**ביולוגיה חישובית   
80-512 89-512**

**תרגיל 2 – אלגוריתם גנטי   
מגישים (הגשה משותפת סטודנטים מביולוגיה חישובית ומדעי המחשב):   
 אייל הבר - 203786298 (ביולוגיה חישובית)   
אייל ברילינג - 208204230 (מדעי המחשב)**

**קישור להורדת הקבצים מ-GitHub: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**בתרגיל נתבקשנו לפתור לוחות של חידת מספרים הנקראת פוטושיקי Futoshiki על ידי שימוש באלגוריתם גנטי**

הוראות מפורטות ומספר רב של לוחות ניתן למצוא באתר: <https://www.futoshiki.org/.>))

* לצורך הרצת התרגיל יש להשתמש בספריות:

**import sys**

**import random as rand**

**import numpy as np**

**import copy**

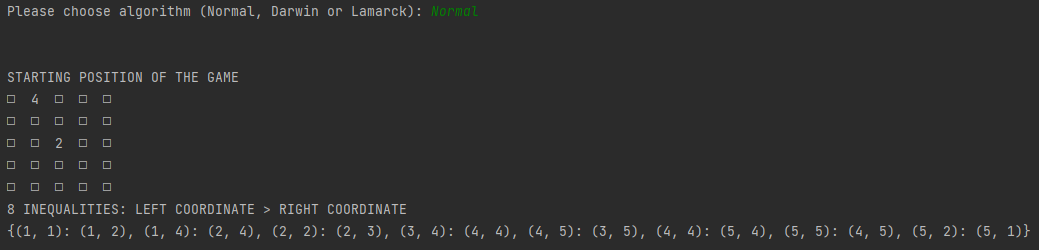
ניתן להוסיף ספרייה נוספת על מנת לראות גרף ניתוח תוצאות לאלגוריתם הנבחר:

**import matplotlib.pyplot as plt** # for algorithm analysis plot

* לאחר מכן יש להריץ את הקובץ בעזרת הכנסת קובץ txt. בפורמט המתבקש כארגומנט  
  באופן הבא (לאחר קידומת python מתאימה): **ex2.py test.txt**

חלק א

לאחר בחירת האלגוריתם הגנטי הרגיל (Normal) מוצג לנו מסך המשחק ההתחלתי, כולל אילוצים:



זה למעשה דור 0.  
לאחר יצרנו את האוכלוסייה בדור הראשון בעזרת הצבות רנדומליות בתאים הריקים, הצבות אלה יהיו בהתאם לטווח המספרים 1 ל-size וכל עוד המספרים במקומות ההתחלתיים נשארים קבועים לכל אורך המשחק.  
כל אוכלוסייה תהיה בגודל 100   
את האוכלוסייה בכל אחד מהדורות הבאים יצרנו בעזרת שימוש באלגוריתם הגנטי:

מימוש האלגוריתם הגנטי **1.** ייצוג הפתרונות **-**כל הפתרונות מיוצגים באמצעות מטריצה בהתאם לגודל שהוגדר בקובץ הקלט בשורה 1 (5\*5, 6\*6, 7\*7)  
מימשנו פונקציה (print\_mat) שתקבל ארגומנט מטריצה בייצוג מערך ותחזיר אותה כייצוג דו ממדי.

2. פונקציית הערכה **-**מימשנו פונקציה (mat\_score) שתחזיר את ה-fitness score של כל מטריצה.  
הגדרנו ערך שגיאות מקסימלי המורכב ממספר השורות והעמודות (size + size) ומספר אילוצי ה"<".  
זה יהיה ערך ה-fitness הגרוע ביותר.   
כמובן שהערך ששואפים אליו הוא 0 שגיאות שמגיע רק לאחר הגעה לפתרון הפאזל.  
בדקנו fitness score בעזרת מעבר על כל השורות והעמודות במטריצה לראות אם הן מכילות מספרים ייחודיים בלבד ולאחר מכן בדקנו אם אילוצי ה"<" מתקיימים. בכל פעם ששורה / עמוד / אילוץ מתקיים החסרנו נקודה ממספר השגיאות המקסימלי.

3. אופן ביצוע הכלאה (**cross-over**) בין פתרונות שונים **-**מימשנו פונקציה (crossover) המקבלת שתי מטריצות, הופכת אותן לייצוג בינארי ("גדילים") ואז הגרלת אינדקס בו הם הם "ישוחלפו" (בדומה לתהליך שחלוף בין שני מקטעים כרומוזומליים) ל"גדיל" בינארי משותף אותו נמיר חזרה לייצוג מטירציוני.

