výsledky:




*Graf dosiahnutých výsledkov - testfn3b(Pop)*

```
Najmenší fitness: -4650.743
Najmenší fitness: -4650.743
Najmenší fitness: -4650.742
Najmenší fitness: -4650.742
Najmenší fitness: -4650.742
```

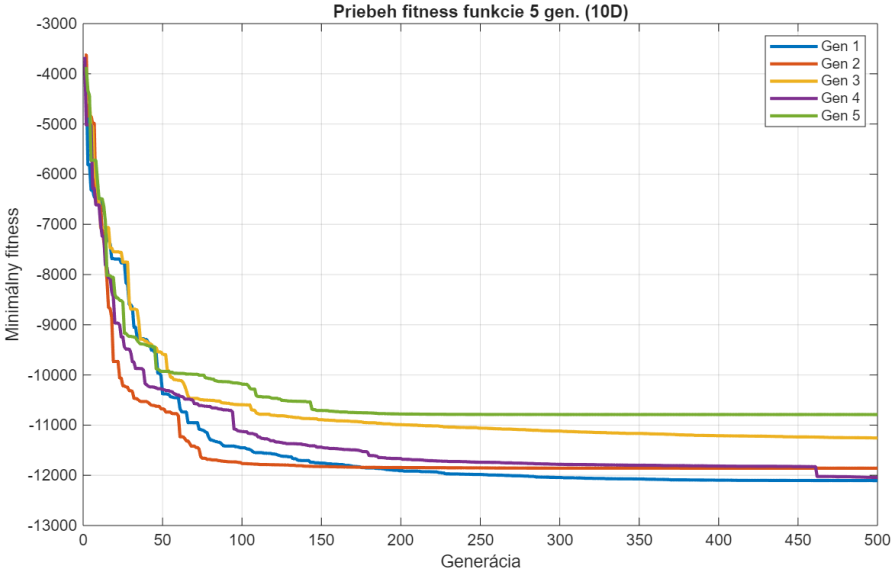
*Dosiahnuté výsledky - testfn3b(Pop)*

**Záver:**

- Pri 5 spusteniach sme dosiahli globálne minimum vždy. Dosiahnuté výsledky boli zabezpečené schwefelovou funkciou, pri populácii 100 s počtom generácii 500 v 10D priestore.
- Populáciu sme vytvorili pomocou funkcie *genrpop*.
- Výber pozostával z dvoch skupín. V prvej sme vybrali pomocou funkcie *selbest* prvých 3 najlepších ([1 1 1]). V druhej sme pomocou funkcie *seltourn* vybrali zbytok populácie (97).
- Následne sme druhú skupinu nechali dvojbodovo krížiť na náhodne vybrané páry pomocou funkcie *crossov*.

- Potom sme na danú skupinu použili mutačné funkcie *mutx* a *muta* s mut. intenzitou 0.1.
- Na záver sme spojili dva výbery dokopy a opakovali tento cyklus 500-krát (podľa generácii).

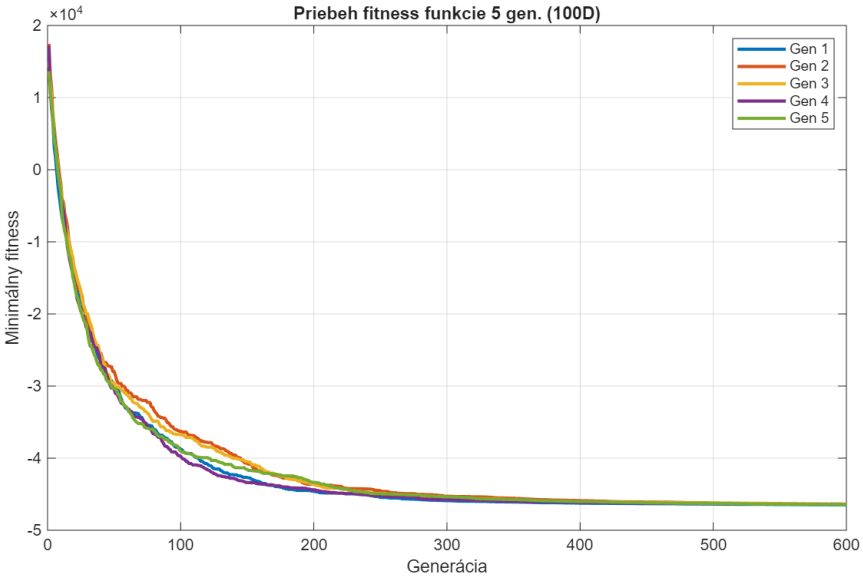
## Bonus (100D + eggholder funkcia):




