Assume that the *step_size* for cyclic minimization is 0.001:
step_size <- 0.001

Name: find_local_minimum
Input: *func*, an arbitrary function, the func takes in a sequence to evaluate the input.
n, an integer representing the number of variables that func has.
Output:
a sequence: representing the location of the local minimum of *func* in the form of (X1,X2,X3…,Xn) where X1,X2,X3,X4, ….,Xn are all fractional numbers representing the coordinates of the location of the found local minimum.

1. *res_location* <- an empty sequence
2. for each idx in 0,1,2….n-1 do:
insert integer 0 to the end of res_location
3. for each *jdx* in 0,1,2….n-1 do:
a. *step_size* <- 0.001
b. *location_left* <- make a copy of *res_location*
c. *location_rigt* <- make a copy of *res_location*
d. *location_left*$_{\text{-i-1}}$ <- location_left$_{\text{-i-1}}$ — *step_size*
e. *location_rigt*$_{\text{-i-1}}$ <- *location_rigt*$_{\text{-i-1}}$ + *step_size*
f. *one_dimensional_minimum* <- *func* evaluate the input at position *res_location*
g. *left_value* <- *func* evaluate the input at position *location_left*
h. *right_value* <- *func* evaluate the input at position *location_rigt*
k. While *left_value* is smaller than *one_dimensional_minimum* or *right_value* is smaller *one_dimensional_minimum* than do:
I. if *left_value* is smaller than *one_dimensional_minimum:*
res_location <- *location_left*
else:
res_location <- *location_rigt*
II. *location_left* <- make a copy of *res_location*
III *location_rigt* <- make a copy of *res_location*
IIII. *location_left*$_{\text{-i-1}}$ <- *location_left*$_{\text{-i-1}}$ — *step_size*
IIIII. *location_rigt*$_{\text{-i-1}}$ <- *location_rigt*$_{\text{-i-1}}$ + *step_size*

IIIIII. *one_dimensional_minimum* <- *func* evaluate the input at position *res_location*

IV. *left_value* <- *func* evaluate the input at position *location_left*

VI. *right_value* <- *func* evaluate the input at position *location_rigt*

4. Return *res_location*