4. אופן ביצוע מוטציות **-**מימשנו פונקציה (mutation) המקבלת מטריצה ומבצעת בה מוטציה בעזרת המרה לגדיל חד ממדי והגרלת שני אינדקסים שונים בהם הערכים יוחלפו ולאחר מכן המרה חזרה להצגה מטריציונית.

5. התמודדות עם בעיות ההתכנסות המוקדמת **-**זהו אחד המכשולים הטכניים הגדולים ביותר עבור אלגוריתמים גנטיים. זה קורה בגלל עקרון הסלקטיביות בו אנחנו נותנים עדיפות לפתרונות הטובים ביותר בכל דור

לאחר שראינו בהרצות כי קיימת התכנסות מוקדמת באופן די שכיח, בנינו מנגנון שמזהה הגעה להתכנסות מוקדמת בעזרת ספירת מספר דורות רב בו ערך ה-fitness הטוב ביותר לא השתנה (הגדרנו 100 דורות ללא שינוי) ולאחר מכן הגדלת משמעותית של שכיחות המוטציות מ-0.15 ל-0.3 כדי לרענן את האוכלוסייה

6. יצירת הדור הבא **-**החלטנו כי בכל דור יהיו 100 פרטים שייוצרו באופן הבא:   
נמצא את המטריצה הטובה ביותר בדור הנוכחי ומיצור לה 10 עותקים בדור הבא (שכפול),   
את 90 המטריצות הבאות ניצור בעזרת הכלאות ולבסוף ביצוע מוטציות על כל האוכלוסייה בהסתברות מתאימה.

7. מספר הדורות להרצת האלגוריתם **-**לאחר מספר נסיונות ראינו כי האלגוריתמים מתכנסים ל-fitness score יחסית נמוך לאחר כ-500 דורות ולכן זה המספר אותו בחרנו

חלק ב

**בחלק זה יצרנו שני אלגוריתמים נוספים**:

* האלגוריתם הגנטי הדארוויני **(Darwin)** -  
  כל פתרון (מטריצה) עובר אופטימיזציה וה-fitness שלו נקבע רק לאחר האופטימיזציה.   
  אבל הדור הבא נוצר על פי הפתרון המקורי **לפני** האופטימיזציה.
* האלגוריתם הגנטי הלאמארקי **(Lamarck)** -  
  כל פתרון (מטריצה) עובר אופטימיזציה וה-fitness שלו נקבע רק לאחר האופטימיזציה.   
  אבל הדור הבא נוצר על פי הפתרון **אחרי** האופטימיזציה.

שני האלגוריתמים פועלים באופן זהה לאלגוריתם הנורמלי (Normal) שבסעיף א' אך בנוסף לכך מפעילים פונקציית אופטימיזציה (optimize) על כל הפרטים (מטריצות) באוכלוסייה על מנת לנסות ולשפר את הניקוד.

פונקציית האופטימיזציה  
מותאמת לכל גודל של לוח משחק שניתן. כלומר מוגדר שלכל היותר יהיו size צעדים לשיפור בכל איטרציית אופטימיזציה. יתרה מכך, קיימת התאמת צעדים מקסימלי לכל מטריצה בהתאם לציון שלה. כלומר, ככל שציון ה-fitness של המטריצה נמוך יותר (טוב יותר) יש פחות צעדים שצריך לשפר בפונקציה (בכל מקרה קיים מנגנון "הגנה" בפונקציה שמחזיר את המטריצה שהוכנסה אם לא התקיים שיפור בציון ה-fitness, מנגנון זה מבטיח שגם אם היינו מחליטים על סף של מספר שינויים קבוע כגודל המטריצה זה עדיין זה היה עובד אך החלטנו שהמימוש בו השתמשנו אלגנטי יותר)

**מימשנו את הפונקציה באופן הבא**:  
תחילה, הפונקציה עוברת שורה שורה ובודקת אם יש כפילויות, ואם כן מחליפה את אחד המספרים במספר שחסר. לאחר מכן אם עדיין קיים מקום לשיפור היא עוברת על ערכי האילוצים "<" , בודקת את תקינותם ומחליפה ערכים בתאם לצורך

