Artem Delikatnyi
129094

Úloha 1b

Spustenie programu

Program je možné spustiť v niekoľkých režimoch:

- **by default** - Spustí program s počtom testov 10 a metódou výberu rulety
- **--mode t** - Spustenie programu metódou výberu turnaja
- **--mode r** - Spustí program s výberovou metódou rulety
- **--test 100** - Spustí program s počtom testov 100

Príklady správneho spustenia programu:

- **python <file_name>** - by default
- **python <file_name> --mode t** - Spustenie programu metódou výberu turnaja
- **python <file_name> --mode r --test 100** - Spustí program s počtom testov 100 a metódou výberu rulety

V predvolenom nastavení program negeneruje grafy, ak chcete spustiť program v režime generovania grafov, musíte nastaviť príslušný príznak, ale je dôležité si uvedomiť, že čas behu programu bude potom oveľa vyšší.

```
CREATE_PLOT = False
SELECTION_MODE = 1  # 1 -> Roulette / 0 -> Tournament
```

Pred spustením programu sa tiež uistite, že sú nainštalované všetky knižnice. Ďalšie použité knižnice boli **matplotlib.**

Popis algoritmu

Genetický algoritmus implementovaný v tomto projekte je určený na riešenie problému nájdenia všetkých pokladov na mape s obmedzeným počtom inštrukcií (maximálne 64 inštrukcií a maximálny počet vykonaných inštrukcií nesmie prekročiť 500). Agent (lovec pokladov) sa pohybuje po mriežke mapy s cieľom nazbierať čo najviac pokladov a algoritmus sa musí v priebehu času vyvíjať tak, aby maximalizoval efektivitu úlohy a našiel čo najviac pokladov.

Všeobecný priebeh algoritmu

Inicializácia počiatočnej populácie: Každý agent je reprezentovaný ako zoznam 64 génov (genóm). Každý gén je reťazec 8 bitov predstavujúci špecifickú inštrukciu.

1. **Fitness:** Funkcia fitness hodnotí úspešnosť agenta na základe počtu nájdených pokladov a počtu vykonaných inštrukcií. Za každý nájdený poklad sa hodnota fitness *zvyšuje o 1000*, na konci sa hodnota fitness zvyšuje o *1000 - počet vykonaných inštrukcií*.
2. **Elitarstvo (Elitism):** počet genómov, ktoré sa odovzdávajú ďalšej generácii v nezmenenej podobe.

3. **Výber (Selection):** Na základe hodnôt fitness sa vyberú agenti, ktorí vytvoria novú populáciu. Používa sa metóda výberu pomocou rulety (čím vyššia je adaptabilita jedinca, tým vyššia je pravdepodobnosť jeho výberu).
4. **Kríženie (Crossover):** Operácia, pri ktorej sa mieša genetický materiál dvoch rodičov s cieľom vytvoriť nových agentov. Používa sa jednobodový kríženie - vyberie sa náhodný bod, v ktorom si rodičovské genómy vymenia svoje prvky.
5. **Mutácia (Mutation)**: Nízka pravdepodobnosť náhodných zmien v genóme agensa, čo podporuje vnášanie diverzity do populácie. V mojom prípade vedie mutácia k úplnej zmene jednej alebo viacerých inštrukcií.

Nastavenia parametrov

Počas behu algoritmu boli použité nasledujúce nastavenia:

- **Generation Size** (počet generácií) - 1000
- **Population Size** (počet genómov v jednej generácii) - 100
- **Mutation Rate** (šanca na mutáciu) - 10 %
- **Crossover Rate** (šanca na kríženie) - 95 %.
- **Elitism** (počet genómov, ktoré prechádzajú nezmenené do ďalšej generácie) - 5.

Tieto hodnoty boli zvolené tak, aby sa maximalizovala úspešnosť algoritmu.

