**Dokumentácia k úlohám 1 - 4**

*Jozef Barčák, API\_2*

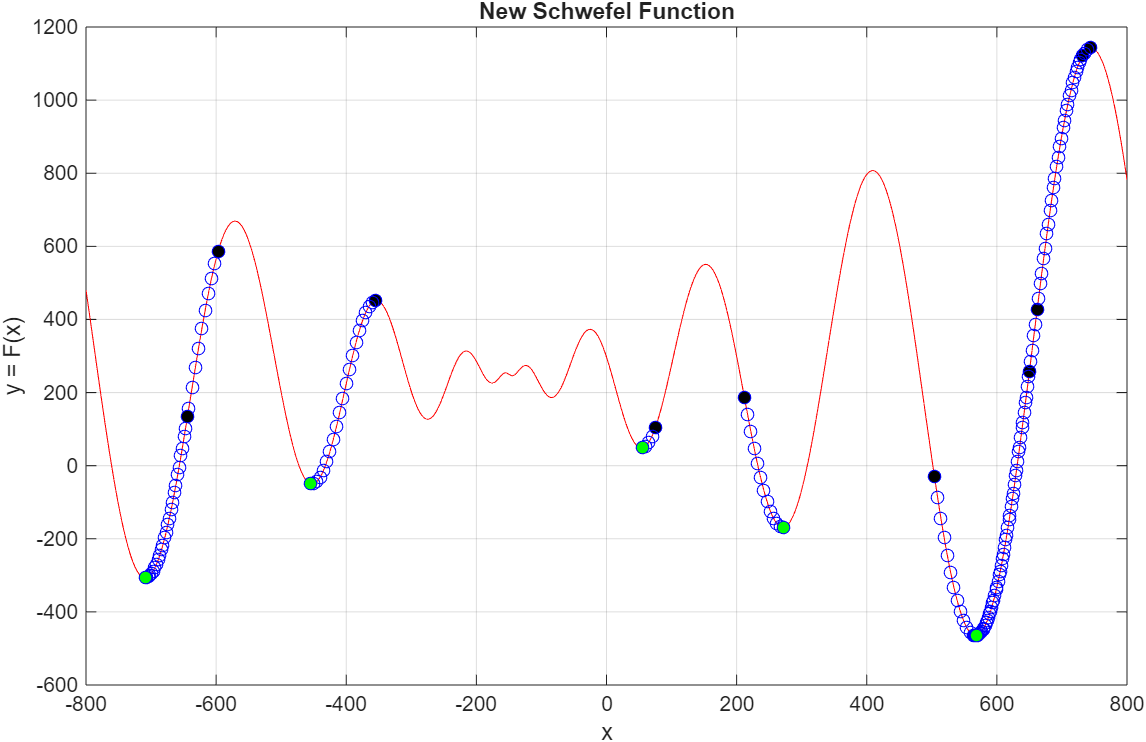
**Úloha č. 1**

Použité funkcie:

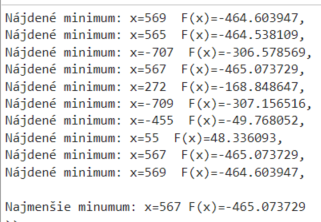
* testfn3b(Pop) - schwefelova funkcia na nájdenie globálneho minima

*\*funkcia z toolboxu genetic*

Výsledky:

****

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

**

*Dosiahnuté výsledky*

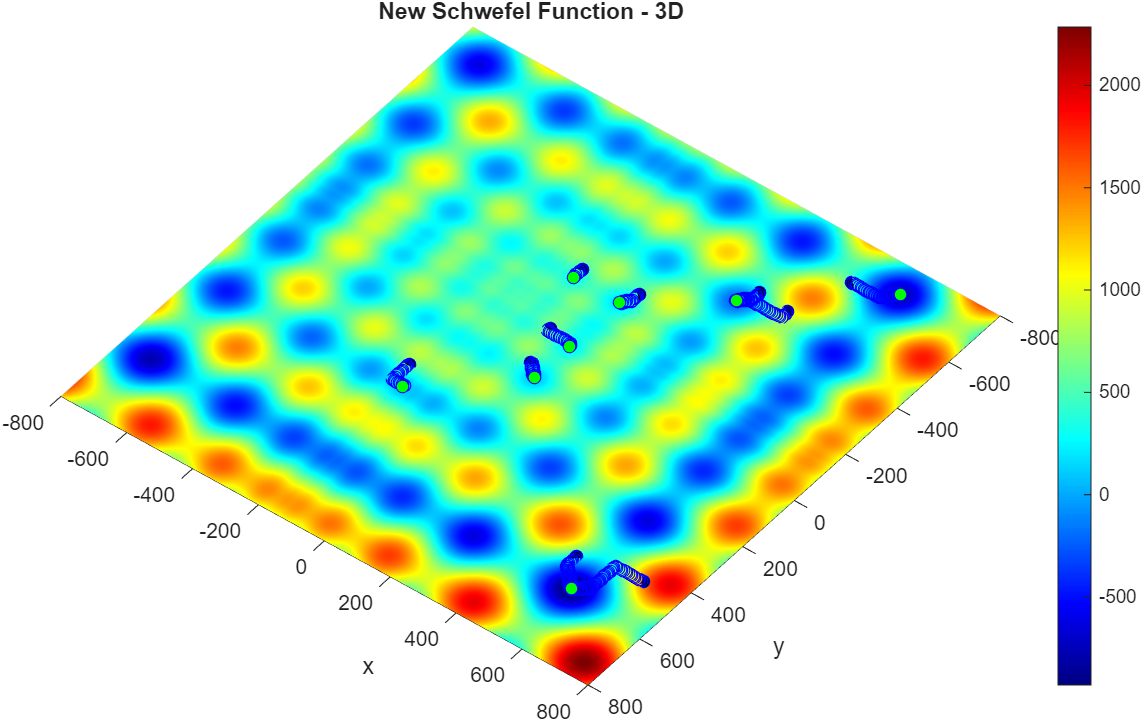
**Záver:**

* 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):**

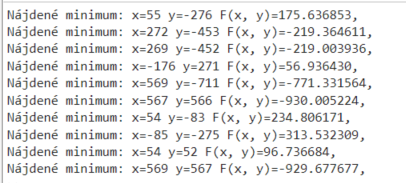
**Obrázok, na ktorom je diagram, text, snímka obrazovky, vývoj

Obsah vygenerovaný umelou inteligenciou môže byť nesprávny. Obrázok, na ktorom je snímka obrazovky, pestrofarebnosť, diagram, rad

Obsah vygenerovaný umelou inteligenciou môže byť nesprávny.**

*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,*

**

*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.