ניתוח תוצאות והשוואת אלגוריתמיםלכל אלגוריתם בדקנו שלושה גדלים שונים (5,6,7) ושתי דרגות קושי שונות (easy, tricky)  
נעזרנו באפשרויות באתר - [https://www.futoshiki.org](https://www.futoshiki.org/.) על מנת להפיק 6 קבצים (בנוסף לקובץ הדוגמא):

test.txt, 5\_easy.txt, 5\_tricky.txt, 6\_easy.txt, 6\_tricky.txt, 7\_easy.txt, 7\_tricky.txt

**success accuracy table:**

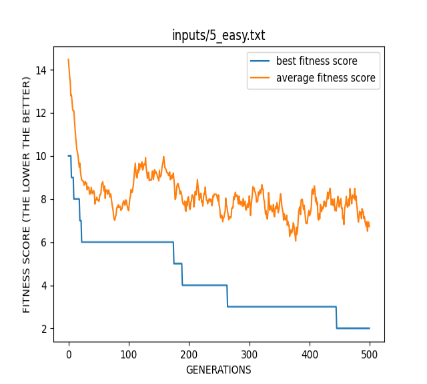
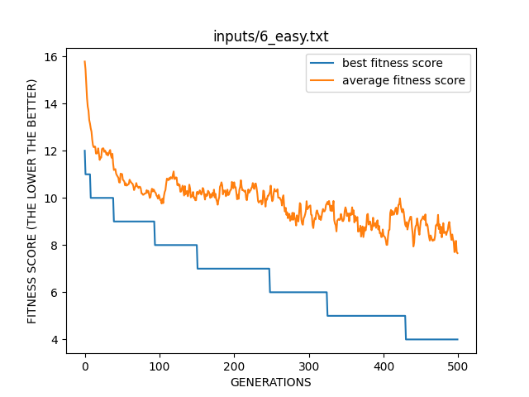
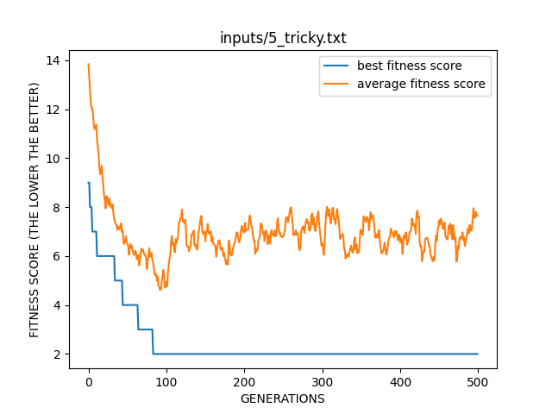
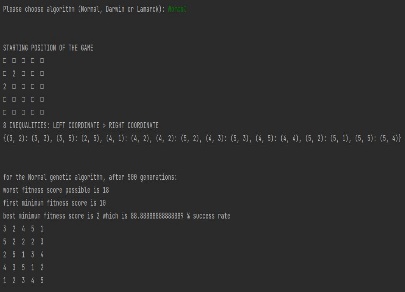
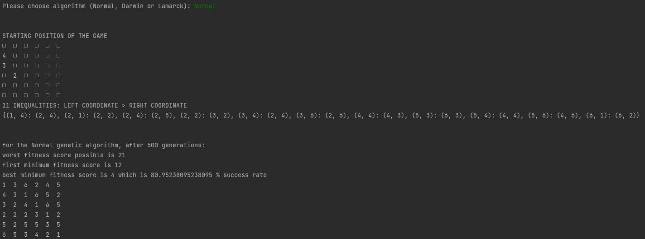
**אחוז ההצלחה חושב באופן הבא**:

|  | **5 easy** | **5 tricky** | **6 easy** | **6 tricky** | **7 easy** | **7 tricky** | average |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Normal** | 88.8% | 88.23% | 80.95% | 83.33% | 80.76% | 86.66% | **84.78%** |
| **Darwin** | 83.33% | 82.35% | 80.95% | 83.33% | 88.46% | 90.0% | **84.73%** |
| **Lamarck** | 88.8% | 88.23 | 85.71% | 87.5% | 88.46% | 93.33% | **88.67%** |

