Nonogram

Chromosom dla tego zadania przyjmuje następującą postać:

- Jest ciągiem binarnym 0 oznaczają puste pola, a 1 zamalowane
- Jego długość określa rozmiar planszy, są to wszystkie pola na planszy ułożone rzędami, np. plansza 5x5 daje długość 25 bitów.

Funkcja fitness po pierwsze zamienia otrzymany chromosom na macierz, aby ułatwić sprawdzanie warunków. Następnie puszczona jest pętla: najpierw po wszystkich wierszach, a następnie po wszystkich kolumnach, w pętlach dodajemy do sumy wynik z obliczeń wiersza/kolumny przez funkcję *better_checker*.

```
for(i in 1:m){
  suma = suma + better_checker(matrx[i,], rows[[i]])
}
for(i in 1:n){
 suma = suma + better_checker(matrx[,i], cols[[i]])
}
```

Zostały utworzone dwie funkcje do sprawdzania poprawności:

- *basic_checker* - za każdy poprawny wiersz/kolumne odejmuje -1 od sumy, a za niepoprawny zwraca 0. Na początku funkcji sprawdzamy czy ilość zamalowanych pól jest równa ustalonym regułom, jeśli nie to zwraca 0. W czasie przetwarzania działa on następująca, szuka pierwszej jedynki, jeśli znajdzie zwiększa licznik sekwencji i ustawie flagę sekwencji na true. Jeśli znowu trafi na jedynkę, znowu zwiększa licznik i sprawdza czy długość sekwencji nie jest za długa, jeśli tak zwraca 0. Jeśli trafi na 0 i długość sekwencji jest odpowiednia ustawia flagę sekwencja na false i zeruje licznik. Sprawdzą czy są jeszcze jakieś reguły. Jeśli długość sekwencji przy trafieniu na 0 jest nieodpowiednia zwraca 0. Jeśli przejdzie wszystko bez problemów zwraca -1.
- *better_checker* - nieco bardziej skomplikowana, która liczy punkty w następujący sposób:
    - Jeśli zamalujemy wszystkie pola poprawnie, mimo tego próbuje jeszcze pomalować kolejne, to za każde pomalowane + 1.
    - Za dobrze pomalowaną sekwencję -1
    - Jeśli po dobrze pomalowanej sekwencji, zamalujemy następne pole +1, jeśli nie pomalujemy lub jest to koniec wiersza/kolumny +1
    - Za niedokończenie sekwencji +1

Przykład:

```
rows = list(c(1,1), c(2,2), 3, 2, 2)
cols = list(c(2,2), 4, 1, 2, 2)
```

Reguły dla wierszy i kolumn podajemy w postaci listy.

Program drukuje następujące rozwiązanie dla następujących parametrów funkcji rbga.bin:

- Size = n*m
- popSize = 300
- Iters = 100
- mutationChance = 0.05
- Elitism = T
- evalFunc = fitnessFunc

```
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    0    0    0    1
[2,]    1    1    0    1    1
[3,]    0    1    1    1    0
[4,]    1    1    0    0    0
[5,]    1    1    0    0    0
```