به نام خداوند

تمرین شماره ۷ رباتیک

عرفان اعتصامی **۹۶۱۰۶۲۱۴**

مقدمه) در حل این تمرین، از مطالب موجود در جزوه‌ی جلسات ۱۹ و ۲۰ استفاده شده است. برنامه‌ی $Run.m$ مسئول اجرای برنامه‌ی مربوط به بخش اجباری تمرین بوده و برنامه‌ی $Run\_Bonus.m$ وظیفه‌ی حل بخش امتیازی تمرین را برعهده دارد.

نکته‌ی ۱: فرض می‌شود که ورودی برنامه کاملاً مطابق با فایل $input.txt$ می‌باشد؛ یعنی به عنوان مثال، طبق صورت تمرین، فرض می‌شود که در ورودی بخش امتیازی، اولین رأس ربات برابر با همان نقطه‌ی شروع بوده و همچنین مختصات ربات به گونه‌ای متوالی مرتب شده و وارد برنامه می‌شوند که هندسه‌ی ربات به صورت یک چندضلعی محدب باشد.

نکته‌ی ۲: در بخش اجباری، فرض می‌شود که نقاط شروع و پایان می‌توانند در هر جایی روی موانع باشند.

نکته‌ی ۳: در برنامه‌ی نوشته‌شده برای بخش امتیازی، همان اولین رأس ربات (یا همان نقطه‌ی شروع) به عنوان نقطه‌ی نماینده‌ی ربات انتخاب شده و پوش‌های محدب، متناظر با این نقطه از ربات رسم می‌شوند؛ در نتیجه، عملاً نقطه‌ی شروع نمی‌تواند داخل یا روی یک پوش محدب باشد.

نکته‌ی ۴: در قسمت امتیازی، پوش‌های محدب بدون استفاده از تابع $convhull$ نرم‌افزار $MATLAB$ و به کمک الگوریتم موجود در جزوه‌ی جلسه‌ی ۲۰ تولید می‌شوند. برای عدم تداخل احتمالی با نام تابع $built-in$ نرم‌افزار $MATLAB$، پوش‌های محدب تولیدشده برای هر مانع در یک آرایه‌ی سلولی به نام $cnvhull$ ذخیره می‌شوند.

نکته‌ی ۵: اگر نقطه‌ی پایانی داخل یک پوش محدب باشد (نه روی اضلاع آن)، برنامه به روند اجرای خود ادامه می‌دهد؛ منتها در انتها، پیامی به شرح زیر در خروجی چاپ می‌شود که بیانگر عدم امکان رسیدن ربات به نقطه‌ی مقصد در واقعیت می‌باشد.