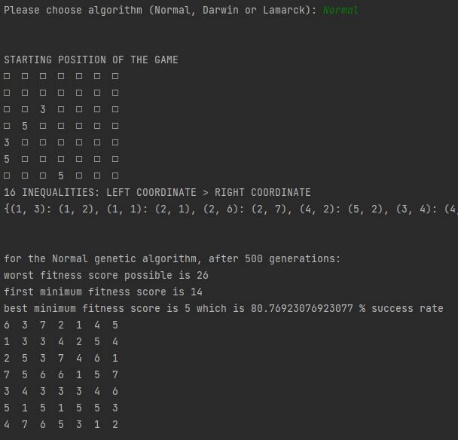
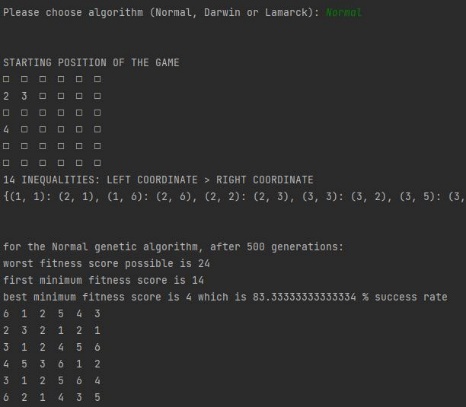
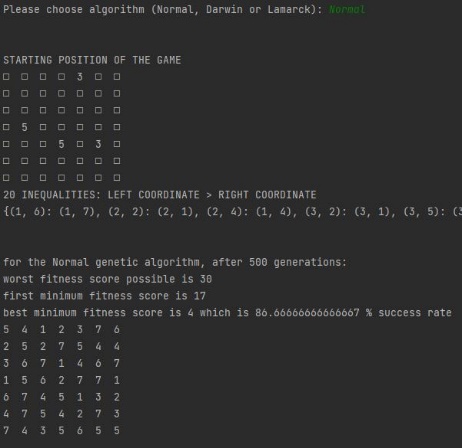
ניתן לראות כי האלגוריתם הגנטי של למארק משיג את אחוז ההצלחה הממוצע הגבוה ביותר מבין שלושת האלגוריתמים. תוצאה זו הגיונית כיוון שעל פי אופן פעולתו של אלגוריתם זה, הדור הבא נוצר מתוך כלל הפתרונות האפשריים לאחר שעברו אופטימיזציה, ולכן הסיכוי לקבל בדור הבא מטריצה מינימלית בעלת fitness score נמוך יותר הינו גבוה יותר.  
  
לעומת זאת, האלגוריתם של דרווין והאלגוריתם הנורמלי די זהים מבחינת אחוז הצלחה (לטובת הנורמלי), דבר שאולי נשמע מפתיע בהתחלה בהתחשב בזה שהאלגוריתם של דרווין נעזר באופטימיזציה, אבל כיוון שהוא גם מעביר לדור הבא את הפתרונות לפני האופטימזציה (בדומה לאלגוריתם הנורמלי) זה לא יבטיח תוצאות טובות יותר בהכרח כפי שנוכחנו לראות הלכה למעשה.

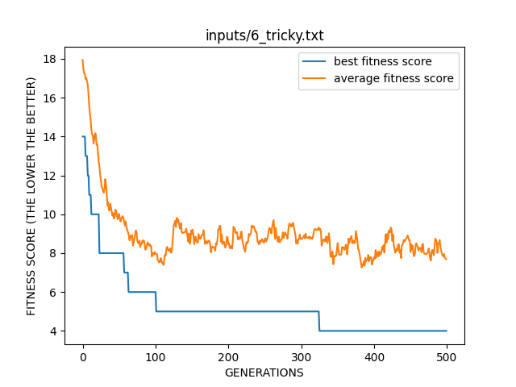
בנוסף, לא ניתן לראות קשר ישיר חד משמעי בין דרגות הקושי השונות לאחוז ההצלחה היחסי

