In [1]:
```
#from google.colab import drive
#drive.mount('/content/drive/',True)
#!cp "/content/drive/Shared drives/Proyecto Final/4. Diseño Basico/Algortimo Ge
netico Cinematico/ga.py" "ga.py"
#!cp "/content/drive/Shared drives/Proyecto Final/4. Diseño Basico/Algortimo Ge
netico Cinematico/GlobalIndexKinematical.py" "GlobalIndexKinematical.py"
```

In [2]:
```
import numpy as np
import matplotlib.pyplot as plt
import ga
import GlobalIndexKinematical as km
```

In [3]:
```
# Inputs or constants
span = [[500,50,50,-50],[700,150,250,50]]

# Number of variables for optimization
num_var = 4
num_kromo = 25 # Number of Chromosomes
k = 100 # Number of Generations
pop_size = (num_kromo, 1)

# Random Population
new_pop1 = np.random.uniform(span[0][0], span[1][0], size=pop_size)
new_pop2 = np.random.uniform(span[0][1], span[1][1], size=pop_size)
new_pop3 = np.random.uniform(span[0][2], span[1][2], size=pop_size)
new_pop4 = np.random.uniform(span[0][3], span[1][3], size=pop_size)

new_population = np.concatenate((new_pop1,new_pop2, new_pop3, new_pop4),axis=1)
print('Random Population:\n',new_population,'\n')

# Point Cloud
P = km.WorkspaceDesired(500.0,650.0,50.0)

# Number of Parents
num_parents = int(num_kromo/2)

Global_fitness = []
Avg_fitness=[]
Elite = []
```

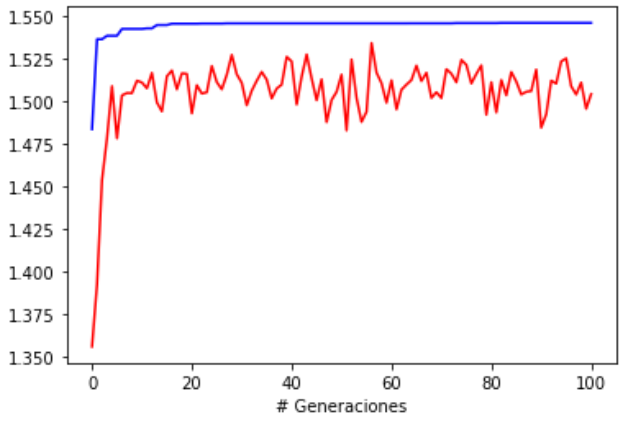
```
Random Population:
 [[ 6.92767189e+02  1.25598934e+02  6.86698035e+01 -3.97599361e+00]
 [ 6.61699780e+02  1.31247010e+02  5.19448931e+01 -1.08372879e+01]
 [ 6.45899126e+02  1.40420739e+02  8.72193984e+01  1.74541933e+01]
 [ 5.08185869e+02  8.34649762e+01  2.27108711e+02  2.64237721e+01]
 [ 5.98485322e+02  1.37573348e+02  2.38480591e+02 -4.42772946e+01]
 [ 6.11746130e+02  9.04814880e+01  1.98885528e+02 -3.32970202e+00]
 [ 6.77270510e+02  9.45392171e+01  2.20080874e+02 -6.81344092e-01]
 [ 6.73177581e+02  8.31149856e+01  2.16679455e+02 -4.34454584e+01]
 [ 6.38062526e+02  7.99022358e+01  1.58112699e+02  3.32518559e+01]
 [ 5.30262639e+02  6.08773455e+01  2.44846099e+02  4.71769862e+00]
 [ 6.16648876e+02  1.47617218e+02  1.01504824e+02 -2.22421104e+01]
 [ 6.60613624e+02  1.05087542e+02  1.63825099e+02 -1.89462208e+01]
 [ 6.37311976e+02  5.86457647e+01  1.73102720e+02 -4.73284536e+01]
 [ 6.82271920e+02  6.40093835e+01  1.03301237e+02  1.96015777e+01]
 [ 6.38770540e+02  1.37928283e+02  1.18377233e+02 -3.22768470e+01]
 [ 5.10323577e+02  6.35758927e+01  2.28212119e+02 -3.03427517e+01]
 [ 6.99594607e+02  6.80215330e+01  6.98369397e+01 -4.69071078e+00]
 [ 6.40792007e+02  1.05794555e+02  1.67061123e+02  6.94095879e+00]
 [ 6.24089972e+02  9.85904424e+01  6.85477105e+01 -2.46683483e+00]
 [ 6.05953081e+02  1.05818773e+02  1.10732588e+02  3.00445403e+01]
 [ 5.18713018e+02  1.48304605e+02  2.27888187e+02 -4.73997781e+01]
 [ 6.96899219e+02  1.33225876e+02  2.31577663e+02  3.69459969e+01]
 [ 5.29362761e+02  7.76391866e+01  5.97513235e+01 -1.27021103e+00]
 [ 5.87706631e+02  6.94406374e+01  1.54570029e+02  3.53311009e+01]
 [ 6.56590360e+02  5.03822189e+01  1.83972610e+02 -1.81821333e+01]]
```

