Ejercicio 21-10-2017 - Ordenaci?N y utilizaci?n de bucles

October 29, 2017

0.1 ALEJANDRO SANTORUM VARELA - Ordenación+bucles - Ejercicios 21/10/2017

EJERCICIO DE ORDENACION UTILIZANDO INSERTSORT:

```
In [1]: def intercambiar(L, i, j):
            if i<j and L[j]<L[i]:
                a = L[i]
                L[i] = L[j]
                L[j] = a
            return L

        def ordenarIS(L):
            longitud = len(L)
            for i in xsrange(0, longitud):
                for j in xsrange(i+1, longitud):
                    L = intercambiar(L, i, j)

            return L


        #comprobamos que las funciones funcionan

        L = [8, 83, 59, 3, 5, 7, 1, 34, 43, 56, 9, 45]
        LL = list()

        LL = ordenarIS(L)
        print(LL)
        print(" ")
```

```
[1, 3, 5, 7, 8, 9, 34, 43, 45, 56, 59, 83]
```

EJERCICIO DE ORDENACIÓN UTILIZANDO MERGESORT

```
In [2]: def intercalar(L, L1, L2):
            l1 = len(L1)
            l2 = len(L2)
```

```
    if l1==0 or l2==0: #comprobación de parada
        return L+L1+L2


    if L1[0]<=L2[0]:
        a = L1.pop(0)
        L.append(a)
        intercalar(L, L1, L2)

    else:
        b = L2.pop(0)
        L.append(b)
        intercalar(L, L1, L2)
    return L+L1+L2


def ordenarMS(L):
    l = len(L)
    if l==0 or l==1:
        return L

    L1 = list()
    L2 = list()
    m = floor(l/2)

    for i in xsrange(0, m):
        a = L.pop(0)
        L1.append(a)

    for i in xsrange(m, l):
        b = L.pop(0)
        L2.append(b)

    L3 = ordenarMS(L1)
    L4 = ordenarMS(L2)

    return intercalar([], L3, L4)


#comprobamos que las funciones funcionan

L = [8,3,7,9,4,2,1]
LL = list()

LL = ordenarMS(L)
print(LL)
print(" ")
```

```
[1, 2, 3, 4, 7, 8, 9]
```