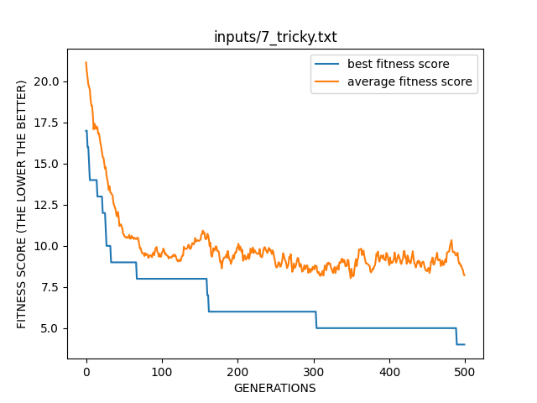
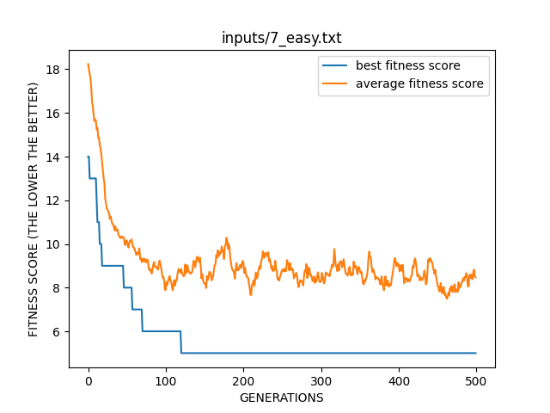
**בשלושת העמודים הבאים מצורפים כל ה-figures המתאימים בהם השתמשנו לעיבוד הנתונים.  
(כל אלגוריתם יוצג במסגרת של עמוד אחד - סה"כ: 4 עמודי דו"ח + 3 עמודי נספחים, כך שמסמך זה אכן עומד בדרישות התרגיל)**

**תמונה שמכילה טקסט

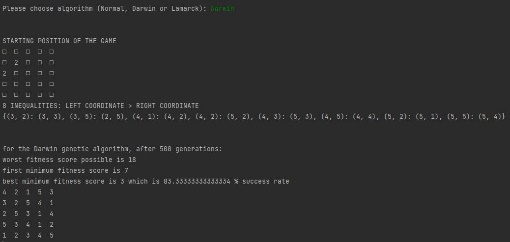
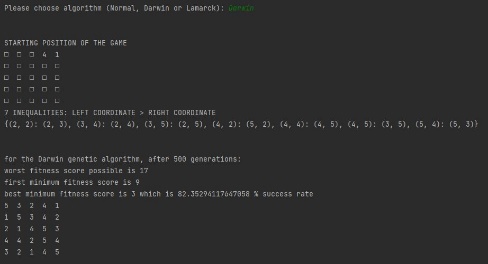
התיאור נוצר באופן אוטומטיNormal genetic algorithm results**

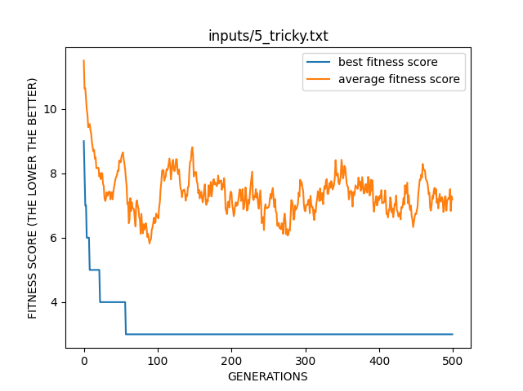
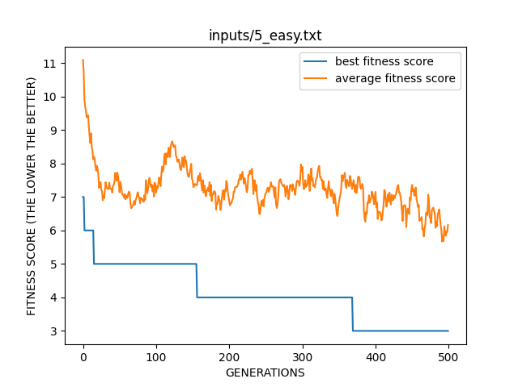
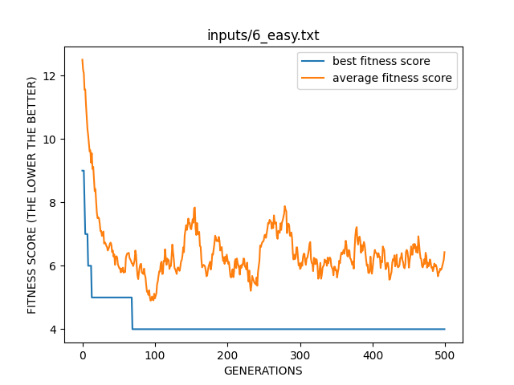
****