Gény

Každý gén je 8-bitový reťazec, pričom prvé dva bity označujú typ inštrukcie (čo sa bude vykonávať):

- **„00“:** zvýši hodnotu v pamäti.
- **„01“:** zníženie hodnoty v pamäti.
- **„10“:** skok na inú adresu pamäte.
- **„11“:** posun agenta na mape (hore, dole, vľavo, vpravo) v závislosti od počtu jednotlivých bitov v géne.

Genóm každého agenta pozostáva zo 64 takýchto inštrukcií, ktoré agent vykonáva, kým nevyzbiera všetky poklady alebo nevyjde mimo mapy, alebo nevykoná maximálny počet inštrukcií.

Pravidlá pre pohyb agenta na mape

Ak sú prvé dva bity inštrukcie „11“, znamená to, že pozícia agenta na mape sa zmení v závislosti od počtu „1“ v bajte.

- Ak je počet „1“ v bajte >= 7 - hľadajúci sa posunie DOLE.
- Ak je počet „1“ v bajte >= 5 a < 7 - hľadač sa posunie DOPRAVA.
- Ak je počet „1“ v bajte >= 3 a < 5 - hľadač sa pohybuje VĽAVO.
- Ak je počet „1“ v bajte < 3 - hľadač sa pohybuje HORE.

Inicializácia prvej populácie

Počiatočná populácia pozostáva z náhodných agentov, z ktorých každý má náhodne vygenerovaný genóm (64 náhodných 8-bitových reťazcov). Inicializácia sa uskutočňuje prostredníctvom funkcie **init_population()**, ktorá vytvorí zoznam 64 náhodných inštrukcií pre každého agenta. Každá inštrukcia je náhodná sada 8 bitov.

Artem Delikatnyi
129094

Vytvorenie novej populácie

Každá generácia prechádza nasledujúcimi fázami:

- **Elitarstvo:** Najlepší agenti aktuálnej generácie (v tomto prípade 5%) sa skopírujú v nezmenenej podobe do ďalšej generácie.
- **Výber:** Používa sa metóda rulety, pri ktorej je šanca, že agent bude vybraný, úmerná jeho hodnote fitness. Predvolená metóda výberu je ruleta, ale program je možné spustiť aj s metódou výberu Turnaj.
- **Kríženie:** S určitou pravdepodobnosťou (v tomto projekte **95%**) sa medzi dvoma agentmi vymení genetický materiál.
- **Mutácia:** Každý gén agenta môže náhodne zmutovať s pravdepodobnosťou **10%**.

Hodnotenie úspešnosti algoritmu

Na vyhodnotenie úspešnosti programu sa uskutočnili testy. Čo sú to testy?

Spustenie algoritmu určitý počtekrát v slučke, aby sa pochopilo percent úspešnosti vykonávania programu. Napríklad algoritmus sa spustí 10-krát. Pri prvom spustení algoritmu sa získa výsledok, potom sa všetky údaje vynulujú a algoritmus sa spustí znova a hľadá sa najlepší výsledok.

Najväčší počet testov sa vykonal na pôvodnej mape (mape uvedenej v podmienke úlohy), ale uskutočnili sa aj testy pre vlastné mapy so zmenenými východiskovými bodmi a so zmenenými polohami pokladov.

Algoritmus vykazuje postupné zlepšovanie výsledkov s rastúcim počtom generácií. Adaptabilita agentov vyjadrená počtom zozbieraných pokladov a minimálnym počtom vykonaných inštrukcií sa s vývojom populácie zvyšuje.

Ako testy slúžilo viacnásobné spustenie algoritmu v cykle 10 / 100 / 1000 krát.