Comparamos los tiempos de ordenación de InsertSort y Mergesort

```
In [3]: #-------INSERTSORT---------
        def intercambiar(L, i, j):
            if i<j and L[j]<L[i]:
                a = L[i]
                L[i] = L[j]
                L[j] = a
            return L

        def ordenarIS(L):
            longitud = len(L)
            for i in xsrange(0, longitud):
                for j in xsrange(i+1, longitud):
                    L = intercambiar(L, i, j)

            return L
        #-------------------------------------

        print("TIEMPO DE INSERTSORT:")
        L = [randint(-1000,1000) for muda in srange(800)]
        %time ordenarIS(L)
```

```
TIEMPO DE INSERTSORT:
CPU times: user 236 ms, sys: 20 ms, total: 256 ms
Wall time: 230 ms
```

```
Out[3]: [-1000,
         -999,
         -997,
         -996,
         -995,
         -994,
         -993,
         -992,
         -989,
         -980,
         -979,
         -973,
         -966,
         -963,
         -963,
```

```
-962,
-962,
-959,
-958,
-957,
-956,
-955,
-951,
-951,
-946,
-944,
-943,
-938,
-936,
-935,
-933,
-932,
-931,
-930,
-929,
-927,
-927,
-923,
-921,
-916,
-914,
-904,
-901,
-887,
-887,
-887,
-887,
-887,
-877,
-877,
-875,
-868,
-866,
-857,
-856,
-853,
-852,
-850,
-844,
-842,
-841,
-839,
-837,
```

```
-835,
-833,
-828,
-828,
-827,
-827,
-827,
-824,
-822,
-818,
-817,
-817,
-815,
-815,
-815,
-812,
-809,
-800,
-800,
-799,
-790,
-775,
-773,
-773,
-771,
-766,
-762,
-759,
-759,
-756,
-746,
-742,
-742,
-739,
-738,
-738,
-737,
-735,
-735,
-727,
-727,
-727,
-726,
-726,
-724,
-724,
-724,
-723,
```

```
-719,
-719,
-717,
-713,
-713,
-712,
-706,
-705,
-697,
-695,
-689,
-687,
-687,
-682,
-682,
-680,
-678,
-675,
-675,
-674,
-673,
-672,
-666,
-666,
-664,
-662,
-662,
-661,
-661,
-656,
-655,
-654,
-650,
-650,
-649,
-648,
-645,
-640,
-640,
-640,
-640,
-639,
-639,
-636,
-627,
-625,
-624,
-623,
```

```
-616,
-613,
-611,
-610,
-608,
-606,
-602,
-599,
-598,
-597,
-596,
-596,
-596,
-593,
-592,
-589,
-589,
-588,
-585,
-579,
-579,
-579,
-575,
-575,
-570,
-569,
-567,
-565,
-563,
-563,
-561,
-558,
-557,
-555,
-555,
-553,
-553,
-550,
-548,
-545,
-538,
-537,
-536,
-531,
-526,
-523,
-523,
-522,
```

```
-515,
-514,
-512,
-509,
-508,
-506,
-500,
-495,
-495,
-492,
-492,
-484,
-477,
-470,
-467,
-464,
-463,
-463,
-463,
-462,
-452,
-449,
-448,
-447,
-444,
-444,
-441,
-440,
-437,
-426,
-425,
-424,
-422,
-421,
-419,
-417,
-412,
-412,
-411,
-411,
-404,
-403,
-402,
-396,
-395,
-387,
-375,
-375,
```

```
-371,
-368,
-367,
-363,
-359,
-354,
-350,
-348,
-347,
-347,
-346,
-344,
-341,
-336,
-334,
-333,
-333,
-328,
-327,
-324,
-321,
-312,
-298,
-296,
-295,
-294,
-292,
-290,
-289,
-286,
-284,
-284,
-282,
-281,
-278,
-273,
-271,
-269,
-268,
-267,
-262,
-257,
-257,
-249,
-246,
-245,
-239,
-236,
```

```
-236,
-235,
-235,
-233,
-231,
-231,
-230,
-229,
-222,
-222,
-221,
-217,
-215,
-213,
-210,
-205,
-201,
-197,
-196,
-194,
-192,
-186,
-186,
-182,
-181,
-180,
-177,
-171,
-169,
-168,
-164,
-163,
-155,
-152,
-143,
-137,
-134,
-133,
-133,
-131,
-122,
-121,
-119,
-112,
-112,
-112,
-107,
-106,
```

```
-101,
-101,
-94,
-93,
-92,
-90,
-88,
-86,
-79,
-79,
-78,
-73,
-68,
-67,
-63,
-63,
-62,
-61,
-59,
-57,
-50,
-41,
-39,
-39,
-32,
-32,
-29,
-29,
-25,
-23,
-23,
-22,
-19,
-8,
-8,
-6,
-4,
-3,
-2,
4,
4,
5,
13,
14,
15,
16,
22,
27,
```

```
30,
33,
40,
44,
46,
46,
47,
50,
51,
52,
55,
55,
55,
56,
56,
59,
63,
69,
71,
74,
77,
79,
85,
86,
88,
90,
90,
91,
98,
101,
109,
118,
123,
129,
138,
138,
140,
143,
148,
150,
154,
154,
159,
161,
164,
168,
169,
170,
```

```
170,
172,
174,
180,
185,
186,
187,
190,
196,
198,
199,
202,
204,
207,
207,
211,
213,
214,
214,
219,
219,
227,
228,
229,
230,
231,
234,
237,
238,
239,
241,
241,
247,
252,
252,
252,
255,
256,
258,
262,
263,
263,
264,
266,
269,
269,
271,
273,
```

273,
278,
278,
279,
280,
283,
283,
288,
293,
297,
298,
299,
300,
303,
305,
316,
324,
325,
326,
326,
327,
329,
330,
333,
336,
338,
339,
340,
340,
341,
344,
345,
350,
350,
352,
352,
355,
356,
358,
359,
363,
363,
365,
366,
367,
370,
378,
379,

