

*

Sistema eficiente de entrega domiciliar

Mateo Ramírez Hernández
Universidad Eafit
Medellín, Colombia
marami26@eafit.edu.co

Juan Camilo Echeverri S.
Universidad Eafit
Medellín, Colombia
jechev60@eafit.edu.co

†

RESUMEN

Este informe se encarga de proporcionar una solución a un popular problema de domicilios, para lograr esto, se buscarán diferentes soluciones e identificaremos cual será la más conveniente para cumplir el objetivo.

1. Introducción

El problema del domiciliario consiste en encontrar el camino más eficiente para completar un circuito entre un conjunto dado de punto sin pasar por el mismo punto dos veces, La importancia de este problema radica en su utilidad en muchos áreas, desde planificación y logística, hasta fabricación de microchips y Secuencia ADN.

En nuestro caso podemos pasar por el mismo lugar más de una vez, por lo tanto, la importancia principal del problema discutido en este informe es que es un paso adelante para lograr una solución eficiente para el domiciliario.

2. Problema

Trabajaremos en una variante del problema del domiciliario, es decir se encontrara el camino mas corto para visitar todos los vértices

3. Problemas similares

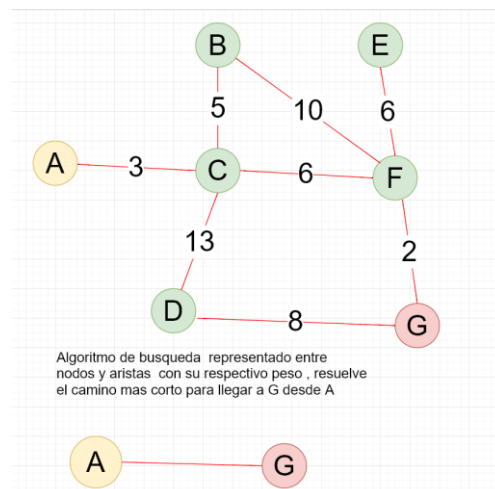
3.1 Una formulación equivalente en términos de Teoría de grafos es: dado una grafo ponderado completo (donde los vértices representan las ciudades, las aristas representan los caminos y los pesos son el costo o las distancias de estoscaminos), encontrar un ciclo de Hamilton con menor peso.

3.2 Las restricciones de retorno a la ciudad de comienzo no cambia la complejidad computacional del problema, vea Problema del camino de Hamiltoniano.

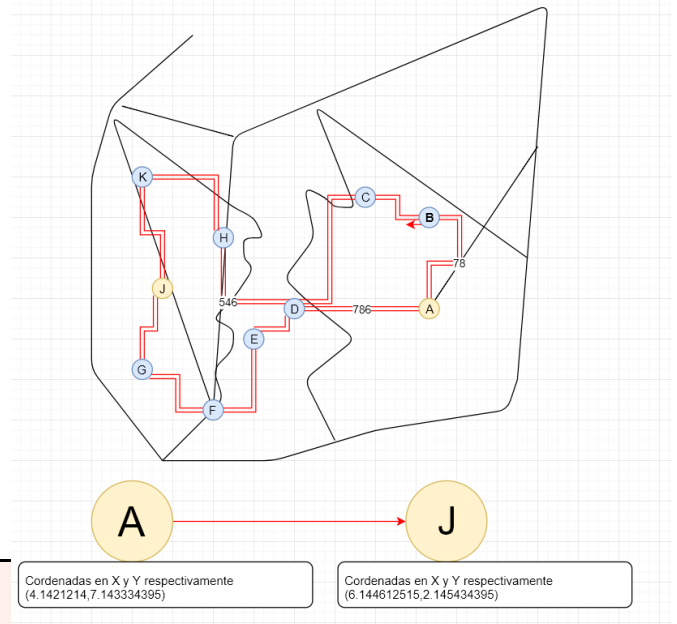
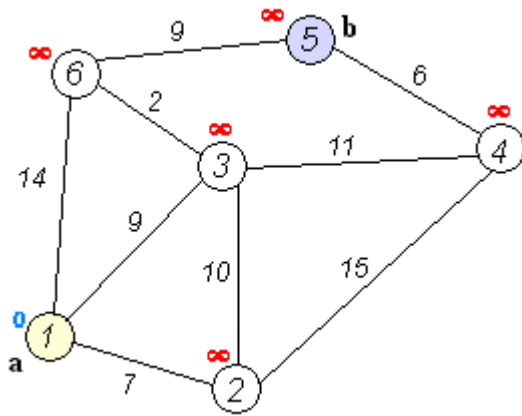
3.3 El problema de ordenamiento secuencial trata con el problema de visitar un conjunto de ciudades donde se tiene en cuenta las relaciones de precedencias entre las ciudades.