*Graf dosiahnutých výsledkov - eggholder(Pop)*

```
Najmenší fitness: -12104.928
Najmenší fitness: -11858.272
Najmenší fitness: -11256.090
Najmenší fitness: -12042.128
Najmenší fitness: -10788.912
```

*Dosiahnuté výsledky - eggholder(Pop)*




*Graf dosiahnutých výsledkov - testfn3b(Pop) pri 100D*

```
Najmenší fitness: -46466.381
Najmenší fitness: -46350.234
Najmenší fitness: -46370.840
Najmenší fitness: -46415.259
Najmenší fitness: -46423.337
```

*Dosiahnuté výsledky - testfn3b(Pop) pri 100D*

**Záver:**

- Pri eggholderovej funkcii sme pri rovnakých podmienkach ako pri Schwefelovej funkcii dosiahli najlepší výsledok **-12104,928**. Tento jav je spôsobený komplexnejšou topológiou fitness funkcie, ktorá obsahuje viac lokálnych extrémov.
- Pri Schwefelovej funkcii v 100D priestore sme pri odlišných podmienkach – populácia 1000 a 600 generácií s inak určeným výberom (1. skupina pomocou *selbest* s výberom [9 5 5 3 3] a 2. skupina pomocou *selsus* so zbytkom populácie (975) ) – dosiahli najlepší výsledok **-46466.381**.
- Kríženie (*crossov*) a mutácie (*mutx*, *muta* s intenzitou 0.1) zostali rovnaké ako pri Schwefelovej funkcii.
- Tento výsledok je veľmi blízko k globálnemu minimu, pričom väčšina dosiahnutých hodnôt sa pohybovala maximálne 200 od globálneho minima, čo naznačuje dobrú konvergenciu algoritmu v tomto nastavení.

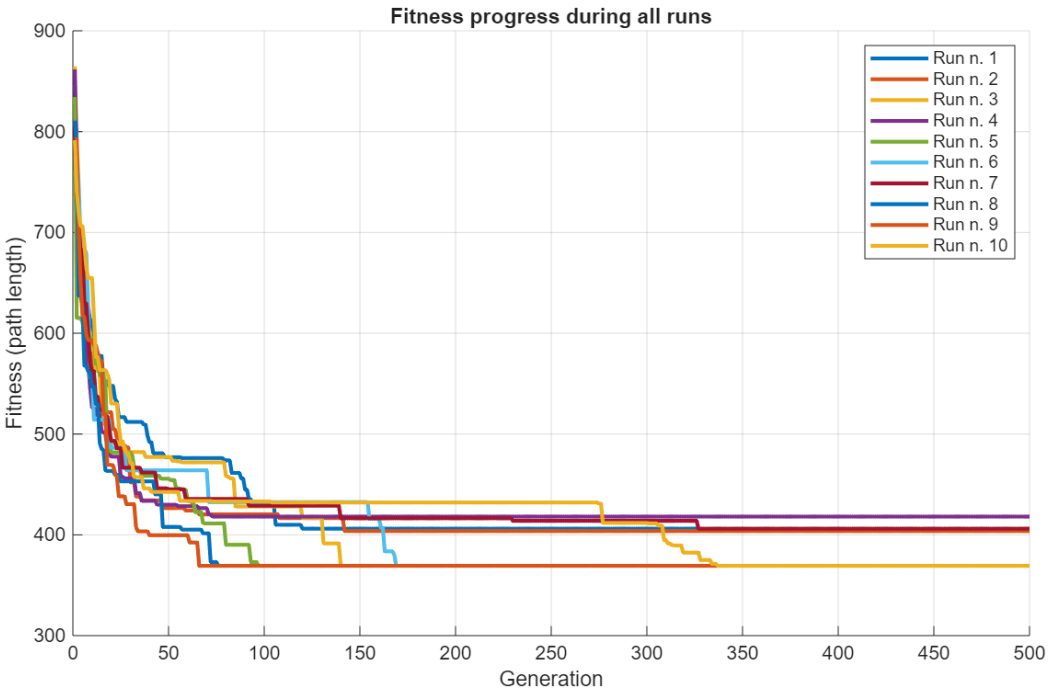
# Úloha č. 3

Použité funkcie:

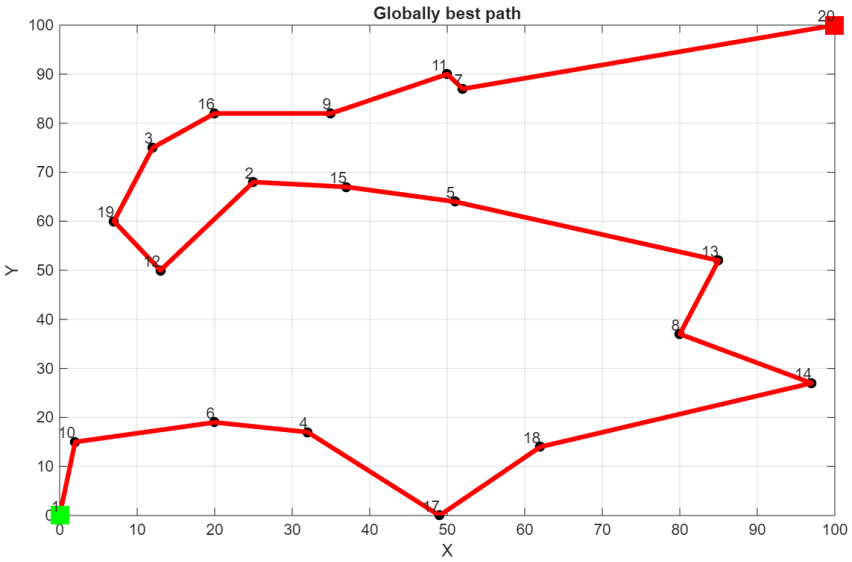
- *selbest(Oldpop,Fvpop,Nums)  - funkcia na výber jedincov s najlepšou fitness hodnotou
- *seltourn(Oldpop,Fit,n) -  funkcia na turnajový výber jedincov
- *crosord(Oldpop,sel) – funkcia na dvojbodové kríženie permutačného typu
- *swapgen(Oldpop,factor) – funkcia na mutáciu poradia génov v náhodne vybraných reťazcoch populácie.
- *swappart(Oldpop,factor) - funckia na mutáciu, ktorá spočíva vo výmene poradia dvoch podreťazcov.
- *invord(Oldpop,rate) – funkcia na vykonanie inverzie poradia génov v náhodne vybraných reťazcoch populácie.
- Fitness(popSize, B) - funkcia na vypočítanie fitness hodnoty pre každého jedinca v populácii na základe vzdialeností medzi bodmi v permutácii.
- genPermPop(popSize, numPoints) – funkcia na generovanie počiatočnej populácie permutácií.

**funkcie z toolboxu genetic*

Výsledky:




*Graf dosiahnutých výsledkov - Fitness(popSize, B)*




*Graf najlepšej cesty - Fitness(popSize, B) - (zelený bod – začiatok, červený bod - koniec)*

```
Run n. 1: Best Fitness = 406.13
Run n. 2: Best Fitness = 403.72
Run n. 3: Best Fitness = 369.24
Run n. 4: Best Fitness = 418.07
Run n. 5: Best Fitness = 369.24
Run n. 6: Best Fitness = 369.24
Run n. 7: Best Fitness = 406.13
Run n. 8: Best Fitness = 369.24
Run n. 9: Best Fitness = 369.24
Run n. 10: Best Fitness = 369.24
Best path from all runs:
[1 10 6 4 17 18 14 8 13 5 15 2 12 19 3 16 9 11 7 20 ]
Length of best run: 369.24 (Iteration: 3)
```