379,
380,
386,
387,
389,
392,
394,
399,
401,
401,
402,
404,
407,
407,
408,
409,
411,
412,
416,
419,
419,
422,
426,
428,
430,
431,
435,
435,
437,
440,
441,
441,
441,
444,
445,
448,
452,
452,
454,
455,
461,
462,
463,
463,
469,
473,
475,
476,

```
482,
488,
488,
494,
494,
496,
496,
499,
499,
501,
502,
503,
507,
508,
515,
518,
519,
520,
521,
522,
522,
527,
532,
533,
534,
540,
543,
545,
556,
559,
559,
562,
566,
574,
574,
579,
581,
592,
592,
593,
597,
600,
601,
601,
605,
607,
610,
616,
```

```
616,
620,
624,
637,
637,
639,
640,
640,
640,
644,
652,
653,
660,
666,
666,
668,
671,
673,
676,
684,
689,
689,
695,
696,
697,
699,
701,
706,
711,
713,
713,
715,
716,
717,
717,
719,
721,
721,
724,
728,
729,
732,
733,
734,
737,
741,
742,
743,
```

```
746,
748,
751,
755,
757,
761,
761,
763,
769,
771,
773,
775,
775,
775,
776,
777,
778,
781,
782,
789,
792,
794,
794,
796,
796,
804,
811,
814,
818,
820,
830,
838,
845,
846,
848,
848,
848,
849,
850,
853,
855,
856,
866,
868,
870,
872,
876,
881,
```