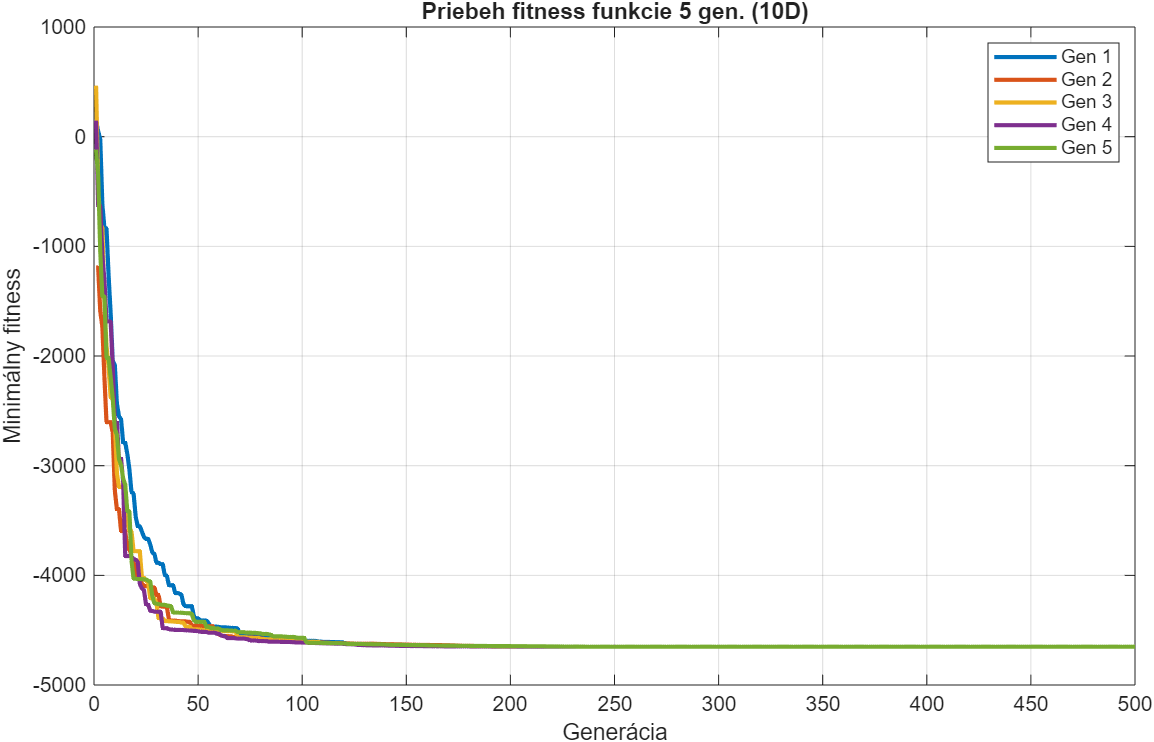
**Úloha č. 2**

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)*

Obrázok, na ktorom je text, písmo, biely

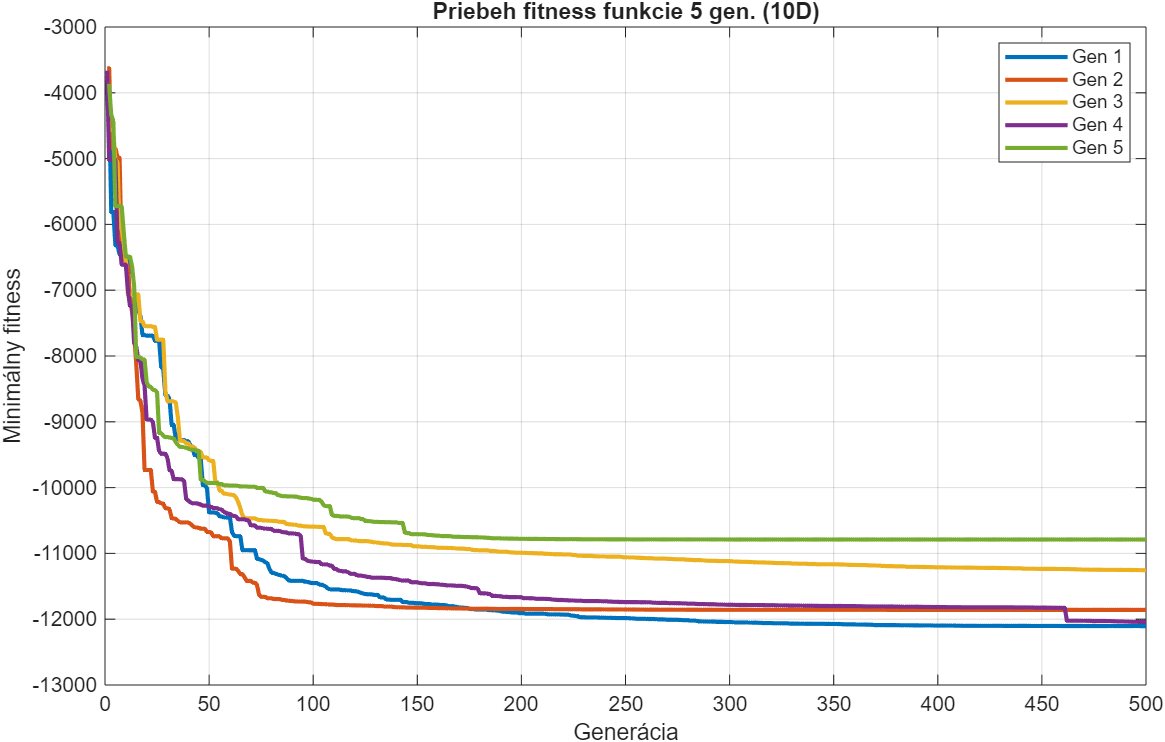
Obsah vygenerovaný umelou inteligenciou môže byť nesprávny.