Príklad spustenia **1000 testov** s výstupnou úspešnosťou **98.7 %** - inými slovami, vo viac ako 980 prípadoch sa algoritmu podarilo nájsť všetkých 5 pokladov, v ostatných prípadoch algoritmus našiel 4/5 pokladov

```
Wave 992: Generation 50  |    The amount of treasure found = 5 |    The number of completed instructions = 227 |    The number of steps = 62
Wave 993: Generation 351 |    The amount of treasure found = 5 |    The number of completed instructions = 135 |    The number of steps = 46
Wave 994: Generation 49  |    The amount of treasure found = 5 |    The number of completed instructions = 163 |    The number of steps = 50
Wave 995: Generation 299 |    The amount of treasure found = 5 |    The number of completed instructions = 86  |    The number of steps = 50
Wave 996: Generation 50  |    The amount of treasure found = 5 |    The number of completed instructions = 236 |    The number of steps = 88
Wave 997: Generation 274 |    The amount of treasure found = 5 |    The number of completed instructions = 244 |    The number of steps = 104
Wave 998: Generation 517 |    The amount of treasure found = 5 |    The number of completed instructions = 44  |    The number of steps = 22
Wave 999: Generation 64  |    The amount of treasure found = 5 |    The number of completed instructions = 293 |    The number of steps = 42
Wave 1000: Generation 162|    The amount of treasure found = 5 |    The number of completed instructions = 190 |    The number of steps = 92
Percent of Success: 98.7%
```

Výsledok testov - šanca na úspešné splnenie úlohy (nájdenie všetkých 5 pokladov) je približne 96-99 %. (Závisí od počtu testov - čím vyšší počet, tým presnejšie bude toto číslo)

Porovnanie metódy výberu Rulety a Turnaja

Roulette Wheel Selection: Pri metóde rulety je pravdepodobnosť výberu jedinca úmerná jeho fitness. Proces výberu sa podobá rulete, kde každý sektor zodpovedá jednému jedincovi a veľkosť sektora závisí od hodnoty fitness: čím vyššia je fitness, tým väčší je sektor. Algoritmus roztočí ruletu a náhodne vyberie jedinca v závislosti od toho, do ktorého sektora padne „ukazovateľ“.

Implementation:

Artem Delikatnyi
129094

```python
def roulette_select(population, fitnessed_population):  1 usage
    total_fitness = sum(x[0] for x in fitnessed_population)
    pick = random.uniform( a: 0, total_fitness)
    current = 0
    for i, fit in enumerate(fitnessed_population):
        current += fit[0]
        if current > pick:
            return population[i]
```

Tournament Selection: Pri tournament selection sa náhodne vyberie určitý počet jedincov, ktorí sa zúčastnia turnaja. Z týchto jedincov vyhráva ten, ktorý má najvyššiu kondíciu a je vybraný na kríženie. Tento proces sa opakuje, kým sa nevyberie správny počet jedincov.

Pri výbere turnaja bolo dôležité zabezpečiť, aby sa rovnaké genómy neprenášali do ďalšej generácie, aby bol výber rovnomernejší. Riešením tohto problému je pomerne veľká miera mutácií, ktorá zabezpečuje rozmanitosť genómov.

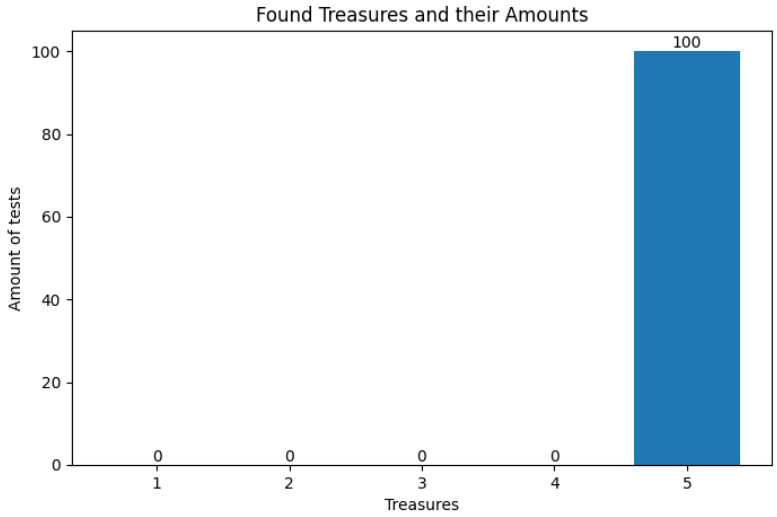
```python
def tournament_select(population, fitnessed_population, k=10):  1 usage
    selected = random.sample(list(zip(population, fitnessed_population)), k)
    selected.sort(key=lambda x: x[1], reverse=True)
    return selected[0]
```