881,
885,
886,
887,
888,
888,
889,
889,
890,
890,
891,
891,
893,
895,
897,
900,
903,
904,
906,
906,
906,
910,
912,
912,
913,
913,
913,
917,
921,
925,
926,
928,
929,
931,
935,
935,
936,
938,
939,
940,
947,
949,
949,
952,
954,
955,
956,
958,

```
 963,
 963,
 964,
 965,
 966,
 968,
 970,
 971,
 980,
 981,
 983,
 983,
 984,
 987,
 988,
 996,
 999]
```

```
In [4]: #--------MERGESORT-------------
        def intercalar(L, L1, L2):
            l1 = len(L1)
            l2 = len(L2)

            if l1==0 or l2==0: #comprobación de parada
                return L+L1+L2


            if L1[0]<=L2[0]:
                a = L1.pop(0)
                L.append(a)
                intercalar(L, L1, L2)

            else:
                b = L2.pop(0)
                L.append(b)
                intercalar(L, L1, L2)
            return L+L1+L2


        def ordenarMS(L):
            l = len(L)
            if l==0 or l==1:
                return L

            L1 = list()
            L2 = list()
            m = floor(l/2)
```

```
            for i in xsrange(0, m):
                a = L.pop(0)
                L1.append(a)

            for i in xsrange(m, l):
                b = L.pop(0)
                L2.append(b)

            L3 = ordenarMS(L1)
            L4 = ordenarMS(L2)

            return intercalar([], L3, L4)
        #---------------------------------------

        print("TIEMPO DE MERGESORT:")
        L = [randint(-1000,1000) for muda in srange(800)]
        %time ordenarMS(L)
```

```
TIEMPO DE MERGESORT:
CPU times: user 128 ms, sys: 28 ms, total: 156 ms
Wall time: 124 ms
```

```
Out[4]: [-999,
         -996,
         -995,
         -994,
         -994,
         -993,
         -993,
         -992,
         -991,
         -991,
         -987,
         -987,
         -986,
         -985,
         -977,
         -976,
         -970,
         -970,
         -969,
         -969,
         -967,
         -966,
         -961,
         -959,
         -953,
```

```
-950,
-944,
-943,
-941,
-936,
-936,
-935,
-935,
-932,
-931,
-928,
-925,
-924,
-922,
-921,
-920,
-920,
-914,
-913,
-910,
-907,
-902,
-901,
-899,
-897,
-896,
-895,
-885,
-884,
-883,
-883,
-880,
-880,
-879,
-878,
-876,
-872,
-871,
-861,
-858,
-855,
-851,
-847,
-846,
-842,
-842,
-842,
-841,
```

```
-841,
-838,
-829,
-829,
-819,
-817,
-815,
-812,
-810,
-809,
-805,
-799,
-796,
-793,
-793,
-790,
-785,
-785,
-783,
-781,
-778,
-775,
-775,
-773,
-773,
-772,
-771,
-766,
-765,
-762,
-760,
-760,
-759,
-758,
-755,
-752,
-752,
-751,
-750,
-749,
-748,
-744,
-739,
-736,
-736,
-734,
-732,
-731,
```

```
-730,
-730,
-725,
-722,
-722,
-721,
-720,
-719,
-718,
-713,
-710,
-709,
-709,
-708,
-706,
-704,
-704,
-701,
-697,
-694,
-686,
-681,
-681,
-673,
-673,
-673,
-672,
-671,
-669,
-668,
-665,
-664,
-661,
-660,
-660,
-660,
-654,
-654,
-651,
-649,
-648,
-648,
-648,
-647,
-637,
-637,
-635,
-632,
```

```
-622,
-621,
-620,
-615,
-609,
-605,
-604,
-595,
-592,
-591,
-590,
-588,
-587,
-585,
-584,
-583,
-576,
-575,
-573,
-564,
-559,
-558,
-546,
-544,
-541,
-536,
-533,
-529,
-527,
-526,
-525,
-521,
-517,
-514,
-514,
-512,
-511,
-510,
-509,
-508,
-508,
-506,
-505,
-502,
-501,
-500,
-499,
-498,
```

```
-493,
-483,
-483,
-480,
-480,
-478,
-476,
-475,
-474,
-472,
-470,
-458,
-456,
-454,
-452,
-447,
-443,
-440,
-439,
-436,
-436,
-436,
-428,
-422,
-422,
-421,
-419,
-419,
-414,
-413,
-412,
-410,
-410,
-408,
-403,
-400,
-398,
-397,
-392,
-389,
-388,
-384,
-383,
-377,
-373,
-373,
-372,
-371,
```

```
-365,
-360,
-358,
-354,
-353,
-348,
-344,
-343,
-343,
-342,
-341,
-337,
-336,
-336,
-336,
-334,
-334,
-331,
-331,
-322,
-321,
-319,
-315,
-311,
-310,
-310,
-308,
-307,
-307,
-306,
-304,
-301,
-300,
-298,
-297,
-297,
-297,
-297,
-296,
-296,
-293,
-291,
-290,
-279,
-279,
-275,
-275,
-267,
```

```
-265,
-257,
-256,
-254,
-250,
-248,
-248,
-247,
-246,
-244,
-242,
-240,
-236,
-235,
-234,
-233,
-231,
-229,
-228,
-225,
-218,
-218,
-216,
-212,
-209,
-208,
-207,
-201,
-198,
-198,
-197,
-196,
-195,
-193,
-192,
-192,
-190,
-189,
-186,
-181,
-180,
-178,
-178,
-175,
-172,
-171,
-167,
-160,
```

```
-155,
-154,
-142,
-138,
-136,
-134,
-129,
-127,
-125,
-125,
-125,
-124,
-123,
-117,
-114,
-111,
-110,
-106,
-105,
-104,
-100,
-99,
-99,
-97,
-95,
-95,
-93,
-93,
-89,
-87,
-87,
-85,
-84,
-83,
-81,
-80,
-80,
-80,
-79,
-75,
-73,
-72,
-72,
-72,
-71,
-69,
-67,
-67,
```

```
-64,
-62,
-58,
-55,
-54,
-51,
-43,
-41,
-34,
-30,
-29,
-28,
-26,
-20,
-15,
-15,
-13,
-5,
-3,
9,
10,
12,
17,
18,
19,
22,
23,
23,
27,
30,
33,
34,
36,
38,
39,
39,
40,
46,
46,
48,
49,
51,
52,
55,
56,
56,
59,
60,
```

```
64,
67,
70,
72,
77,
79,
82,
82,
82,
85,
88,
111,
116,
122,
122,
122,
122,
125,
126,
135,
138,
138,
142,
143,
143,
144,
147,
148,
154,
157,
170,
170,
173,
177,
178,
179,
180,
181,
182,
182,
183,
185,
185,
188,
191,
192,
195,
196,
```

```
197,
197,
199,
199,
201,
202,
203,
205,
205,
207,
207,
209,
212,
214,
214,
214,
222,
225,
227,
230,
230,
232,
235,
240,
240,
244,
244,
248,
251,
255,
256,
257,
260,
262,
263,
263,
266,
268,
270,
271,
271,
273,
273,
275,
283,
285,
288,
288,
```

```
289,
290,
295,
298,
306,
307,
309,
310,
319,
320,
326,
326,
332,
332,
332,
334,
337,
341,
342,
348,
348,
348,
348,
349,
354,
355,
355,
358,
358,
358,
359,
361,
364,
365,
366,
368,
371,
371,
371,
375,
376,
377,
380,
385,
391,
398,
401,
404,
```