$Unreachable\ in\ Reality: Xf\ is\ in\ Convexhull\ \#$

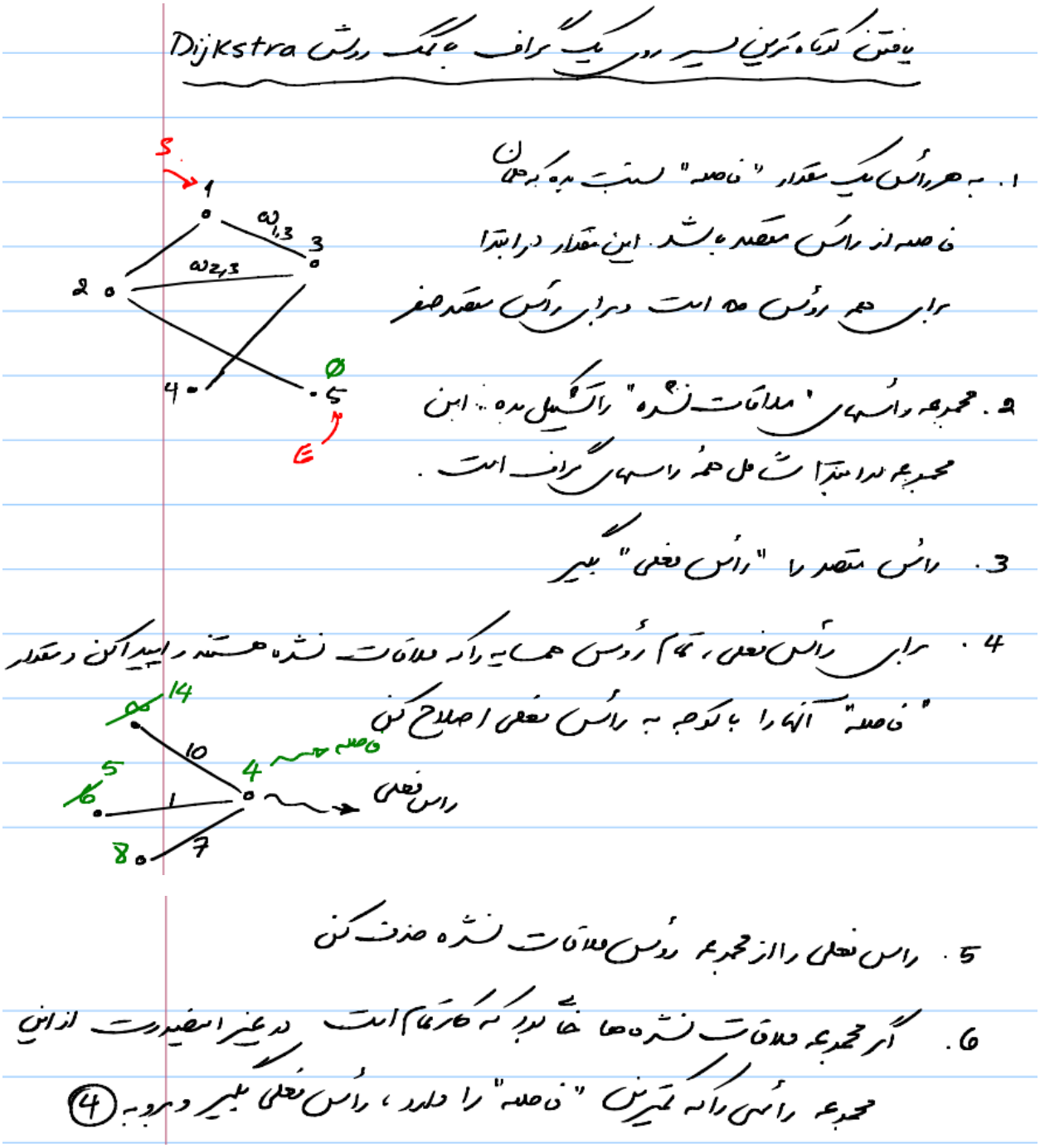
نکته‌ی ۶: با توجه به کسر نمره بابت تأخیر زیاد در تحویل تمرین، این فرصت فراهم نشد که نکته‌ی ۵ و همچنین حالتی که دو پوش محدب با یک‌دیگر برخورد می‌کنند، به صورت عمیق بررسی شوند.

نکته‌ی ۷: برای توضیح برنامه‌ی نوشته‌شده، مطالبی به صورت $comment$ در فایل برنامه درج شده‌اند. شماره‌های مشاهده‌شده در برخی از این قسمت‌ها، برای تطبیق هر بخش با ترتیب مراحل پیاده‌سازی الگوریتم‌ها طبق جزوه هستند.

نکته‌ی ۸: هشدارهای نارنجی‌رنگ چاپ‌شده در خروجی توسط $MATLAB$، حاکی از عدم تعریف ماترس معکوس در عملیات تشخیص برخورد یال‌های گراف با موانع می‌باشد و خللی در اجرای برنامه ایجاد نمی‌کنند.