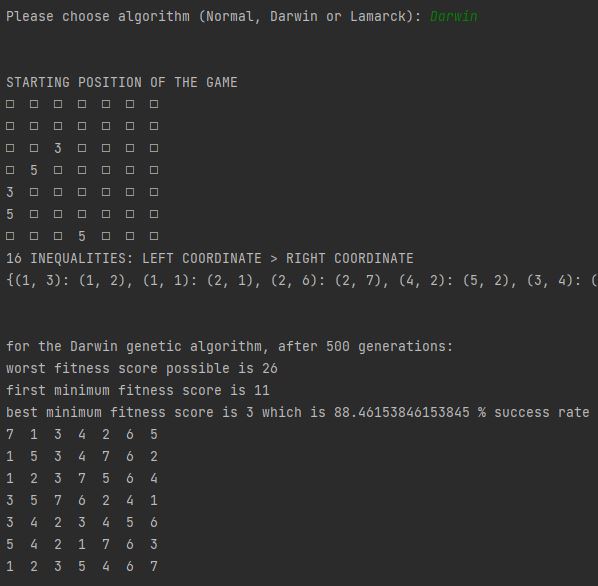
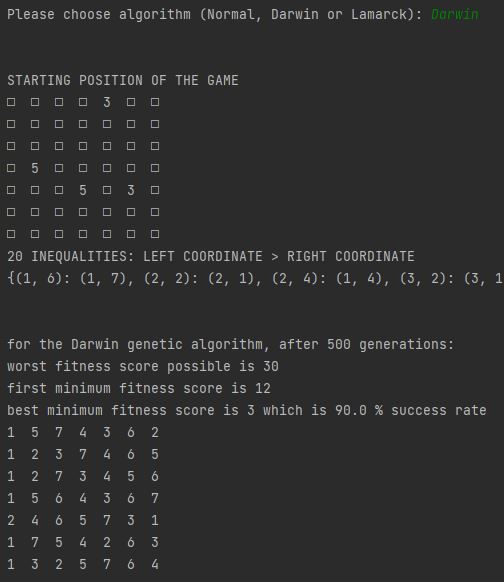
****

****

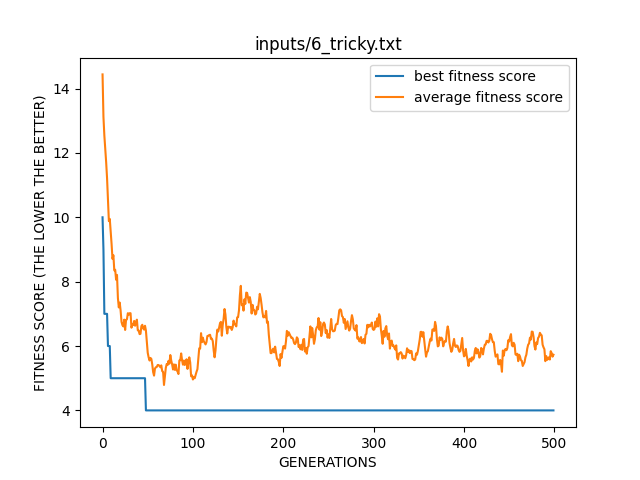
**Darwin genetic algorithm results**

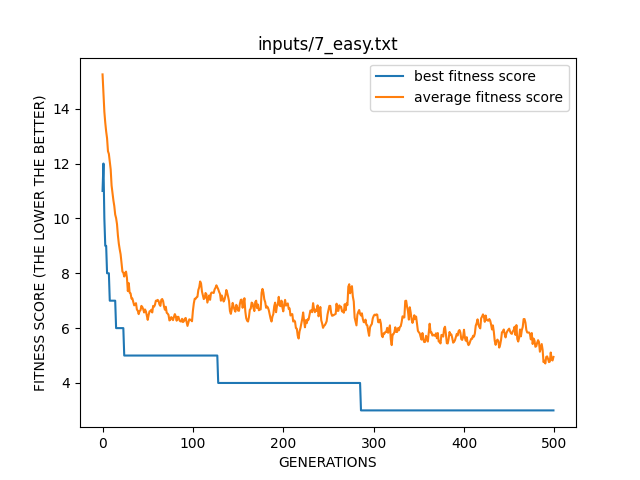
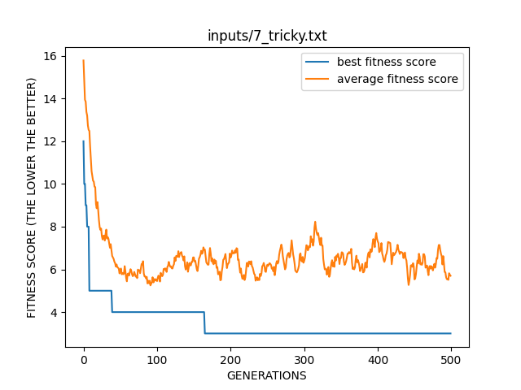
****