*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):**

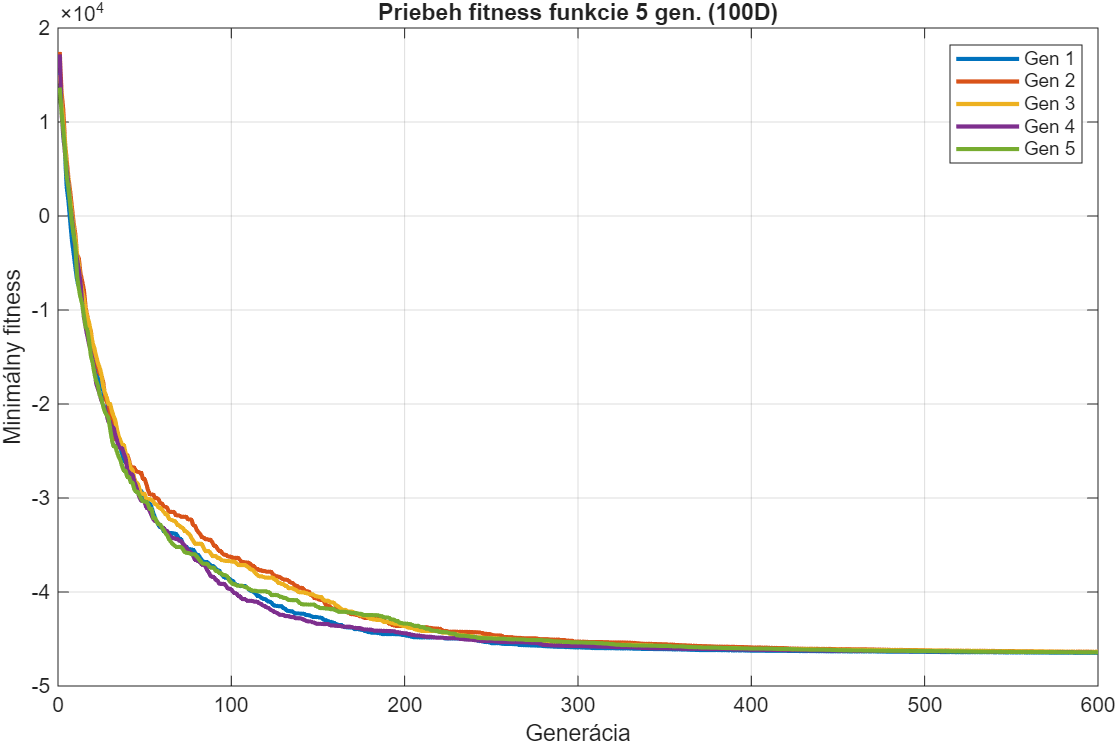
****

*Graf dosiahnutých výsledkov - eggholder(Pop)*

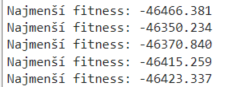
**Obrázok, na ktorom je text, písmo, biely, algebra

Obsah vygenerovaný umelou inteligenciou môže byť nesprávny.**

*Dosiahnuté výsledky - eggholder(Pop)*



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



*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.