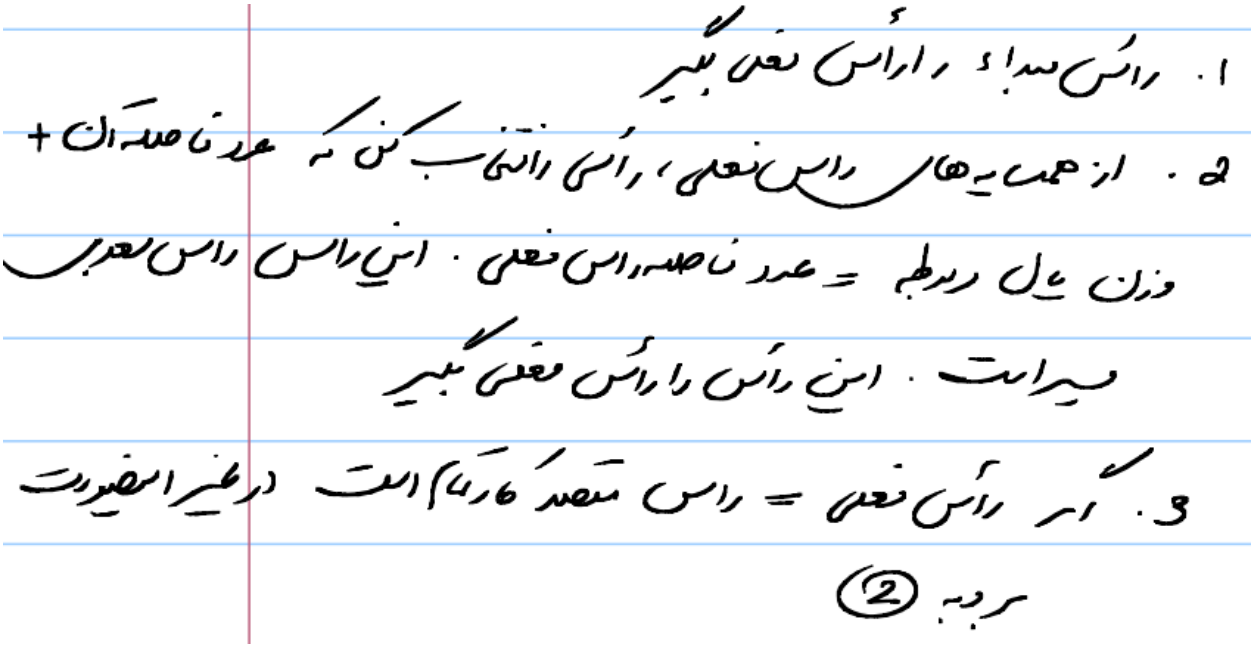
نکته‌ی ۹: در پایان هر کدام از برنامه‌ها، بخش $Plot$ وجود دارد. در حالت اجباری، زیربخش $Plot\ Edges$ (رسم تمام یال‌های گراف) و در حالت امتیازی، زیربخش $Plot\ Edges$ و $Plot\ Robot$ (رسم ربات به صورت mضلعی) به صورت $comment$شده درج شده‌اند تا از شلوغ‌شدن نمودارها جلوگیری شود.

الگوریتم دایکسترا) برای پیاده‌سازی این الگوریتم، از مطالب موجود در جزوه‌ی جلسه‌ی ۱۹ استفاده شده است. شرح مرحله‌ی ۱ در تصویر زیر مشاهده می‌شود.