*Dosiahnuté výsledky - Fitness(popSize, B)*

## Záver:

- Po desiatich spusteniach sme najkratšiu cestu z bodu 1 do bodu 20 našli 6-krát. Použili sme populáciu 100 jedincov a 500 generácií s 20 pevne určenými bodmi (B = [0,0; 25,68; 12,75; 32,17; 51,64; 20,19; 52,87; 80,37; 35,82; 2,15; 50,90; 13,50; 85,52; 97,27; 37,67; 20,82; 49,0; 62,14; 7,60; 100,100]).
- Populáciu sme vytvorili pomocou vlastne definovanej funkcie genPermPop, ktorá zabezpečila, že každý jedinec obsahoval 20 bodov, pričom prvý bod bol vždy bod 1 a posledný bod bol bod 20.
- Fitness hodnoty sme počítali pomocou vlastne definovanej funkcie Fitness, ktorá vychádzala z vzdialeností medzi bodmi.
- Výber pozostával z 3 výberov. V prvom výbere sme vybrali top 3 jedincov po dvakrát pomocou funkcie *selbest*. V druhom výbere sme urobili výber najlepšieho jedinca desaťkrát pomocou funkcie *selbest*. V treťom výbere sme vybrali zbytok populácie (84) pomocou funkcie *seltourn*. Následne sme 2. a 3. výber spojili dokopy.
- Na tento spojený výber sme použili kríženie susedných dvojíc v populácii pomocou funkcie *crosord*
- V rámci tohto výberu sme aplikovali mutácie, pri ktorých prvý a posledný prvok zostali vždy nezmenené. Konkrétne sme použili: výmenu poradia dvoch substrings v reťazcoch pomocou swappart s faktorom 0.1, inverziu poradia podreťazcov pomocou invord s faktorom 0.3 a výmenu poradia génov pomocou swapgen s faktorom 0.1.
- Na záver sme spojili pôvodný 1. výber a vytvorený 2. výber a cyklus opakovali podľa počtu generácií. (500)

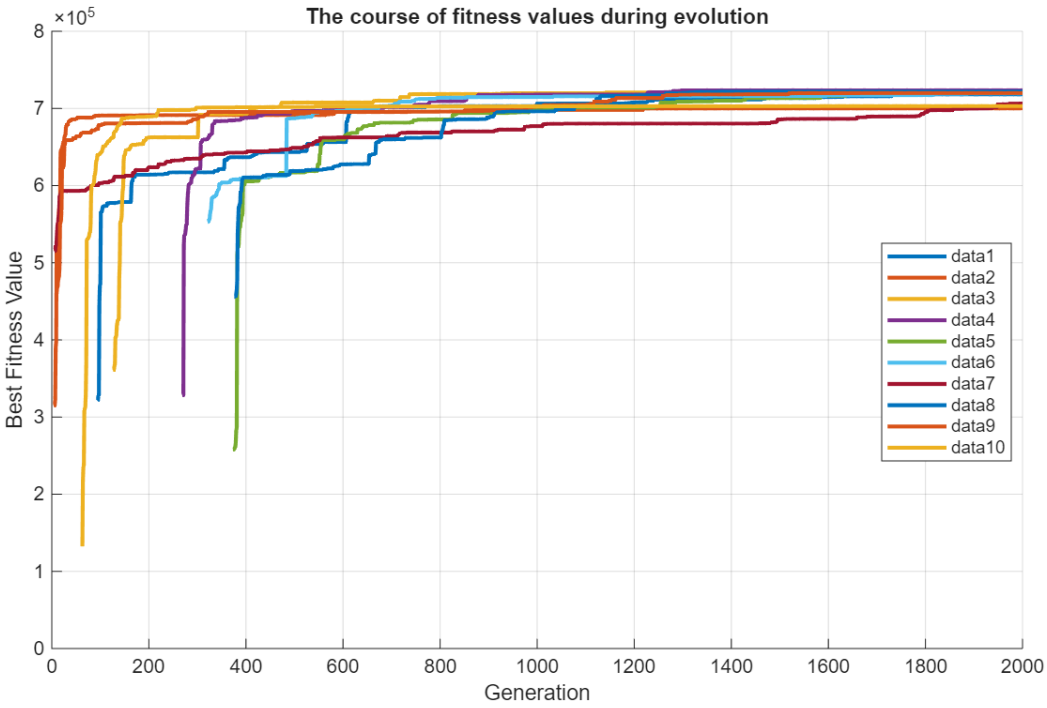
# Úloha č. 4

Použité funkcie:

- *genrpop(popsize, Space) – funkcia na generovanie populácie
- *selbest(Oldpop,Fvpop,Nums) - funkcia na výber jedincov s najwlepšou fitness hodnotou
- *seltourn(Oldpop,Fit,n) - funkcia na turnajový výber jedincov
- *crossov(Oldpop,pts,sel) – funkcia na kríženie jedincov
- *mutx(Oldpop,factor,Space) – funkcia na globálnu mutáciu populácie
- *muta(Oldpop,factor,Amps,Space) – funkcia na lokálnu mutáciu (aditívnu) populácie
- Fitness1(Pop, Limit, J_function) – fitness funkcia s mŕtvou pokutou
- Fitness2(Pop, Limit, J_function) - fitness funkcia so stupňovou pokutou
- Fitness3(Pop, Limit, J_function) - fitness funkcia s úmernou pokutou

**funkcie z toolboxu genetic*

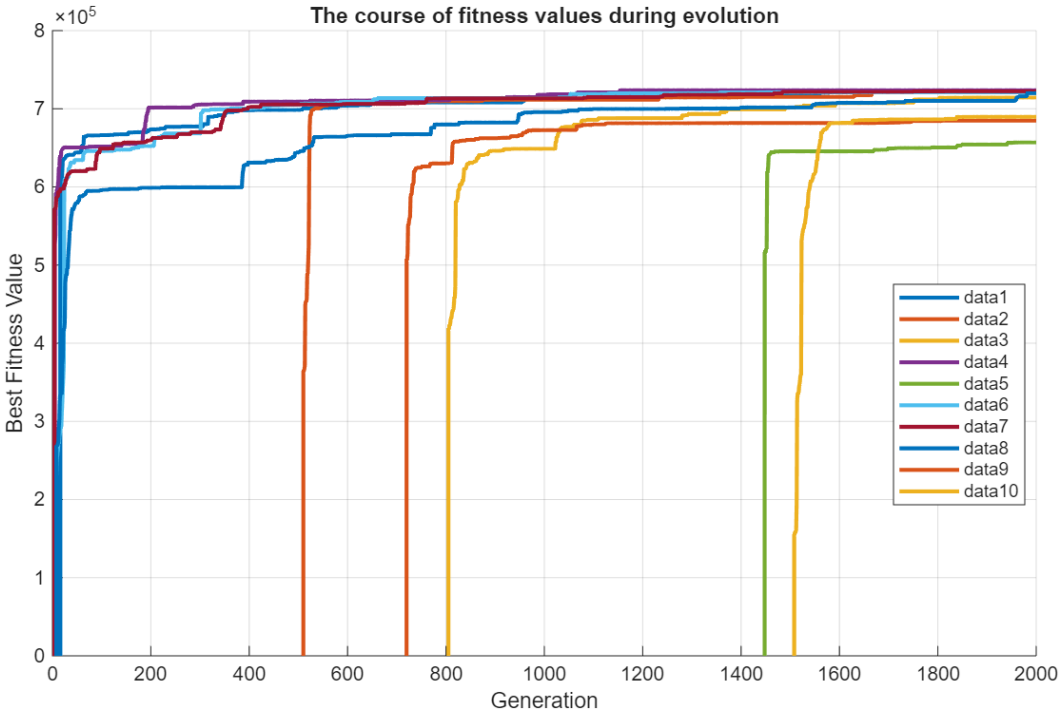
Výsledky:




*Graf dosiahnutých výsledkov - Fitness1(Pop, Limit, J_function) – mŕtva pokuta*

```
Iteration 1: Best Fitness = 718030.72
Investment distribution: [9823.10 2486375.54 2368796.55 2627633.94 2507317.06]

Iteration 2: Best Fitness = 699878.00
Investment distribution: [770050.77 1716958.89 2465120.56 2533379.88 2514455.95]

Iteration 3: Best Fitness = 722272.87
Investment distribution: [11.58 2472632.13 2457256.88 2540523.91 2529169.25]

Iteration 4: Best Fitness = 723590.68
Investment distribution: [0.00 2491705.60 2475715.86 2521739.91 2510762.95]

Iteration 5: Best Fitness = 719780.93
Investment distribution: [0.00 2463778.88 2412166.25 2578866.09 2544923.13]

Iteration 6: Best Fitness = 719264.26
Investment distribution: [9225.31 2483129.48 2402594.21 2555477.88 2549243.00]

Iteration 7: Best Fitness = 706242.93
Investment distribution: [0.00 2300932.96 2204960.73 2793223.54 2700770.61]

Iteration 8: Best Fitness = 721712.48
Investment distribution: [0.00 2498717.22 2435634.99 2560473.45 2505080.36]

Iteration 9: Best Fitness = 720013.18
Investment distribution: [0.00 2477980.37 2411353.07 2577851.52 2532692.50]

Iteration 10: Best Fitness = 703191.60
Investment distribution: [651632.35 1848157.13 2499132.15 2480480.36 2480438.96]


Overall Best Fitness: 723590.68 (iteration 4)
Best Investment Distribution: [0.00 2491705.60 2475715.86 2521739.91 2510762.95]
```