3.4 El problema del viajante comprador trata con un comprador que está cargado con un conjunto de productos. Él puede comprar estos productos en varias ciudades, pero tienen diferentes precios y no todas las ciudades ofrecen los mismos productos. El objetivo es encontrar una ruta entre un subconjunto de ciudades, los cuales minimicen el costo total (costo de viaje + costo de la compra) y habilite la compra de todos los productos requeridos.

4. Diseño de la estructura de datos



5. Diseño de las operaciones de la estructura de datos



6. Calculo de las operaciones de la estructura de datos

Método	Complejidad
Dijkstra	$O(m \log v)$
Coordenadas ingresadas	$O(n + 1)$
Complejidad total	$O((n+1)*m \log v)$

Es decir de A a B se aplicó el algoritmo Dijkstra y el camino que se muestra en rojo es el camino mas corto sea redundante para ir de A a B, este algoritmo se emplea de nodo a nodo para resolver el problema de cual seria el camino mas corto de A hasta J

9. Calculo de la complejidad del algoritmo

Mejor caso	$O(m \log v)$
Caso promedio	$O((n+1)*m \log v)$
Peor de los casos	$O((n+1)*m \log v)$

10. Criterios de diseño del Algoritmo

Para el desarrollo del problema, al haber discutido varias opciones, se concluyo que la mejor manera era una solución basada en el algoritmo de Dijkstra, ya que tiene un orden de complejidad y presenta una solución eficiente en cuanto a camino entre dos nodos.

Se plantearon tener otro tipo de soluciones pero fueron descartadas por la ineficiencia que ellas presentan, por ejemplo es el caso de fuerza bruta que tiene una complejidad de $O(n!)$ y de programación dinámica que tiene complejidad de $\hat{O}(n^2 2^n)$.

En Python se uso un diccionario dentro de un diccionario

7. Criterios de diseño de la estructura de datos

Para el desarrollo del problema, al haber discutido varias opciones, se concluyo que la mejor manera era una solución basada en el algoritmo de Dijkstra, ya que tiene un orden de complejidad y presenta una solución eficiente en cuanto a camino entre dos nodos.

Se plantearon tener otro tipo de soluciones pero fueron descartadas por la ineficiencia que ellas presentan, por ejemplo es el caso de fuerza bruta que tiene una complejidad de $O(n!)$ y de programación dinámica que tiene complejidad de $\hat{O}(n^2 2^n)$