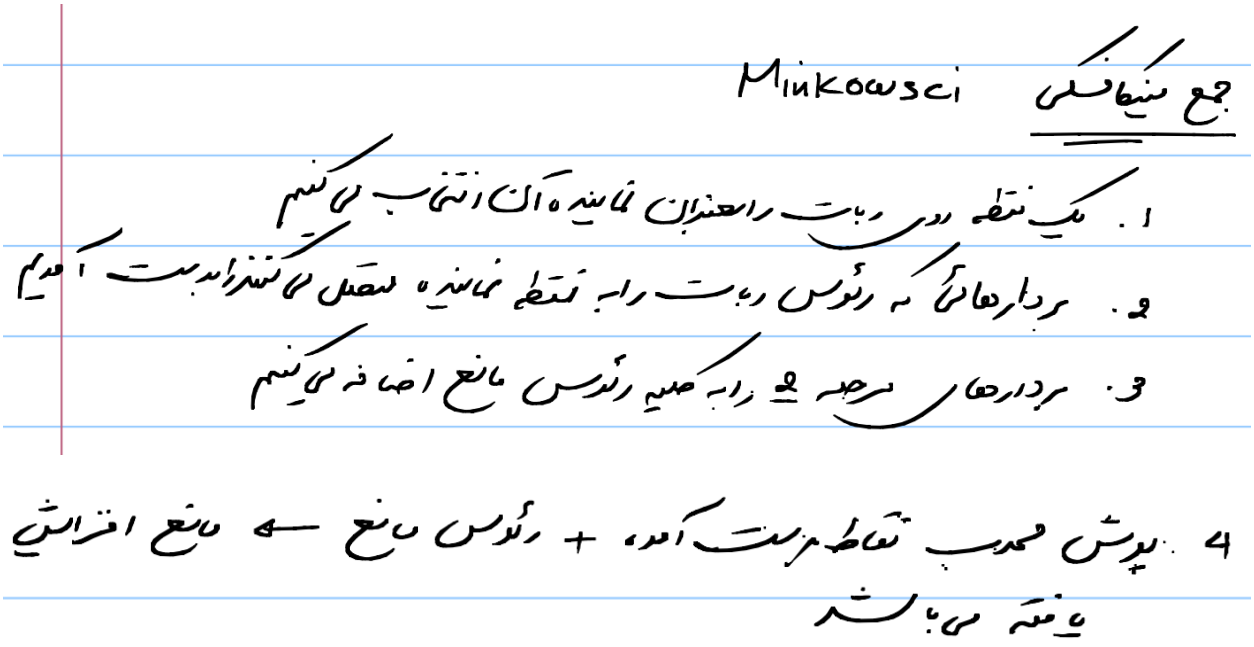
شکل ۱) الگوریتم دایکسترا – مرحله‌ی ۱

شرح مرحله‌ی ۲ از این الگوریتم در شکل ۲ قابل مشاهده است.

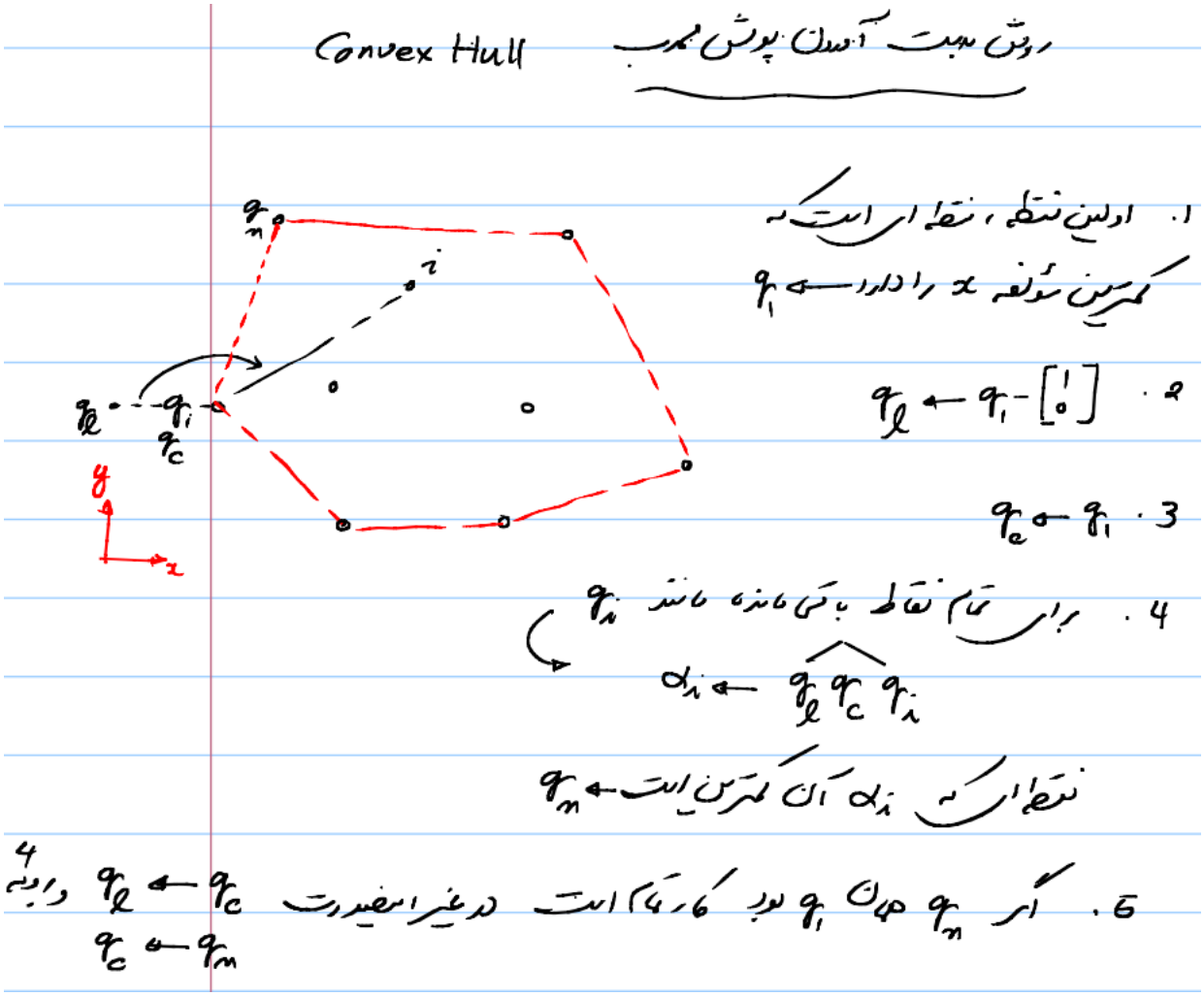


شکل ۲) الگوریتم دایکسترا – مرحله‌ی ۲

الگوریتم ساخت مانع رشدیافته و پوش محدب) برای پیاده‌سازی این الگوریتم از مطالب موجود در جزوه‌ی جلسه‌ی ۲۰ استفاده شده است.