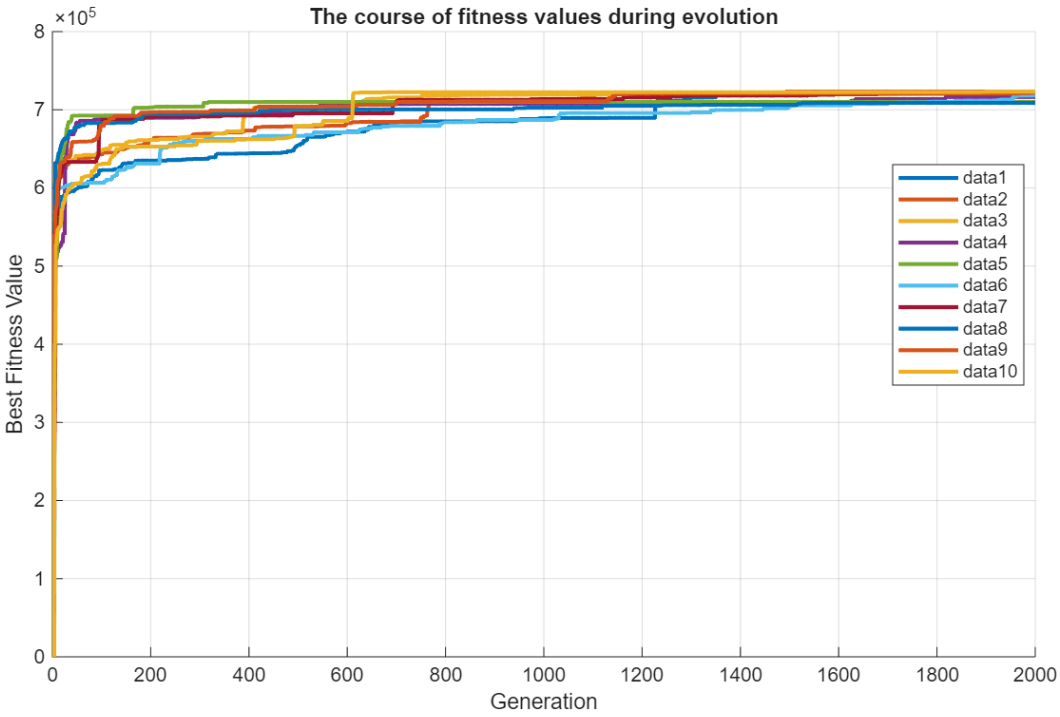
*Dosiahnuté výsledky - Fitness1(Pop, Limit, J_function) – mŕtva pokuta*




*Graf dosiahnutých výsledkov – Fitness2(Pop, Limit, J_function)* – stupňová pokuta

```
Iteration 1: Best Fitness = 721188.01
Investment distribution: [856.69 2483043.03 2434650.82 2545904.61 2535497.19]

Iteration 2: Best Fitness = 721830.60
Investment distribution: [0.00 2491821.94 2442060.52 2548143.16 2517756.38]

Iteration 3: Best Fitness = 714311.19
Investment distribution: [210483.93 2267998.56 2422221.40 2573885.79 2525088.68]

Iteration 4: Best Fitness = 723732.63
Investment distribution: [348.85 2498453.55 2476994.24 2514759.39 2509440.03]

Iteration 5: Best Fitness = 657037.05
Investment distribution: [1756856.91  666865.15 2226369.27 2769799.16 2579872.83]

Iteration 6: Best Fitness = 721465.81
Investment distribution: [0.00 2490904.84 2434245.31 2561907.40 2512420.81]

Iteration 7: Best Fitness = 722424.76
Investment distribution: [543.28 2498570.38 2451824.69 2535570.70 2513362.99]

Iteration 8: Best Fitness = 720006.28
Investment distribution: [0.00 2485706.20 2408548.52 2587344.93 2516516.31]

Iteration 9: Best Fitness = 684849.15
Investment distribution: [1269117.61 1230685.47 2459941.30 2533617.19 2506517.83]

Iteration 10: Best Fitness = 689713.93
Investment distribution: [895929.70 1574181.72 2344457.22 2653814.90 2531296.77]


Overall Best Fitness: 723732.63 (iteration 4)
Best Investment Distribution: [348.85 2498453.55 2476994.24 2514759.39 2509440.03]
```