8. Diseño del algoritmo

A continuación, se muestra un ejemplo de cómo funciona el algoritmo aplicando Dijkstra de un nodo a otro

```

1 def crear_grafo():
2     grafo = {}
3     return grafo
4
5 def crear_grafo_2d():
6     grafo = {}
7     return grafo
8
9 def crear_grafo_3d():
10    grafo = {}
11    return grafo
12
13 def crear_grafo_n():
14    grafo = {}
15    return grafo
16
17 def crear_grafo_2d():
18    grafo = {}
19    return grafo
20
21 def crear_grafo_3d():
22    grafo = {}
23    return grafo
24
25 def crear_grafo_n():
26    grafo = {}
27    return grafo
28
29 def crear_grafo_2d():
30    grafo = {}
31    return grafo
32
33 def crear_grafo_3d():
34    grafo = {}
35    return grafo
36
37 def crear_grafo_n():
38    grafo = {}
39    return grafo
40
41 def crear_grafo_2d():
42    grafo = {}
43    return grafo
44
45 def crear_grafo_3d():
46    grafo = {}
47    return grafo
48
49 def crear_grafo_n():
50    grafo = {}
51    return grafo
52
53 def crear_grafo_2d():
54    grafo = {}
55    return grafo
56
57 def crear_grafo_3d():
58    grafo = {}
59    return grafo
60
61 def crear_grafo_n():
62    grafo = {}
63    return grafo
64
65 def crear_grafo_2d():
66    grafo = {}
67    return grafo
68
69 def crear_grafo_3d():
70    grafo = {}
71    return grafo
72
73 def crear_grafo_n():
74    grafo = {}
75    return grafo
76
77 def crear_grafo_2d():
78    grafo = {}
79    return grafo
80
81 def crear_grafo_3d():
82    grafo = {}
83    return grafo
84
85 def crear_grafo_n():
86    grafo = {}
87    return grafo
88
89 def crear_grafo_2d():
90    grafo = {}
91    return grafo
92
93 def crear_grafo_3d():
94    grafo = {}
95    return grafo
96
97 def crear_grafo_n():
98    grafo = {}
99    return grafo
100
101 def crear_grafo_2d():
102    grafo = {}
103    return grafo
104
105 def crear_grafo_3d():
106    grafo = {}
107    return grafo
108
109 def crear_grafo_n():
110    grafo = {}
111    return grafo
112
113 def crear_grafo_2d():
114    grafo = {}
115    return grafo
116
117 def crear_grafo_3d():
118    grafo = {}
119    return grafo
120
121 def crear_grafo_n():
122    grafo = {}
123    return grafo
124
125 def crear_grafo_2d():
126    grafo = {}
127    return grafo
128
129 def crear_grafo_3d():
130    grafo = {}
131    return grafo
132
133 def crear_grafo_n():
134    grafo = {}
135    return grafo
136
137 def crear_grafo_2d():
138    grafo = {}
139    return grafo
140
141 def crear_grafo_3d():
142    grafo = {}
143    return grafo
144
145 def crear_grafo_n():
146    grafo = {}
147    return grafo
148
149 def crear_grafo_2d():
150    grafo = {}
151    return grafo
152
153 def crear_grafo_3d():
154    grafo = {}
155    return grafo
156
157 def crear_grafo_n():
158    grafo = {}
159    return grafo
160
161 def crear_grafo_2d():
162    grafo = {}
163    return grafo
164
165 def crear_grafo_3d():
166    grafo = {}
167    return grafo
168
169 def crear_grafo_n():
170    grafo = {}
171    return grafo
172
173 def crear_grafo_2d():
174    grafo = {}
175    return grafo
176
177 def crear_grafo_3d():
178    grafo = {}
179    return grafo
180
181 def crear_grafo_n():
182    grafo = {}
183    return grafo
184
185 def crear_grafo_2d():
186    grafo = {}
187    return grafo
188
189 def crear_grafo_3d():
190    grafo = {}
191    return grafo
192
193 def crear_grafo_n():
194    grafo = {}
195    return grafo
196
197 def crear_grafo_2d():
198    grafo = {}
199    return grafo
200
201 def crear_grafo_3d():
202    grafo = {}
203    return grafo
204
205 def crear_grafo_n():
206    grafo = {}
207    return grafo
208
209 def crear_grafo_2d():
210    grafo = {}
211    return grafo
212
213 def crear_grafo_3d():
214    grafo = {}
215    return grafo
216
217 def crear_grafo_n():
218    grafo = {}
219    return grafo
220
221 def crear_grafo_2d():
222    grafo = {}
223    return grafo
224
225 def crear_grafo_3d():
226    grafo = {}
227    return grafo
228
229 def crear_grafo_n():
230    grafo = {}
231    return grafo
232
233 def crear_grafo_2d():
234    grafo = {}
235    return grafo
236
237 def crear_grafo_3d():
238    grafo = {}
239    return grafo
240
241 def crear_grafo_n():
242    grafo = {}
243    return grafo
244
245 def crear_grafo_2d():
246    grafo = {}
247    return grafo
248