****

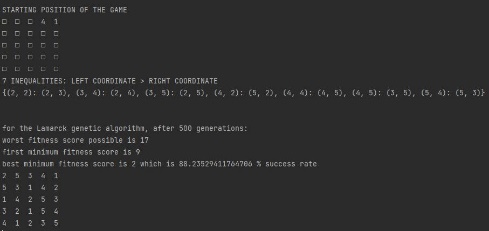
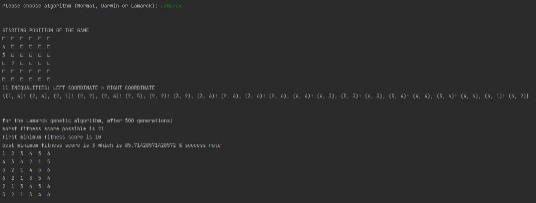
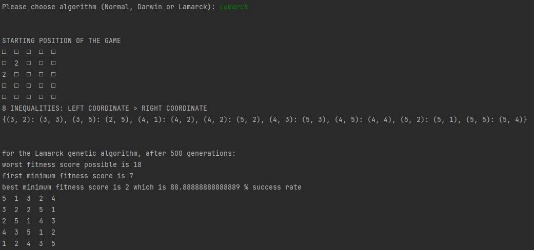
****

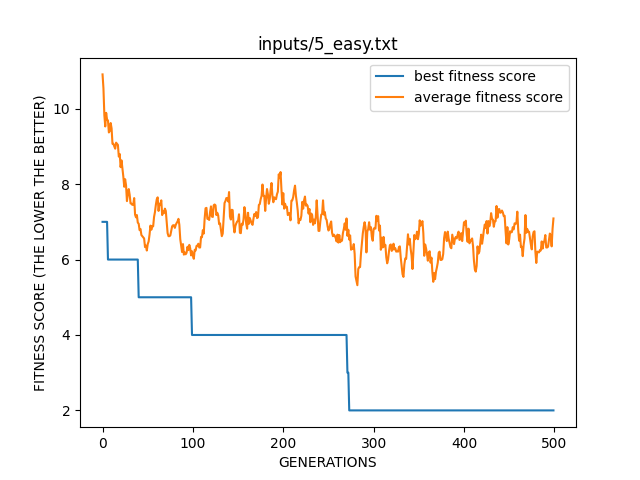
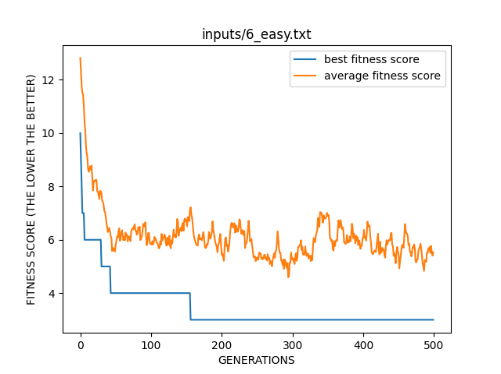
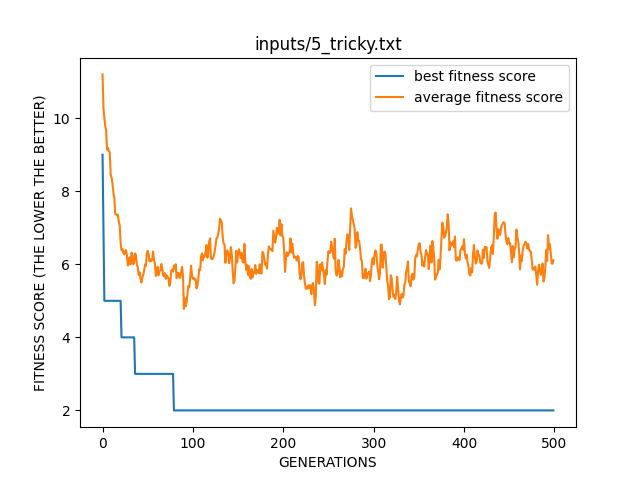
****