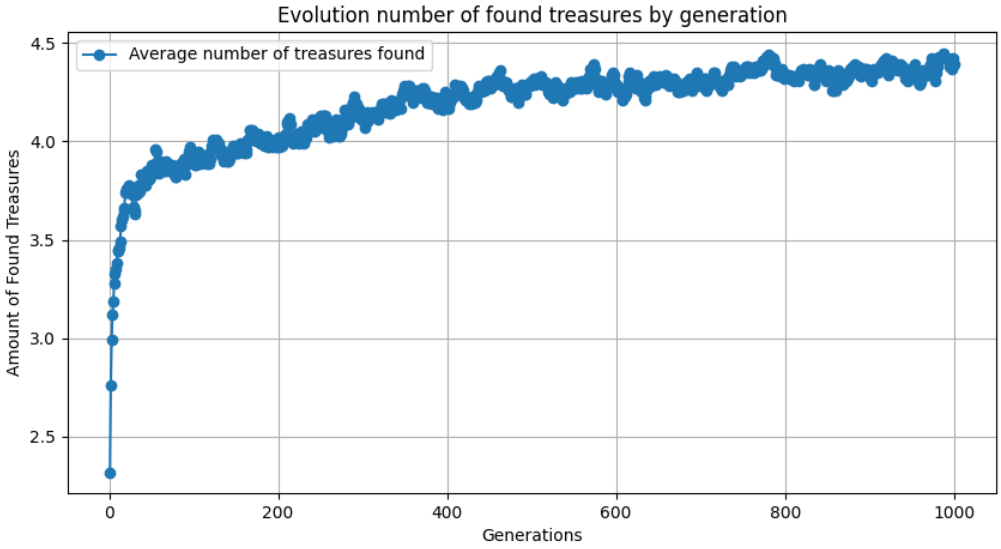
Porovnanie diagramov

Uskutočnilo sa približne 15 testov na 10/100/500 opakovaní (na obrázkoch je jeden z nich na 100 opakovaní)

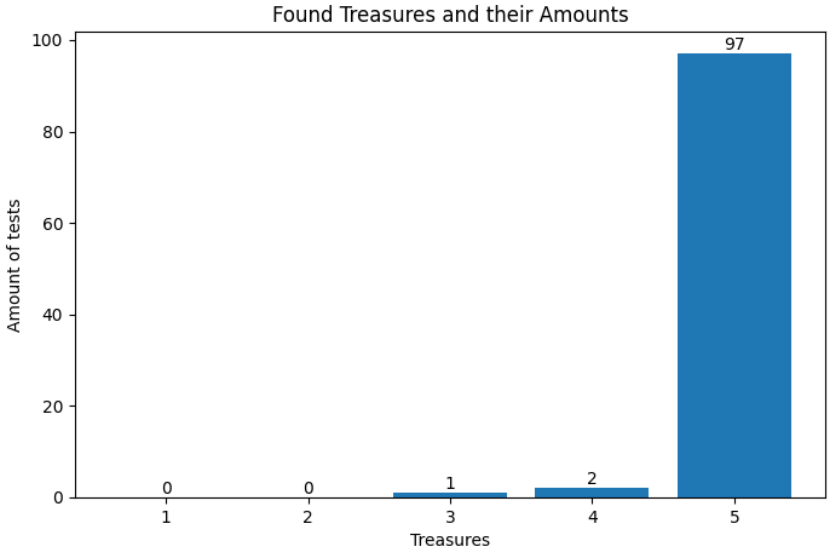
Roulette Wheel Selection:

Artem Delikatnyi
129094

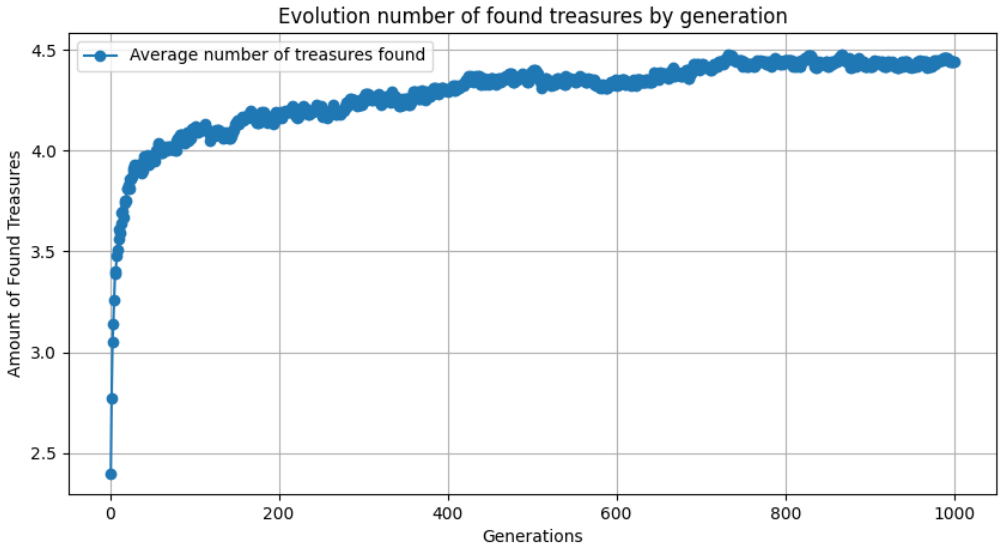




Tournament Selection:



Artem Delikatnyi
129094



Analýza:

Metóda rulety vykazuje stabilnejšie výsledky a rýchlejšie dosahuje maximálny počet nájdených pokladov (5). Je to spôsobené tým, že pri tejto metóde majú jedinci s vyššou fitness vyššiu pravdepodobnosť výberu, čo podporuje rýchly rast efektivity.

Turnajový výber vykazuje menej rýchly rast a menšiu stabilitu, pretože tu je výber náhodnejší a nie vždy vyhrávajú najvhodnejší jedinci. To spôsobuje, že algoritmus v niektorých iteráciách nedokáže nájsť maximálny počet pokladov, čo znižuje úspešnosť.

Porovnanie metódy s použitím Elitarstva a bez neho

Pri analýze sa porovnávali dve metódy:

- **Žiadne elitárstvo** (každá generácia sa vytvára od začiatku, bez zachovania najlepších agentov).
- **Používanie elitárstva** (ponechanie najlepších agentov v novej generácii).

V dvoch prípadoch bola metódou výberu ruleta

Uskutočnilo sa približne 15 testov na 10/100/500 opakovaní (na obrázkoch je jeden z nich na 100 opakovaní) - priemerný rozdiel v percentuálnom pomere je približne **20-25 %**. Algoritmus s elitizmom vykazuje efektívnejšie riešenie ako algoritmus bez neho.

2 Žiadne elitárstvo

```
Wave 97: Generation 999  |    The amount of treasure found = 2 |
Wave 98: Generation 224  |    The amount of treasure found = 5 |
Wave 99: Generation 149  |    The amount of treasure found = 5 |
L P L H P P L L L L P P P P L H
Wave 100: Generation 143 |    The amount of treasure found = 5 |
Percent of Success: 78.0%
```

1 Používanie elitárstva

```
Wave 98: Generation 31   |    The amount of treasure found = 5 |
L P L L P P L L L P P L L L P D L
Wave 99: Generation 57   |    The amount of treasure found = 5 |
P L P L P L P P
Wave 100: Generation 208 |    The amount of treasure found = 5 |
Percent of Success: 100.0%
```