405,
407,
408,
412,
413,
417,
418,
427,
427,
427,
428,
429,
434,
435,
438,
448,
448,
451,
452,
453,
461,
469,
474,
475,
477,
479,
479,
481,
484,
490,
494,
502,
503,
505,
508,
512,
515,
516,
521,
526,
529,
530,
531,
532,
534,
538,
539,
539,

```
540,
542,
546,
552,
555,
555,
560,
561,
562,
562,
570,
571,
573,
575,
584,
587,
590,
596,
608,
610,
612,
612,
627,
631,
633,
634,
634,
636,
639,
640,
642,
647,
648,
650,
652,
664,
666,
676,
680,
681,
682,
687,
689,
696,
697,
700,
702,
703,
```

```
707,
712,
717,
717,
719,
720,
720,
720,
722,
724,
727,
738,
741,
743,
746,
750,
751,
755,
760,
761,
766,
768,
771,
772,
775,
781,
782,
792,
793,
797,
797,
803,
807,
807,
807,
811,
811,
813,
818,
821,
822,
823,
823,
828,
834,
834,
840,
842,
```

```
845,
845,
849,
849,
850,
855,
859,
863,
864,
866,
868,
869,
869,
871,
872,
876,
877,
880,
891,
906,
909,
910,
910,
915,
916,
918,
919,
928,
929,
942,
943,
951,
951,
956,
958,
960,
963,
968,
969,
975,
976,
978,
986,
988,
988,
989,
989,
989,
```

```
 992,
 993,
 995,
 997,
 998,
 999,
 1000]
```

EJERCICIO DE ORDENACIÓN UTILIZANDO QUICKSORT

```
In [5]: def ordenarQS(L):
            l = len(L)

            if l<=1:
                return L

            L1 = list()
            L2 = list()
            L3 = list()

            pivot = L[0]

            for i in xsrange(0, l):
                if L[i] < pivot:
                    L1.append(L[i])

                elif L[i] > pivot:
                    L2.append(L[i])

                elif L[i] == pivot:
                    L3.append(L[i])

            M1 = ordenarQS(L1)
            M2 = ordenarQS(L2)

            LR = M1+L3+M2
            return LR



        #PROGRAMA
        L = [randint(-100, 100) for i in xsrange(0, 20)]

        LL = ordenarQS(L)
        print("a) Lista ordenada con 20 números aleatorios entre -100 y 100:")
        print(LL)
        print(" ")
```

```
a) Lista ordenada con 20 números aleatorios entre -100 y 100:
[-56, -54, -50, -45, -17, -10, -10, 2, 10, 18, 23, 26, 48, 52, 62, 66, 69, 78, 93, 98]
```