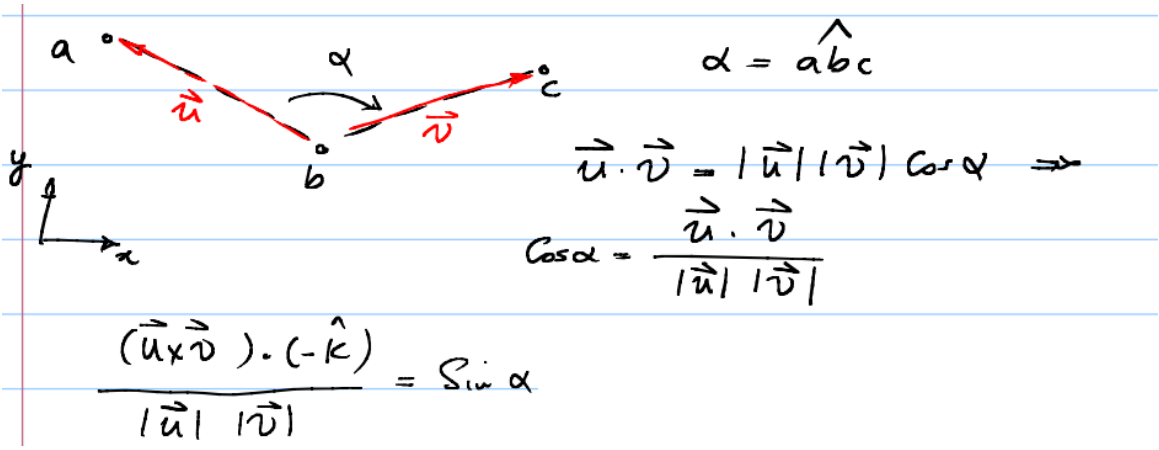


شکل ۳) الگوریتم جمع مینکافسکی



شکل ۴) الگوریتم تولید پوش محدب

برای به دست آوردن زاویه‌ی جهت‌دار نیز از روابط زیر استفاده می‌کنیم.



شکل ۵) الگوریتم محاسبه‌ی زاویه‌ی جهت‌دار

بخش اجباری) فایل $Run.m$ مسئول اجرای این بخش می‌باشد. علاوه بر $comment$های موجود در فایل برنامه، بعضی از بخش‌ها را که نیازمند توضیح بیشتر هستند، در این قسمت شرح می‌دهیم.

خط ۲۲: از آن جا که مانع‌ها به صورت پاره‌خط بوده و هر پاره‌خط شامل دو رأس می‌باشد، مجموع رئوس گراف برابر با دو برابر تعداد موانع به علاوه‌ی دو رأس آغاز و انتهای مسیر می‌باشد.

خط ۲۳: مجموعه‌ی رئوس در متغیر $Nodes$ ذخیره می‌شود. ستون ۱ این متغیر حاوی شماره، ستون ۲ حاوی مختصه‌ی x، ستون ۳ حاوی مختصه‌ی y و ستون ۴ حاوی عدد فاصله‌ی هر رأس می‌باشد.

خط ۴۲: حداکثر تعداد یال‌های ممکن گراف برابر با انتخاب ۲ رأس از مجموعه‌ی رئوس فرض می‌شود. مجموعه‌ی یال‌ها در متغیر $Edges$ ذخیره می‌شود. ستون ۱ این متغیر حاوی شماره‌ی هر یال، ستون ۲ حاوی کوچک‌ترین شماره‌ی رأس مربوط به هر یال، ستون ۳ حاوی بزرگ‌ترین شماره‌ی رأس مربوط به هر یال و ستون ۴ حاوی وزن هر یال گراف می‌باشد.