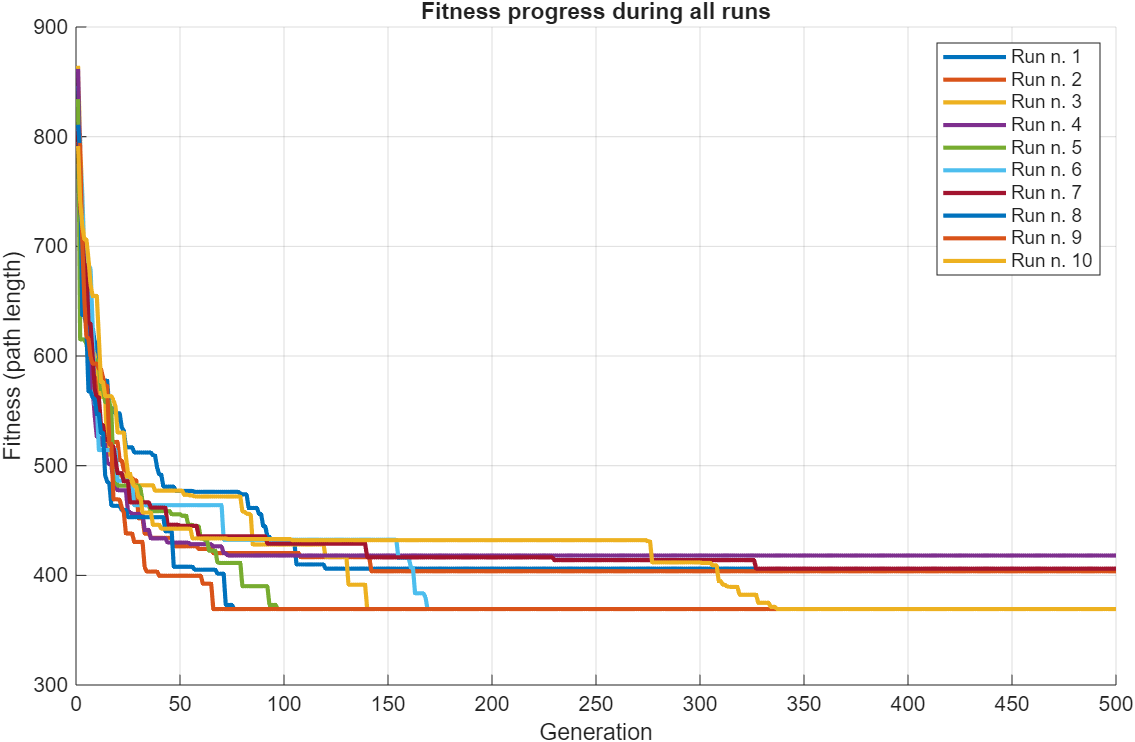
**Úloha č. 3**

Použité funkcie:

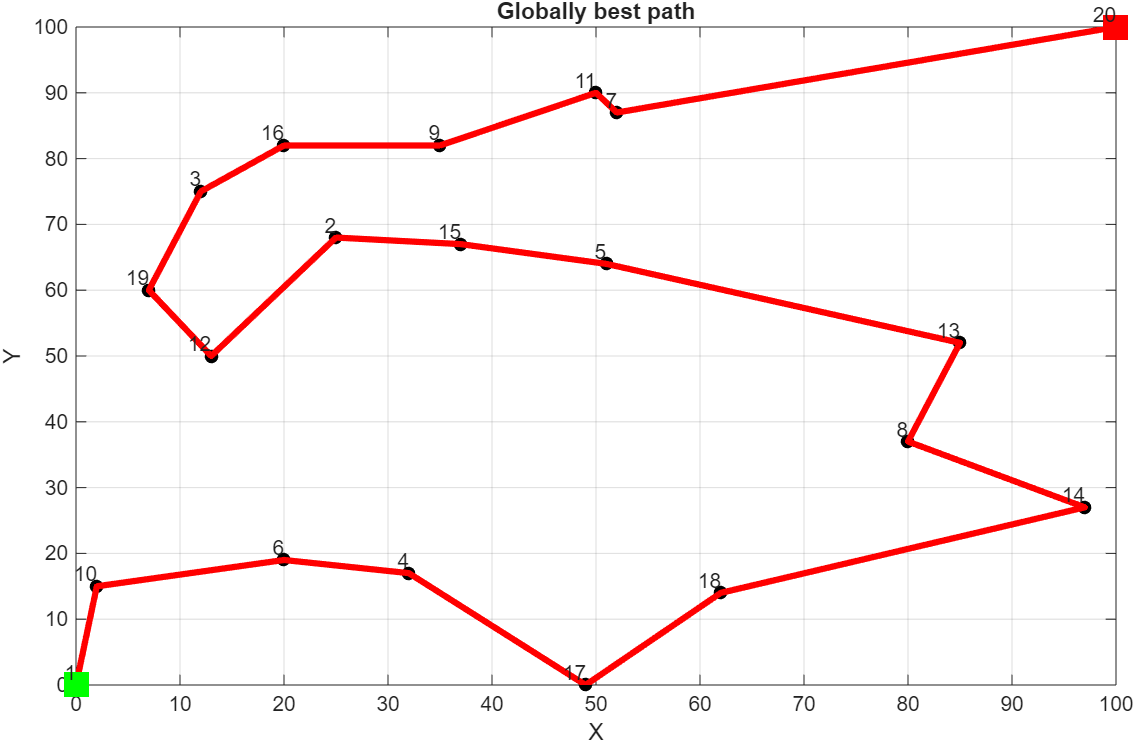
* \*selbest(Oldpop,Fvpop,Nums) - funkcia na výber jedincov s najlepšou fitness hodnotou
* \*seltourn(Oldpop,Fit,n) - funkcia na turnajový výber jedincov
* \*crosord(Oldpop,sel) – funkcia na dvojbodové kríženie permutačného typu
* \*swapgen(Oldpop,factor) – funkcia na mutáciu poradia génov v náhodne vybraných 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)*