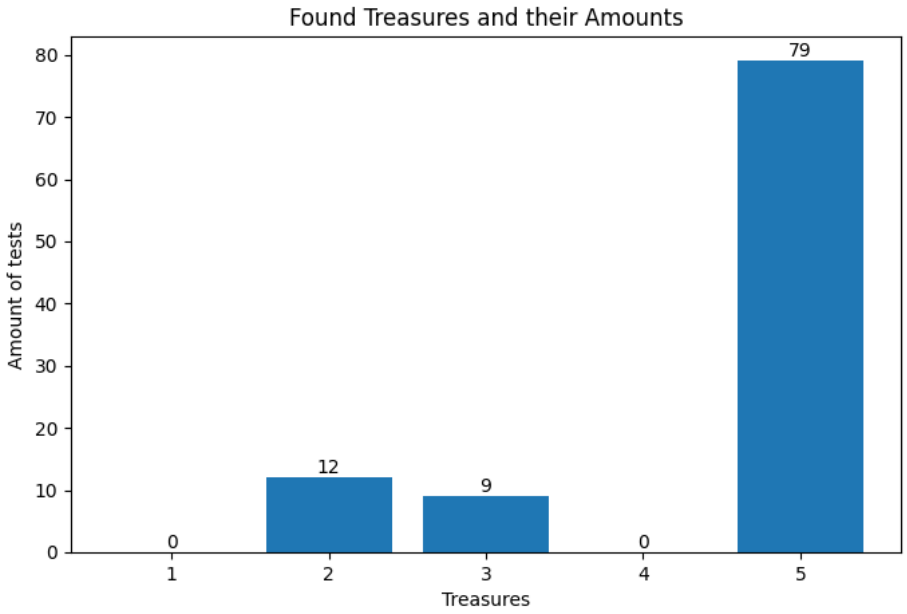
Artem Delikatnyi
129094

Grafic 1

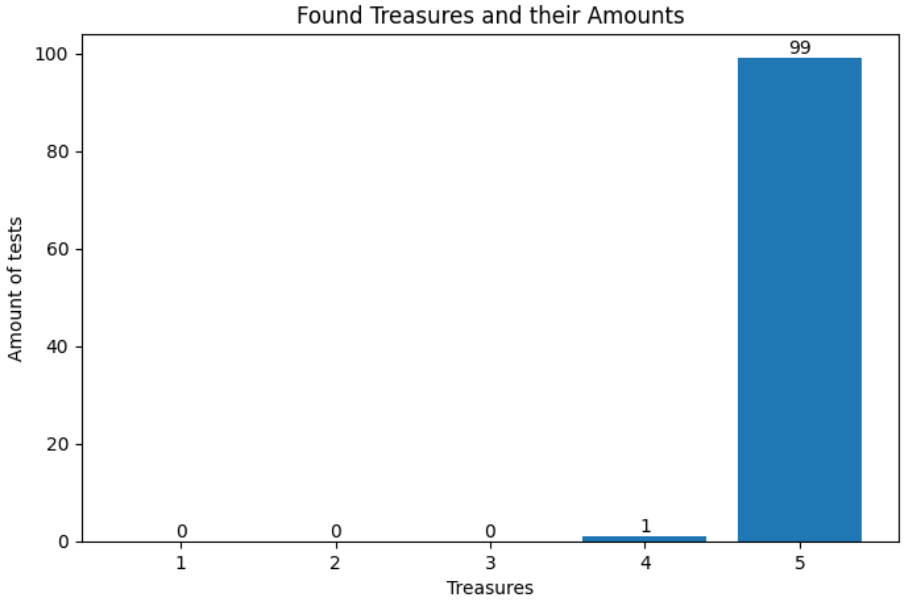
Os X (poklady): Zobrazuje počet nájdených pokladov od 0 do 5. Ide o počet pokladov, ktoré boli nájdené počas generácií.