249 def crear_grafo_3d():
250    grafo = {}
251    return grafo
252
253 def crear_grafo_n():
254    grafo = {}
255    return grafo
256
257 def crear_grafo_2d():
258    grafo = {}
259    return grafo
260
261 def crear_grafo_3d():
262    grafo = {}
263    return grafo
264
265 def crear_grafo_n():
266    grafo = {}
267    return grafo
268
269 def crear_grafo_2d():
270    grafo = {}
271    return grafo
272
273 def crear_grafo_3d():
274    grafo = {}
275    return grafo
276
277 def crear_grafo_n():
278    grafo = {}
279    return grafo
280
281 def crear_grafo_2d():
282    grafo = {}
283    return grafo
284
285 def crear_grafo_3d():
286    grafo = {}
287    return grafo
288
289 def crear_grafo_n():
290    grafo = {}
291    return grafo
292
293 def crear_grafo_2d():
294    grafo = {}
295    return grafo
296
297 def crear_grafo_3d():
298    grafo = {}
299    return grafo
300
301 def crear_grafo_n():
302    grafo = {}
303    return grafo
304
305 def crear_grafo_2d():
306    grafo = {}
307    return grafo
308
309 def crear_grafo_3d():
310    grafo = {}
311    return grafo
312
313 def crear_grafo_n():
314    grafo = {}
315    return grafo
316
317 def crear_grafo_2d():
318    grafo = {}
319    return grafo
320
321 def crear_grafo_3d():
322    grafo = {}
323    return grafo
324
325 def crear_grafo_n():
326    grafo = {}
327    return grafo
328
329 def crear_grafo_2d():
330    grafo = {}
331    return grafo
332
333 def crear_grafo_3d():
334    grafo = {}
335    return grafo
336
337 def crear_grafo_n():
338    grafo = {}
339    return grafo
340
341 def crear_grafo_2d():
342    grafo = {}
343    return grafo
344
345 def crear_grafo_3d():
346    grafo = {}
347    return grafo
348
349 def crear_grafo_n():
350    grafo = {}
351    return grafo
352
353 def crear_grafo_2d():
354    grafo = {}
355    return grafo
356
357 def crear_grafo_3d():
358    grafo = {}
359    return grafo
360
361 def crear_grafo_n():
362    grafo = {}
363    return grafo
364
365 def crear_grafo_2d():
366    grafo = {}
367    return grafo
368
369 def crear_grafo_3d():
370    grafo = {}
371    return grafo
372
373 def crear_grafo_n():
374    grafo = {}
375    return grafo
376
377 def crear_grafo_2d():
378    grafo = {}
379    return grafo
380
381 def crear_grafo_3d():
382    grafo = {}
383    return grafo
384
385 def crear_grafo_n():
386    grafo = {}
387    return grafo
388
389 def crear_grafo_2d():
390    grafo = {}
391    return grafo
392
393 def crear_grafo_3d():
394    grafo = {}
395    return grafo
396
397 def crear_grafo_n():
398    grafo = {}
399    return grafo
400
401 def crear_grafo_2d():
402    grafo = {}
403    return grafo
404
405 def crear_grafo_3d():
406    grafo = {}
407    return grafo
408
409 def crear_grafo_n():
410    grafo = {}
411    return grafo
412
413 def crear_grafo_2d():
414    grafo = {}
415    return grafo
416
417 def crear_grafo_3d():
418    grafo = {}
419    return grafo
420
421 def crear_grafo_n():
422    grafo = {}
423    return grafo
424
425 def crear_grafo_2d():
426    grafo = {}
427    return grafo
428
429 def crear_grafo_3d():
430    grafo = {}
431    return grafo
432
433 def crear_grafo_n():
434    grafo = {}
435    return grafo
436
437 def crear_grafo_2d():
438    grafo = {}
439    return grafo
440
441 def crear_grafo_3d():
442    grafo = {}
443    return grafo
444
445 def crear_grafo_n():
446    grafo = {}
447    return grafo
448
449 def crear_grafo_2d():
450    grafo = {}
451    return grafo
452
453 def crear_grafo_3d():
454    grafo = {}
455    return grafo
456
457 def crear_grafo_n():
458    grafo = {}
459    return grafo
460
461 def crear_grafo_2d():
462    grafo = {}
463    return grafo
464
465 def crear_grafo_3d():
466    grafo = {}
467    return grafo
468
469 def crear_grafo_n():
470    grafo = {}
471    return grafo
472
473 def crear_grafo_2d():
474    grafo = {}
475    return grafo
476
477 def crear_grafo_3d():
478    grafo = {}
479    return grafo
480
481 def crear_grafo_n():
482    grafo = {}
483    return grafo
484
485 def crear_grafo_2d():
486    grafo = {}
487    return grafo
488
489 def crear_grafo_3d():
490    grafo = {}
491    return grafo
492
493 def crear_grafo_n():
494    grafo = {}
495    return grafo
496
497 def crear_grafo_2d():
498    grafo = {}