Obrázok, na ktorom je text, písmo, snímka obrazovky, biely

Obsah vygenerovaný umelou inteligenciou môže byť nesprávny.

*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)

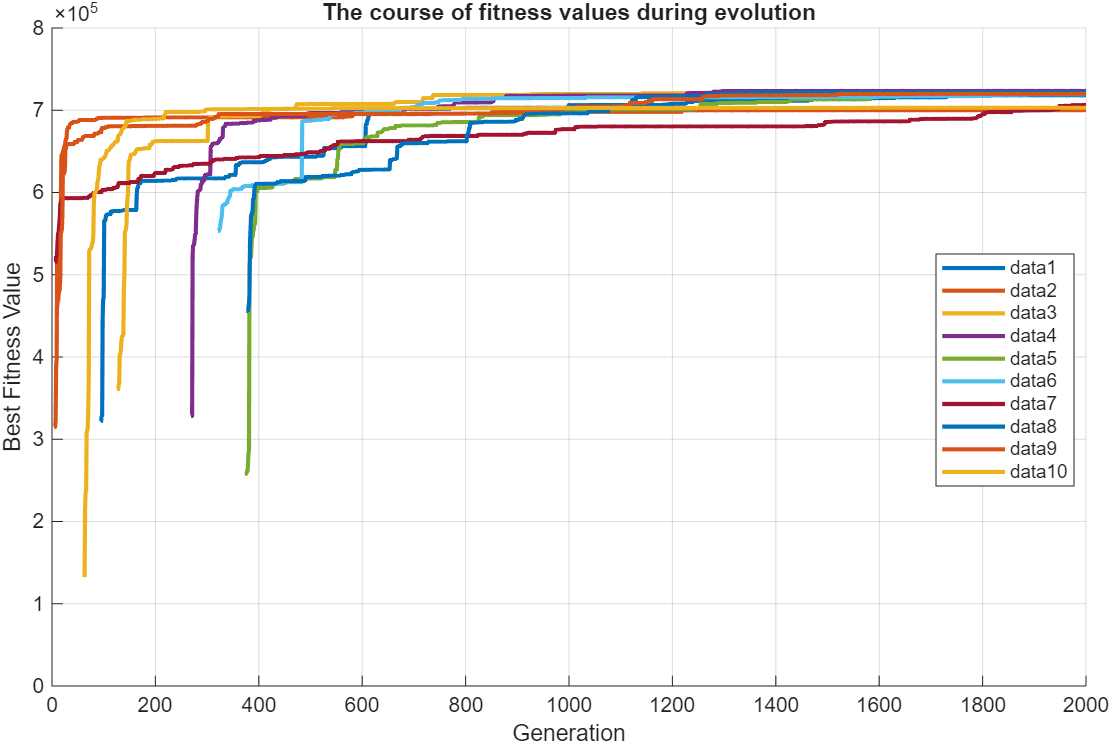
**Úloha č. 4**

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:

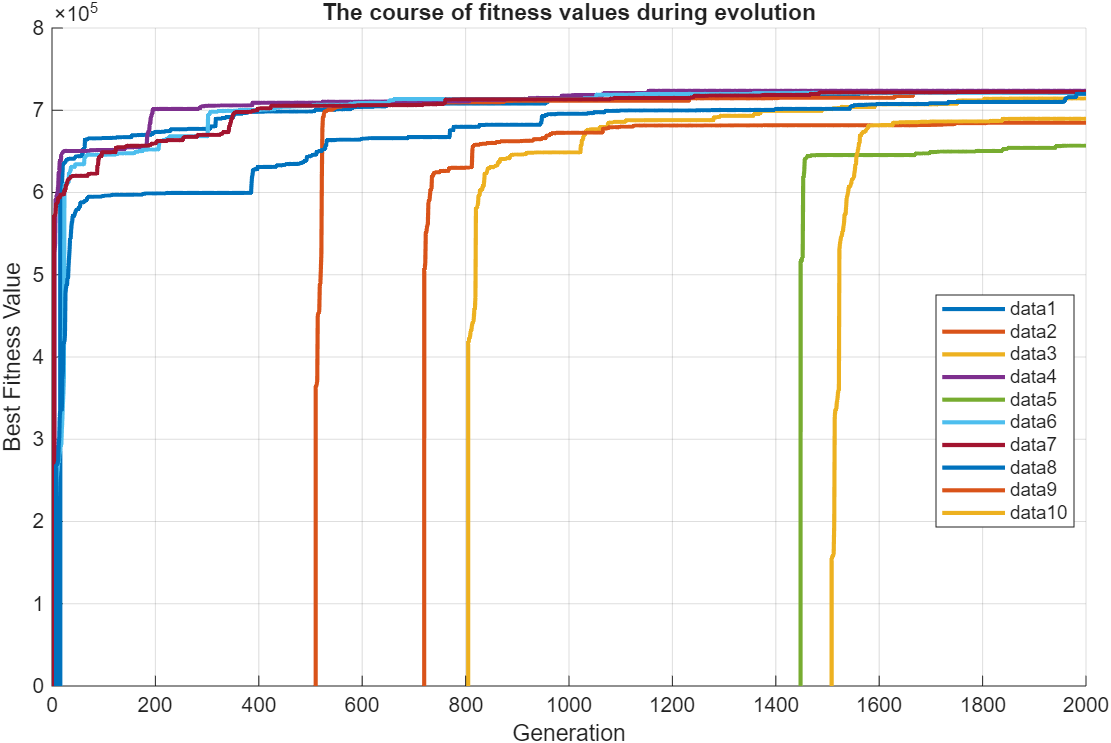


*Graf dosiahnutých výsledkov - Fitness1(Pop, Limit, J\_function) – mŕtva pokuta*

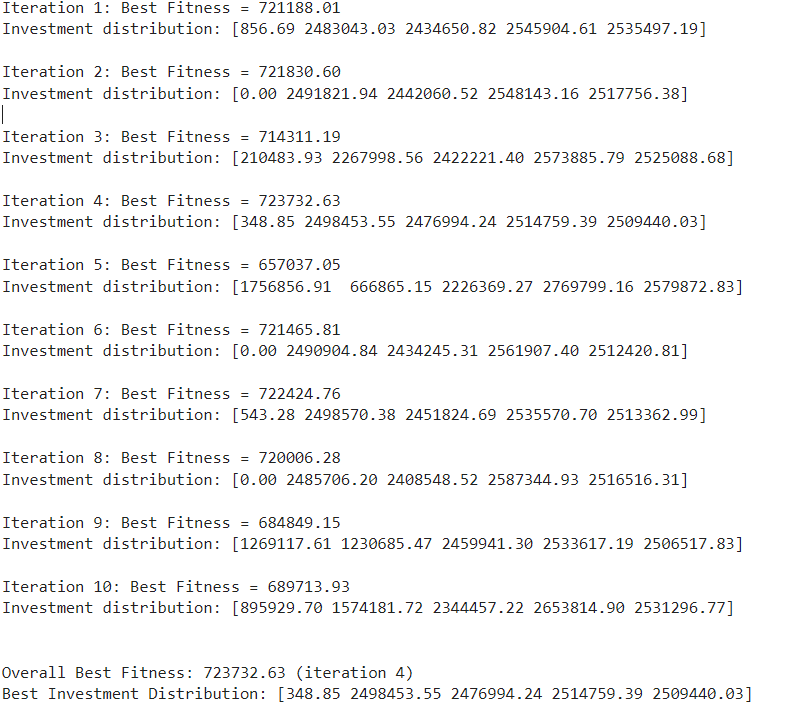
Obrázok, na ktorom je text, písmo, snímka obrazovky, číslo