Comparamos los tiempos de ordenación de la función de sage L.sort() y de Quicksort

```
In [6]: #-----------QUICKSORT-------------
        def ordenarQS(L):
            l = len(L)

            if l<=1:
                return L

            L1 = list()
            L2 = list()
            L3 = list()

            pivot = L[0]

            for i in xsrange(0, l):
                if L[i] < pivot:
                    L1.append(L[i])

                elif L[i] > pivot:
                    L2.append(L[i])

                elif L[i] == pivot:
                    L3.append(L[i])

            M1 = ordenarQS(L1)
            M2 = ordenarQS(L2)

            LR = M1+L3+M2
            return LR
        #---------------------------------

        print("TIEMPO DE QUICKSORT:")
        L = [randint(-1000,1000) for muda in srange(800)]
        %time ordenarQS(L)

TIEMPO DE QUICKSORT:
CPU times: user 44 ms, sys: 0 ns, total: 44 ms
Wall time: 36.2 ms
```

```
Out[6]: [-999,
         -992,
         -988,
         -987,
         -985,
         -984,
         -984,
         -984,
         -982,
         -981,
         -980,
         -979,
         -976,
         -976,
         -972,
         -972,
         -966,
         -965,
         -965,
         -964,
         -963,
         -961,
         -957,
         -957,
         -954,
         -953,
         -953,
         -947,
         -944,
         -941,
         -939,
         -938,
         -937,
         -936,
         -935,
         -932,
         -932,
         -931,
         -927,
         -927,
         -923,
         -921,
         -921,
         -919,
         -918,
         -917,
         -912,
         -911,
```

```
-908,
-908,
-907,
-906,
-906,
-902,
-894,
-888,
-887,
-882,
-881,
-879,
-874,
-872,
-872,
-868,
-868,
-865,
-859,
-859,
-853,
-853,
-852,
-852,
-850,
-849,
-848,
-847,
-845,
-844,
-843,
-841,
-840,
-838,
-836,
-836,
-835,
-835,
-834,
-834,
-833,
-827,
-822,
-822,
-821,
-821,
-816,
-815,
```

```
-813,
-802,
-801,
-800,
-799,
-798,
-797,
-797,
-796,
-795,
-795,
-795,
-792,
-790,
-790,
-789,
-786,
-781,
-781,
-778,
-777,
-774,
-772,
-767,
-765,
-765,
-762,
-761,
-757,
-756,
-752,
-746,
-734,
-734,
-732,
-727,
-724,
-722,
-714,
-710,
-710,
-709,
-692,
-692,
-692,
-689,
-689,
-685,
```

```
-680,
-678,
-675,
-667,
-663,
-663,
-659,
-659,
-656,
-653,
-646,
-638,
-637,
-634,
-630,
-625,
-625,
-622,
-618,
-614,
-604,
-604,
-603,
-603,
-600,
-600,
-599,
-598,
-595,
-593,
-592,
-588,
-584,
-583,
-583,
-579,
-576,
-574,
-573,
-567,
-558,
-557,
-555,
-548,
-547,
-545,
-544,
-544,
```

```
-544,
-543,
-542,
-538,
-519,
-516,
-514,
-513,
-507,
-504,
-497,
-496,
-495,
-495,
-494,
-492,
-491,
-491,
-485,
-476,
-461,
-461,
-458,
-457,
-456,
-456,
-456,
-454,
-454,
-448,
-448,
-446,
-441,
-438,
-435,
-435,
-433,
-429,
-426,
-425,
-423,
-415,
-412,
-411,
-408,
-407,
-405,
-402,
```

```
-401,
-401,
-400,
-397,
-388,
-387,
-386,
-379,
-379,
-378,
-371,
-366,
-366,
-363,
-358,
-358,
-358,
-355,
-349,
-347,
-345,
-344,
-338,
-331,
-330,
-328,
-325,
-323,
-322,
-321,
-321,
-317,
-316,
-316,
-314,
-310,
-310,
-309,
-307,
-305,
-304,
-304,
-299,
-299,
-298,
-292,
-287,
-285,
```

```
-282,
-279,
-276,
-273,
-270,
-269,
-261,
-261,
-259,
-255,
-250,
-247,
-246,
-245,
-245,
-239,
-236,
-234,
-229,
-229,
-227,
-224,
-221,
-218,
-218,
-214,
-212,
-211,
-210,
-208,
-207,
-205,
-201,
-197,
-196,
-195,
-193,
-189,
-188,
-187,
-185,
-184,
-184,
-180,
-180,
-180,
-180,
-175,
```

```
-172,
-172,
-170,
-165,
-163,
-161,
-153,
-149,
-138,
-138,
-134,
-132,
-131,
-131,
-128,
-127,
-125,
-123,
-122,
-118,
-116,
-113,
-112,
-112,
-103,
-102,
-100,
-98,
-97,
-96,
-96,
-90,
-85,
-84,
-84,
-81,
-80,
-76,
-75,
-75,
-72,
-72,
-71,
-70,
-68,
-67,
-59,
-57,
```

```
-53,
-53,
-49,
-49,
-43,
-40,
-38,
-38,
-36,
-33,
-32,
-19,
-17,
-13,
-10,
-9,
-6,
-4,
-3,
3,
8,
8,
9,
10,
25,
27,
31,
32,
37,
38,
39,
41,
41,
45,
46,
50,
55,
56,
56,
60,
61,
62,
67,
72,
73,
76,
82,
82,
```

```
84,
84,
85,
87,
88,
89,
90,
91,
92,
100,
100,
100,
106,
106,
106,
106,
106,
106,
108,
109,
112,
116,
122,
123,
128,
129,
134,
134,
137,
141,
141,
143,
144,
145,
145,
146,
150,
154,
155,
159,
161,
164,
164,
165,
167,
168,
173,
178,
179,
```

```
182,
184,
188,
189,
191,
194,
195,
204,
215,
223,
227,
228,
233,
234,
240,
241,
243,
250,
253,
253,
254,
260,
264,
265,
265,
269,
272,
273,
273,
274,
277,
277,
278,
278,
280,
281,
282,
286,
291,
293,
293,
293,
294,
294,
294,
302,
305,
312,
```