499    return grafo
500
501 def crear_grafo_3d():
502    grafo = {}
503    return grafo
504
505 def crear_grafo_n():
506    grafo = {}
507    return grafo
508
509 def crear_grafo_2d():
510    grafo = {}
511    return grafo
512
513 def crear_grafo_3d():
514    grafo = {}
515    return grafo
516
517 def crear_grafo_n():
518    grafo = {}
519    return grafo
520
521 def crear_grafo_2d():
522    grafo = {}
523    return grafo
524
525 def crear_grafo_3d():
526    grafo = {}
527    return grafo
528
529 def crear_grafo_n():
530    grafo = {}
531    return grafo
532
533 def crear_grafo_2d():
534    grafo = {}
535    return grafo
536
537 def crear_grafo_3d():
538    grafo = {}
539    return grafo
540
541 def crear_grafo_n():
542    grafo = {}
543    return grafo
544
545 def crear_grafo_2d():
546    grafo = {}
547    return grafo
548
549 def crear_grafo_3d():
550    grafo = {}
551    return grafo
552
553 def crear_grafo_n():
554    grafo = {}
555    return grafo
556
557 def crear_grafo_2d():
558    grafo = {}
559    return grafo
560
561 def crear_grafo_3d():
562    grafo = {}
563    return grafo
564
565 def crear_grafo_n():
566    grafo = {}
567    return grafo
568
569 def crear_grafo_2d():
570    grafo = {}
571    return grafo
572
573 def crear_grafo_3d():
574    grafo = {}
575    return grafo
576
577 def crear_grafo_n():
578    grafo = {}
579    return grafo
580
581 def crear_grafo_2d():
582    grafo = {}
583    return grafo
584
585 def crear_grafo_3d():
586    grafo = {}
587    return grafo
588
589 def crear_grafo_n():
590    grafo = {}
591    return grafo
592
593 def crear_grafo_2d():
594    grafo = {}
595    return grafo
596
597 def crear_grafo_3d():
598    grafo = {}
599    return grafo
600
601 def crear_grafo_n():
602    grafo = {}
603    return grafo
604
605 def crear_grafo_2d():
606    grafo = {}
607    return grafo
608
609 def crear_grafo_3d():
610    grafo = {}
611    return grafo
612
613 def crear_grafo_n():
614    grafo = {}
615    return grafo
616
617 def crear_grafo_2d():
618    grafo = {}
619    return grafo
620
621 def crear_grafo_3d():
622    grafo = {}
623    return grafo
624
625 def crear_grafo_n():
626    grafo = {}
627    return grafo
628
629 def crear_grafo_2d():
630    grafo = {}
631    return grafo
632
633 def crear_grafo_3d():
634    grafo = {}
635    return grafo
636
637 def crear_grafo_n():
638    grafo = {}
639    return grafo
640
641 def crear_grafo_2d():
642    grafo = {}
643    return grafo
644
645 def crear_grafo_3d():
646    grafo = {}
647    return grafo
648
649 def crear_grafo_n():
650    grafo = {}
651    return grafo
652
653 def crear_grafo_2d():
654    grafo = {}
655    return grafo
656
657 def crear_grafo_3d():
658    grafo = {}
659    return grafo
660
661 def crear_grafo_n():
662    grafo = {}
663    return grafo
664
665 def crear_grafo_2d():
666    grafo = {}
667    return grafo
668
669 def crear_grafo_3d():
670    grafo = {}
671    return grafo
672
673 def crear_grafo_n():
674    grafo = {}
675    return grafo
676
677 def crear_grafo_2d():
678    grafo = {}
679    return grafo
680
681 def crear_grafo_3d():
682    grafo = {}
683    return grafo
684
685 def crear_grafo_n():
686    grafo = {}
687    return grafo
688
689 def crear_grafo_2d():
690    grafo = {}
691    return grafo
692
693 def crear_grafo_3d():
694    grafo = {}
695    return grafo
696
697 def crear_grafo_n():
698    grafo = {}
699    return grafo
700
701 def crear_grafo_2d():
702    grafo = {}
703    return grafo
704
705 def crear_grafo_3d():
706    grafo = {}
707    return grafo
708
709 def crear_grafo_n():
710    grafo = {}
711    return grafo
712
713 def crear_grafo_2d():
714    grafo = {}
715    return grafo
716
717 def crear_grafo_3d():
718    grafo = {}
719    return grafo
720
721 def crear_grafo_n():
722    grafo = {}
723    return grafo
724
725 def crear_grafo_2d():
726    grafo = {}
727    return grafo
728
729 def crear_grafo_3d():
730    grafo = {}
731    return grafo
732
733 def crear_grafo_n():
734    grafo = {}
735    return grafo
736
737 def crear_grafo_2d():
738    grafo = {}
739    return grafo
740
741 def crear_grafo_3d():
742    grafo = {}
743    return grafo
744
745 def crear_grafo_n():
746    grafo = {}
747    return grafo
748
749 def crear_grafo_2d():