Obsah vygenerovaný umelou inteligenciou môže byť nesprávny.

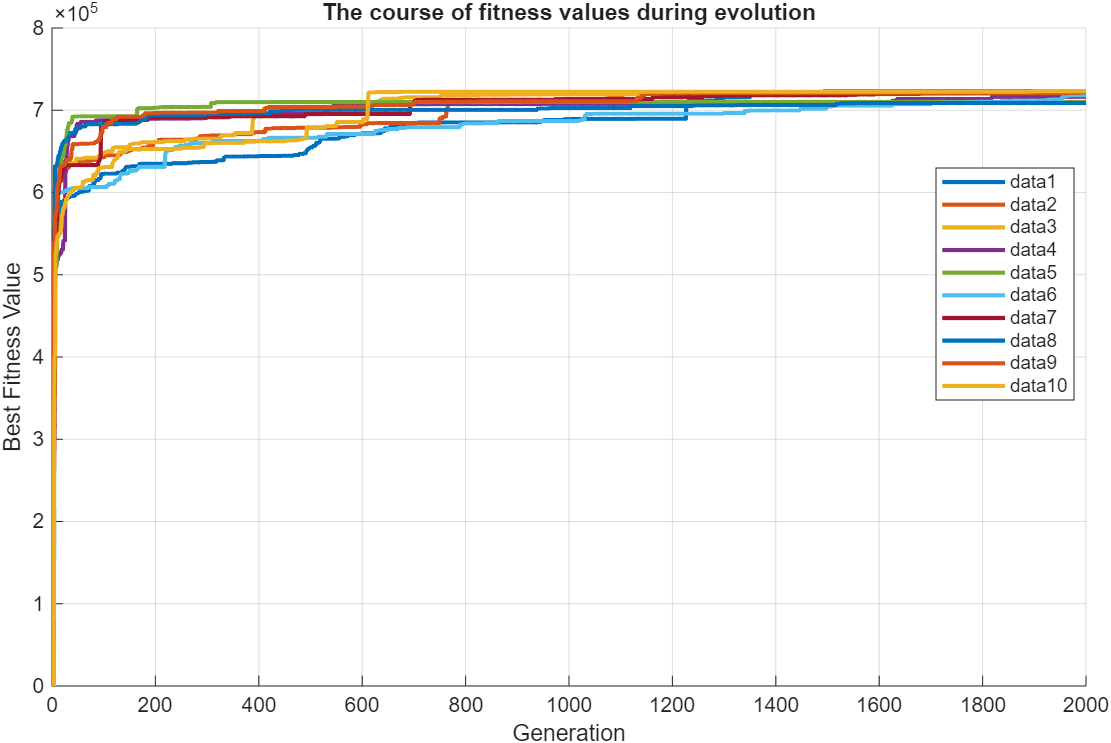
*Dosiahnuté výsledky - Fitness1(Pop, Limit, J\_function) – mŕtva pokuta*

**

*Graf dosiahnutých výsledkov – Fitness2(Pop, Limit, J\_function) –* stupňová pokuta

**

*Dosiahnuté výsledky – Fitness2(Pop, Limit, J\_function) –* stupňová pokuta

**

*Graf dosiahnutých výsledkov – Fitness3(Pop, Limit, J\_function) –* úmerná pokuta

*Obrázok, na ktorom je text, snímka obrazovky, písmo, číslo

Obsah vygenerovaný umelou inteligenciou môže byť nesprávny.*

*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*).