*Dosiahnuté výsledky – Fitness2(Pop, Limit, J_function)* – stupňová pokuta




*Graf dosiahnutých výsledkov – Fitness3(Pop, Limit, J_function)* – úmerná pokuta

```
Iteration 1: Best Fitness = 721242.80
Investment distribution: [578.11 2496590.61 2429240.18 2556293.18 2517286.36]

Iteration 2: Best Fitness = 723139.37
Investment distribution: [1889.00 2493104.25 2467534.04 2527579.70 2509259.81]

Iteration 3: Best Fitness = 720120.33
Investment distribution: [282.45 2469236.62 2415446.52 2582335.34 2532464.70]

Iteration 4: Best Fitness = 716362.40
Investment distribution: [275983.68 2222098.35 2493793.11 2506041.89 2501928.25]

Iteration 5: Best Fitness = 710274.91
Investment distribution: [438082.30 2061815.14 2499172.20 2486552.34 2486449.55]

Iteration 6: Best Fitness = 718101.16
Investment distribution: [2856.11 2496937.73 2366569.48 2623434.08 2509451.83]

Iteration 7: Best Fitness = 722013.09
Investment distribution: [82852.65 2412081.87 2492375.31 2506304.09 2506274.53]

Iteration 8: Best Fitness = 708826.12
Investment distribution: [464259.64 2021314.20 2461496.62 2536087.67 2516677.07]

Iteration 9: Best Fitness = 721587.92
Investment distribution: [138.92 2467504.86 2451669.24 2540629.04 2534713.36]

Iteration 10: Best Fitness = 722954.26
Investment distribution: [0.00 2486714.69 2464975.92 2533790.92 2514535.83]


Overall Best Fitness: 723139.37 (iteration 2)
Best Investment Distribution: [1889.00 2493104.25 2467534.04 2527579.70 2509259.81]
```

*Dosiahnuté výsledky – Fitness3(Pop, Limit, J_function)* – úmerná pokuta

## Záver:

- Všetky testovania boli vykonané za podmienok: veľkosť populácie 100, počet generácii 2000 počet behov 10, každý jedinec sa skladá za 5 génov, úvodná inicializácia je pre každý gén od 0 do 10000000.
- Generovanie populácie pomocou funkcie *genrpop*.
- Fitness jedincov sa hodnotilo **tromi rôznymi funkciami**, ktoré sa líšili spôsobom výpočtu pokuty pri nesplnení podmienok. Pokuty boli stanovené podľa zadania.
- Výber pozostával z 3 výberov. V prvom výbere sme vybrali top 2 jedincov po dvakrát pomocou funkcie *selbest.* V druhom výbere sme urobili výber top 3 jedincov po dvakrát pomocou funkcie *selbest*. V treťom výbere sme vybrali zbytok populácie (90) pomocou funkcie *seltourn.* Následne sme 2. a 3. výber krížili pomocou funkcie crossov.
- Na 2. výber sme použili lokálnu mutáciu *muta* s ratom 0.3 a na 3. výber sme použili globálnu mutáciu *mutx* s ratom 0.2.
- Následne sme spojili všetky tri skupiny dokopy a cyklus sa opakoval počtom generácii (2000).
- Podľa grafov a výsledkov môžeme vidieť, že najstabilnejšie výsledky vychádzali pri použití funkcie *Fitness3,* naopak najmenej stabilné pri použití funkcie *Fitness2.* (Avšak pri viacerých porovnaniach sa výsledky líšili inak a niekedy najmenej stabilné vyšli pri použití funkcie *Fitness1*).