750    grafo = {}
751    return grafo
752
753 def crear_grafo_3d():
754    grafo = {}
755    return grafo
756
757 def crear_grafo_n():
758    grafo = {}
759    return grafo
760
761 def crear_grafo_2d():
762    grafo = {}
763    return grafo
764
765 def crear_grafo_3d():
766    grafo = {}
767    return grafo
768
769 def crear_grafo_n():
770    grafo = {}
771    return grafo
772
773 def crear_grafo_2d():
774    grafo = {}
775    return grafo
776
777 def crear_grafo_3d():
778    grafo = {}
779    return grafo
780
781 def crear_grafo_n():
782    grafo = {}
783    return grafo
784
785 def crear_grafo_2d():
786    grafo = {}
787    return grafo
788
789 def crear_grafo_3d():
790    grafo = {}
791    return grafo
792
793 def crear_grafo_n():
794    grafo = {}
795    return grafo
796
797 def crear_grafo_2d():
798    grafo = {}
799    return grafo
800
801 def crear_grafo_3d():
802    grafo = {}
803    return grafo
804
805 def crear_grafo_n():
806    grafo = {}
807    return grafo
808
809 def crear_grafo_2d():
810    grafo = {}
811    return grafo
812
813 def crear_grafo_3d():
814    grafo = {}
815    return grafo
816
817 def crear_grafo_n():
818    grafo = {}
819    return grafo
820
821 def crear_grafo_2d():
822    grafo = {}
823    return grafo
824
825 def crear_grafo_3d():
826    grafo = {}
827    return grafo
828
829 def crear_grafo_n():
830    grafo = {}
831    return grafo
832
833 def crear_grafo_2d():
834    grafo = {}
835    return grafo
836
837 def crear_grafo_3d():
838    grafo = {}
839    return grafo
840
841 def crear_grafo_n():
842    grafo = {}
843    return grafo
844
845 def crear_grafo_2d():
846    grafo = {}
847    return grafo
848
849 def crear_grafo_3d():
850    grafo = {}
851    return grafo
852
853 def crear_grafo_n():
854    grafo = {}
855    return grafo
856
857 def crear_grafo_2d():
858    grafo = {}
859    return grafo
860
861 def crear_grafo_3d():
862    grafo = {}
863    return grafo
864
865 def crear_grafo_n():
866    grafo = {}
867    return grafo
868
869 def crear_grafo_2d():
870    grafo = {}
871    return grafo
872
873 def crear_grafo_3d():
874    grafo = {}
875    return grafo
876
877 def crear_grafo_n():
878    grafo = {}
879    return grafo
880
881 def crear_grafo_2d():
882    grafo = {}
883    return grafo
884
885 def crear_grafo_3d():
886    grafo = {}
887    return grafo
888
889 def crear_grafo_n():
890    grafo = {}
891    return grafo
892
893 def crear_grafo_2d():
894    grafo = {}
895    return grafo
896
897 def crear_grafo_3d():
898    grafo = {}
899    return grafo
900
901 def crear_grafo_n():
902    grafo = {}
903    return grafo
904
905 def crear_grafo_2d():
906    grafo = {}
907    return grafo
908
909 def crear_grafo_3d():
910    grafo = {}
911    return grafo
912
913 def crear_grafo_n():
914    grafo = {}
915    return grafo
916
917 def crear_grafo_2d():
918    grafo = {}
919    return grafo
920
921 def crear_grafo_3d():
922    grafo = {}
923    return grafo
924
925 def crear_grafo_n():
926    grafo = {}
927    return grafo
928
929 def crear_grafo_2d():
930    grafo = {}
931    return grafo
932
933 def crear_grafo_3d():
934    grafo = {}
935    return grafo
936
937 def crear_grafo_n():
938    grafo = {}
939    return grafo
940
941 def crear_grafo_2d():
942    grafo = {}
943    return grafo
944
945 def crear_grafo_3d():
946    grafo = {}
947    return grafo
948
949 def crear_grafo_n():
950    grafo = {}
951    return grafo
952
953 def crear_grafo_2d():
954    grafo = {}
955    return grafo
956
957 def crear_grafo_3d():
958    grafo = {}
959    return grafo
960
961 def crear_grafo_n():
962    grafo = {}
963    return grafo
964
965 def crear_grafo_2d():
966    grafo = {}
967    return grafo
968
969 def crear_grafo_3d():
970    grafo = {}
971    return grafo
972
973 def crear_grafo_n():
974    grafo = {}
975    return grafo
976
977 def crear_grafo_2d():
978    grafo = {}
979    return grafo
980
981 def crear_grafo_3d():
982    grafo = {}
983    return grafo
984
985 def crear_grafo_n():
986    grafo = {}
987    return grafo
988
989 def crear_grafo_2d():
990    grafo = {}
991    return grafo
992
993 def crear_grafo_3d():
994    grafo = {}
995    return grafo
996
997 def crear_grafo_n():
998    grafo = {}
999    return grafo
1000

