Author: Gabriel Padilha Alves

This work was NOT developed using the incomplete code provided by the teacher.

Observations:

- High crossover probability and/or high mutation probability end up impairing the optimization process, as shown in the figures inside 'results' folder;

- The last generation can be worse than previous generations, because the selection and mutation mechanisms can produce less adapted children;

- Using Selection II, where *parent(i)* (from population "*pop*") is kept instead of *child(i)* (from population "*new_pop*"), if the parent has better fitness - where *(i)* is the position in the population array - solves the latter problem and converges much quicker, since the fittest individuals are not replaced. This can be observed when comparing the results from the folder "\results\without_selectionII" with the ones inside "\results\with_selectionII", specially when looking at the figures which contain the average fitness of the population;

- Besides the Selection II adaptation, when the chromosome is out of bounds, there is a 50% chance that this chromosome will be adjusted to a value inside the domain or 50% chance that will be adjusted to the closest boundary (the lower bound if less than zero or the upper one if more than pi);

- Another implementation worth mentioning is regarding the solution to the generation of "NaN's" (Not a Number), which happens when the crossover or the mutation generate a string of bits that corresponds to this float called "NaN", which can be seen almost as a negative infinity ("11111111110000000000000000000000") or a positive infinity ("01111111110000000000000000000000") - where any of the zeros can be replaced by ones and it will still be "NaN" (source: https://www.h-schmidt.net/FloatConverter/IEEE754.html). The solution adopted to solve this problem was to change the "NaN" value to the lower bound or the upper bound, depending on the first bit (if "0", then upper bound, else it changes to the lower bound);

- There are other types of selection that could be used to improve the algorithm even further (e.g. Tournament selection), just as there are other types of mutations.