Os Y (nožstvo): Zobrazuje počet testov, v ktorých sa našlo určité množstvo pokladu.

- **Žiadne elitárstvo**



- **Používanie elitárstva** (ponechanie najlepších agentov v novej generácii).



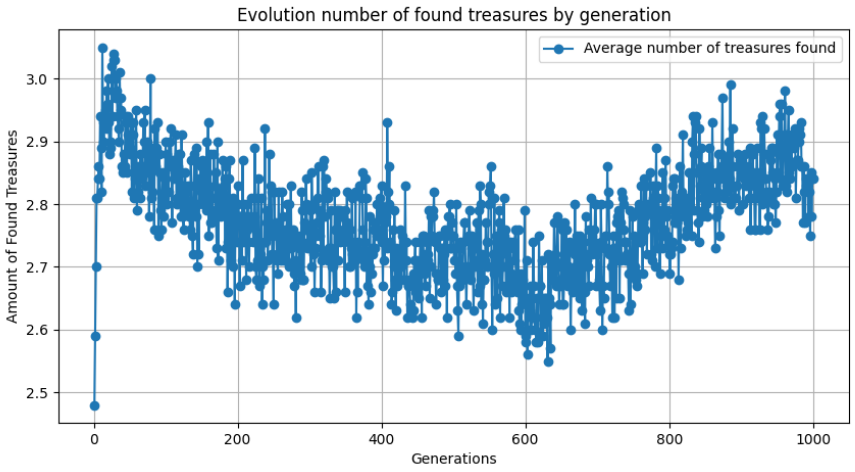
Artem Delikatnyi
129094

Grafy ukazujú, že pravdepodobnosť nájdenia všetkých 5 pokladov pre algoritmus bez použitia elitizmu je oveľa nižšia v porovnaní s algoritmom, ktorý elitizmus používa. Dôvodom je strata kvalitných genómov pri prechode z jednej generácie do druhej.

Grafic 2

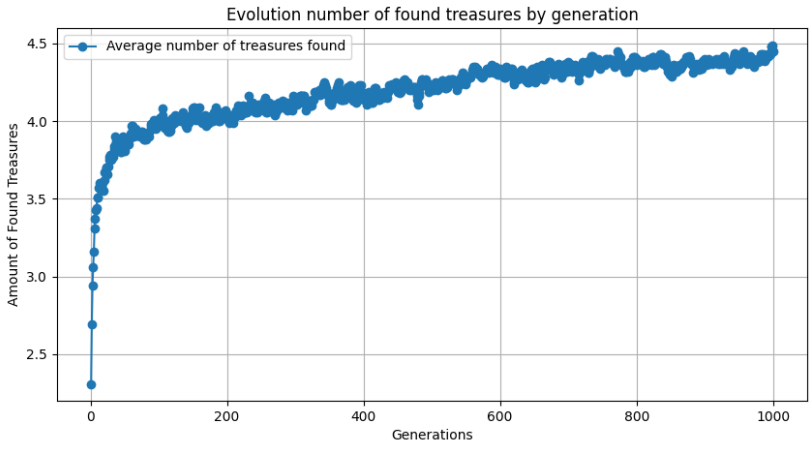
1. **Os X (Generations):** Zobrazuje priemerný počet nájdených pokladov v určitých fázach generovania.
2. **Os Y (Amount of Found Treasures):** Predstavuje generáciu od 0 do 1000.

- **Žiadne elitárstvo**



--

- **Používanie elitárstva** (ponechanie najlepších agentov v novej generácii).



Z týchto grafov vidíme, že výsledok algoritmu bez použitia elitárstva je náhodnejší vzhľadom na to, že najlepšie genómy sa často strácajú. Pri použití elitárstva je vidieť, že výsledok sa postupne neustále zlepšuje, čo je spôsobené zachovaním najlepších genómov, krížením a mutáciou. Ak sa elitárstvo nepoužije, väčšina dôležitých genómov sa stratí a výsledok sa stane len náhodným.