```

11. Resultados obtenidos

	Conjunto de datos 1 (3 coordenadas)	Conjunto de datos 2 (4 coordenadas)	Conjunto de datos 3 (6 coordenadas)	Conjunto de datos n (mas de 8 Coordenadas)
Mejor caso	2.0713 seg	2.7588 seg	48.248 seg	87.631 seg
Caso promedio	2.07945 seg	2.7643 seg	48.517 seg	90.752 seg
Peor caso	2.08245 seg	2.7767 seg	50.973 seg	118.730 seg

	Conjunto de datos 1 (3 coordenadas)	Conjunto de datos 2 (4 coordenadas)	Conjunto de datos 3 (6 coordenadas)	Conjunto de datos n (mas de 8 Coordenadas)
Mejor caso (bytes)		183427072	184217600	184545280
Caso promedio (bytes)		183537664	186326178	187635212
Peor caso (bytes)		183541760	190021213	190127262

12. Conclusiones

- El resultado fue lo que se esperaba, tener una eficiencia a la hora de resolver el problema
- Al realizar las pruebas de ejecución respecto al tiempo pudimos observar que por mayor que sean las coordenadas, si están muy cercas entre si , el tiempo de ejecución es muy bajo respecto a coordenadas de mayor distancia
- Al realizar la practica nos encontramos con problemas manejables como por ejemplo la ineficiencia del algoritmo en algunos casos