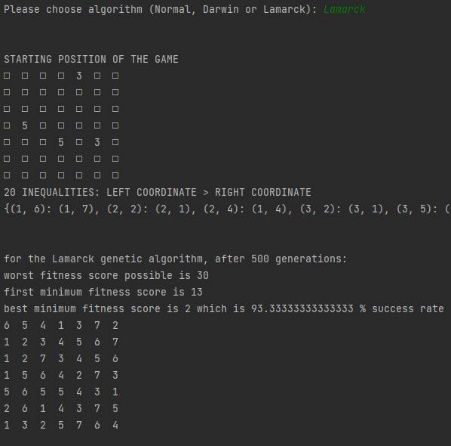
****

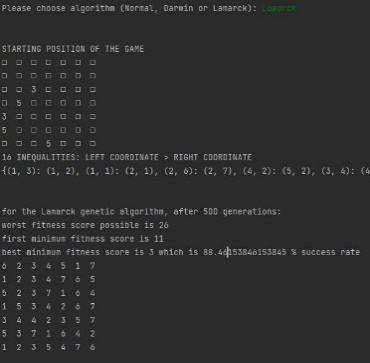
****

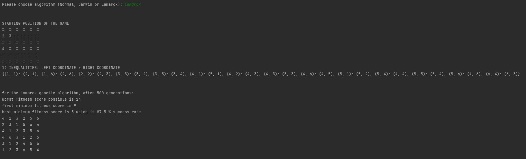
**Lamarck genetic algorithm results**

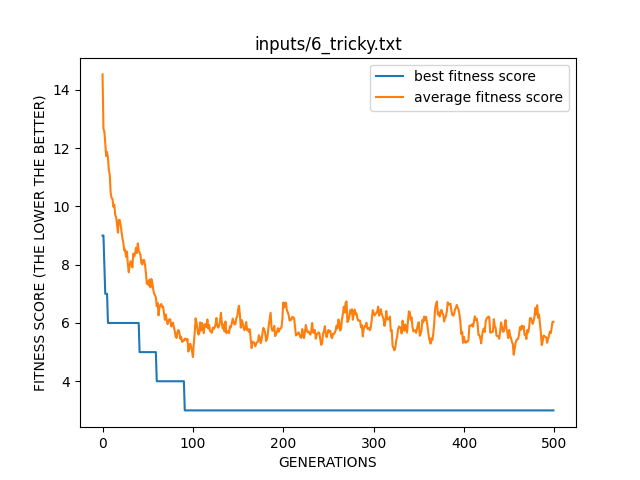
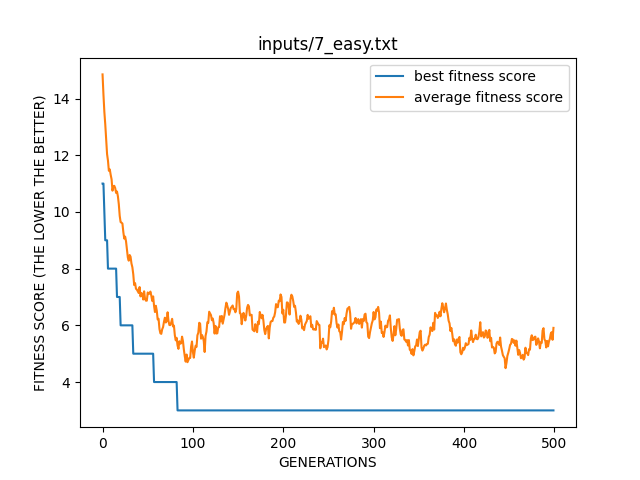
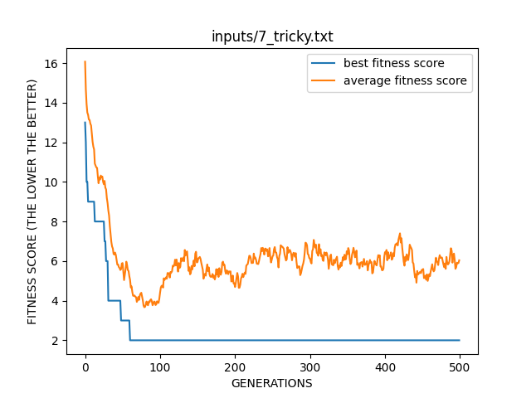
****

****

****

****

****

****