316,
318,
319,
320,
321,
324,
326,
330,
333,
337,
339,
341,
342,
344,
346,
351,
357,
358,
358,
359,
368,
368,
371,
373,
374,
378,
384,
388,
388,
389,
390,
392,
393,
393,
400,
402,
404,
404,
405,
408,
408,
409,
409,
410,
414,
415,
418,
421,

426,
427,
428,
428,
434,
435,
436,
439,
440,
443,
444,
445,
450,
452,
467,
471,
478,
479,
482,
487,
491,
492,
493,
495,
498,
499,
500,
505,
507,
509,
513,
516,
516,
518,
519,
522,
523,
524,
525,
527,
532,
533,
539,
544,
547,
550,
551,
552,

552,
553,
558,
559,
560,
561,
562,
563,
565,
565,
568,
569,
571,
573,
574,
581,
581,
591,
593,
596,
598,
605,
607,
614,
616,
621,
624,
626,
627,
628,
629,
630,
633,
634,
636,
638,
641,
642,
642,
645,
645,
649,
649,
650,
654,
654,
656,
660,

660,
665,
665,
666,
671,
672,
675,
677,
677,
682,
686,
688,
689,
692,
694,
701,
703,
705,
708,
709,
710,
711,
712,
716,
717,
721,
723,
725,
728,
732,
739,
741,
742,
745,
748,
748,
753,
756,
756,
759,
763,
764,
767,
769,
774,
781,
781,
784,

786,
790,
791,
792,
793,
793,
794,
797,
797,
805,
808,
810,
816,
817,
817,
829,
830,
832,
833,
834,
840,
844,
846,
848,
856,
866,
867,
869,
869,
874,
875,
875,
877,
878,
878,
885,
888,
897,
905,
905,
908,
909,
910,
911,
912,
913,
913,
924,

```
 928,
 929,
 930,
 932,
 933,
 940,
 944,
 944,
 948,
 952,
 954,
 956,
 957,
 963,
 964,
 964,
 964,
 971,
 971,
 977,
 983,
 983,
 985,
 986,
 986,
 991,
 992,
 993,
 993,
 994,
 998,
 998]
```

```
In [6]: print("TIEMPO DE L.SORT:")
        L = [randint(-1000,1000) for muda in srange(800)]
        %time L.sort()
```

```
TIEMPO DE L.SORT:
CPU times: user 0 ns, sys: 0 ns, total: 0 ns
Wall time: 389 µs
```