In [4]:
```
for i in range(k):
    # Evaluate Fitness
    print('\n',f'Evaluate Fitness of Generation {i}:')
    fitness = ga.fitnessK(new_population,P,False)
    Global_fitness.append(max(fitness))
    Avg_fitness.append(np.average(fitness))
    print(f'Maximun Fitness: {Global_fitness[-1]} and Mean Fitness: {Avg_fitnes
s[-1]}')
    Elite.append(new_population[fitness.index(Global_fitness[-1])])

    # Select Best Fitness
    parents = ga.select_parents(new_population,fitness,num_parents)
    #print('Parents Selected:\n',parents, '\n')

    # Crossover
    offspring_cross = ga.crossover(parents,num_kromo)
    #print('crossover:\n',offspring_cross, '\n')

    # Mutation
    mut_prob = 0.6
    offspring_mut = ga.mutation(offspring_cross,span, mut_prob)
    #print('Mutation:\n',offspring_mut, '\n')

    # Pop
    new_population = np.concatenate((parents, offspring_mut))
    #print('Generation ', i, ':\n', new_population)

# Evaluate Fitness
print('\n',f'Evaluate Fitness of Generation {k}:')
fitness = ga.fitnessK(new_population,P)
Global_fitness.append(max(fitness))
Avg_fitness.append(np.average(fitness))
print(f'Maximun Fitness: {Global_fitness[-1]} and Mean Fitness: {Avg_fitness[-
1]}')
Elite.append(new_population[fitness.index(Global_fitness[-1])])
```

```
 Evaluate Fitness of Generation 0:
Maximun Fitness: 1.4837485162959745 and Mean Fitness: 1.3559988228445803

 Evaluate Fitness of Generation 1:
Maximun Fitness: 1.5365919197764721 and Mean Fitness: 1.392435074156334

 Evaluate Fitness of Generation 2:
Maximun Fitness: 1.5365919197764721 and Mean Fitness: 1.4537818980600963

 Evaluate Fitness of Generation 3:
Maximun Fitness: 1.5386399753441866 and Mean Fitness: 1.478479720846861

 Evaluate Fitness of Generation 4:
Maximun Fitness: 1.5386399753441866 and Mean Fitness: 1.5090215663119815

 Evaluate Fitness of Generation 5:
Maximun Fitness: 1.5386399753441866 and Mean Fitness: 1.4783307904954195

 Evaluate Fitness of Generation 6:
Maximun Fitness: 1.542659300528366 and Mean Fitness: 1.5036208175653905

 Evaluate Fitness of Generation 7:
Maximun Fitness: 1.542659300528366 and Mean Fitness: 1.5049938186670175

 Evaluate Fitness of Generation 8:
Maximun Fitness: 1.542659300528366 and Mean Fitness: 1.5049327250002946

 Evaluate Fitness of Generation 9:
Maximun Fitness: 1.542663563130835 and Mean Fitness: 1.5122285081917262

 Evaluate Fitness of Generation 10:
Maximun Fitness: 1.542663563130835 and Mean Fitness: 1.5111865007278829

 Evaluate Fitness of Generation 11:
Maximun Fitness: 1.5429912966141803 and Mean Fitness: 1.5077290007255395

 Evaluate Fitness of Generation 12:
Maximun Fitness: 1.5429912966141803 and Mean Fitness: 1.5167891288966868

 Evaluate Fitness of Generation 13:
Maximun Fitness: 1.544937211926666 and Mean Fitness: 1.499448679621598

 Evaluate Fitness of Generation 14:
Maximun Fitness: 1.544937211926666 and Mean Fitness: 1.4942196410106288

 Evaluate Fitness of Generation 15:
Maximun Fitness: 1.544937211926666 and Mean Fitness: 1.514828875322669

 Evaluate Fitness of Generation 16:
Maximun Fitness: 1.5457214018398262 and Mean Fitness: 1.5182459443815055

 Evaluate Fitness of Generation 17:
Maximun Fitness: 1.5457214018398262 and Mean Fitness: 1.5071246151175857

 Evaluate Fitness of Generation 18:
Maximun Fitness: 1.5457214018398262 and Mean Fitness: 1.5165333572536097

 Evaluate Fitness of Generation 19:
Maximun Fitness: 1.5457214018398262 and Mean Fitness: 1.5162510091012797

 Evaluate Fitness of Generation 20:
Maximun Fitness: 1.5457214018398262 and Mean Fitness: 1.493037270179711
```

In [5]:
```
#print(Global_fitness)
print(f'Best Solution: {new_population[fitness.index(max(fitness))]}')

plt.plot(Global_fitness,'b-')
plt.plot(Avg_fitness,'r-')

plt.xlabel('# Generaciones')
plt.show()
```

```
Best Solution: [6.75094566e+02 5.00818563e+01 2.49309904e+02 4.83903096e-03]
```



In [10]:
```
np.savetxt('Records_Fitness.csv',(Global_fitness,Avg_fitness),fmt='%4.4f',delim
iter=',',header='Maximun, Mean')
np.savetxt('Records_Elite.csv',Elite,fmt='%4.4f',delimiter=',',header='R_b, L_
A, L_D, e',comments='Best of each Generation')
np.savetxt('Last_Generation.csv',new_population,fmt='%4.4f',delimiter=',',heade
r='R_b, L_A, L_D, e')
np.savetxt('Last_Fitness.csv',np.matrix(fitness).T,fmt='%4.4f',delimiter=',',he
ader='Fitness')

#!cp "Records_Fitness.csv" "/content/drive/Shared drives/Proyecto Final/4. Dise
ño Basico/Algortimo Genetico Cinematico/Records_Fitness.csv"
#!cp "Records_Elite.csv" "/content/drive/Shared drives/Proyecto Final/4. Diseño
Basico/Algortimo Genetico Cinematico/Records_Elite.csv"
#!cp "Last_Generation.csv" "/content/drive/Shared drives/Proyecto Final/4. Dise
ño Basico/Algortimo Genetico Cinematico/Last_Generation.csv"
#!cp "Last_Fitness.csv" "/content/drive/Shared drives/Proyecto Final/4. Diseño
Basico/Algortimo Genetico Cinematico/Last_Fitness.csv"
```