الف) برای این کار می توان از داده ساختار لینک لیست دو طرفه استفاده کرد. چون برای هر داده، به داده ی قبلی و بعدیش دسترسی داریم.

ب) الگوریتمم به این صورته که، یه dll از خانه های موجود، بر حسب ارتفاعشان، به عنوان ورودی می گیرم. از خانه ی دوم شروع به مقایسه می کنم. (چون خانه ی اول در یک سمتش خانه ای وجود ندارد).

اگر خانه ی مورد نظر، از خانه ی بعد از خودش و از خانه ی بعد از خودش ارتفاعش بیشتر بود، یک عدد به شمارنده اضافه می کنم. ادامه می دهم تا برسم به خونه ی یکی مونده به آخر. خانه ی آخری رو هم مقایسه نمی کنیم چون بعد ازش خانه ای نیست.

ج) کدش با عنوان Q1-2 آپلود شده.

د) هزینه ی زمانی با عنوان Q1-3 آپلود شده.

در هر سه حالت، مرتبه ی زمانی این الگوریتم از O(n) می باشد، چون در هر حالت این الگوریتم لیست مورد نظر را یک بار پیمایش می کند.

ه) پیچیدگی مکانی: در ساختمان داده ی لینک لیست دو طرفه، در هر نود، همراه با داده، آدرس داده ی قبلی و آدرس داده ی بعدی هم ذخیره می شود. پس هزینه ی مکانی یک و نیم برابر میشه.