خطوط ۴۴ تا ۷۸: در این خطوط یال‌هایی از گراف انتخاب می‌شوند که با هیچ‌کدام از موانع برخورد نداشته باشند (بدیهی است که برخورد با رئوس هر مانع ایرادی ندارد). برای بررسی عدم برخورد، از الگوریتم زیر (مشابه مطلب موجود در جزوه‌ی جلسه‌ی ۱۸) استفاده شده است.

اگر A و C به ترتیب مختصات سر اول و دوم یک مانع و همچنین S و T مختصات دو رأس از گراف باشند، با نوشتن معادله‌ی خطوط به صورت پارامتری خواهیم داشت:

معادله‌ی خط واصل بین A و C: $(A-C)\alpha + C$

معادله‌ی خط واصل بین S و T: $(S-T)\beta + T$

با برابر قرار دادن معادله‌ی دو خط، خواهیم داشت:

$$(A-C)\alpha + C = (S-T)\beta + T \rightarrow [(A-C) \quad -(S-T)]\begin{bmatrix}\alpha \\ \beta\end{bmatrix} = (T-C) \rightarrow$$

$$\begin{bmatrix}\alpha \\ \beta\end{bmatrix} = [(A-C) \quad -(S-T)]^{-1}(T-C)$$

با به دست آمدن مقدار α و β کافی است شرط زیر را بررسی کنیم.

$if\ (\alpha > 0\ \&\ a < 1)\ or\ (\beta > 0\ \&\ \beta < 1) \rightarrow$ برخورد یال گراف با مانع

خطوط ۵۴ تا ۵۶: این خطوط بررسی می‌کنند که در صورتی که دو رأس انتخاب‌شده از گراف، منطبق با رئوس موانع باشند (موازی بودن این یال با مانع و منطبق بودن بر آن)، برای کاهش زمان محاسباتی، حلقه به تکرار بعدی برود؛ زیرا هر کدام از موانع، خود یکی از یال‌های گراف هستند. علت استفاده از شرط موازی‌بودن، جلوگیری از ایجاد هشدارهای نارنجی‌رنگ $MATLAB$ مبنی بر عدم تعریف ماتریس معکوس در عملیات فوق می‌باشد.

خطوط ۶۱ و ۶۲: علت استفاده از $round$ جلوگیری از خطای ایجادشده در برخی از مراحل به خاطر دقت محاسباتی $MATLAB$ می‌باشد.

خطوط ۹۷ تا ۱۰۴: گاهی پیش می‌آید که اشتراک همسایه‌های یک رأس با مجموعه رئوس ملاقات‌نشده، تهی می‌باشد. در این صورت، برای جلوگیری از توقف برنامه، از آن‌جا که رئوس به ترتیب شماره در مجموعه‌ی ملاقات‌نشده تعریف شده‌اند، برنامه به این صورت عمل می‌کند که رأس مشکل‌ساز را از مجموعه‌ی ملاقات‌نشده حذف کرده و آخرین رأس موجود در مجموعه‌ی ملاقات‌نشده را به عنوان رأس فعلی در نظر می‌گیرد. این رئوس مشکل‌ساز تحت نام $unreachable\ nodes$ در مباحث مربوط به این الگوریتم شناخته می‌شوند.

بخش امتیازی) فایل $Run\_Bonus.m$ مسئول اجرای این بخش می‌باشد. علاوه بر $comment$های موجود در فایل برنامه، بعضی از بخش‌ها را که نیازمند توضیح بیشتر هستند، در این قسمت شرح می‌دهیم. بخش زیادی از این برنامه، مانند نحوه‌ی پیاده‌سازی الگوریتم دایکسترا، مشابه فایل $Run.m$ (بخش اجباری) می‌باشد.

خطوط ۲۹ تا ۳۹: گام اول و دوم الگوریتم جمع مینکافسکی در این خطوط پیاده شده است.

خط ۴۲: از آن‌جا که موانع اولیه به صورت پاره‌خط بوده و هر پاره‌خط دو رأس دارد، مشخص می‌شود که پس از افزودن بردارهای مربوط به هندسه‌ی ربات (mضلعی) به هر رأس، m رأس جدید (با احتساب رئوس موانع اولیه) ایجاد می‌شود؛ پس تعداد کل رئوس برابر با $2 \times N \times m$ خواهد بود. مختصات x و y هر رأس در متغیر obs ذخیره می‌شود.

خط ۶۲: فرض می‌شود هر پوش محدب (با احتساب افزودن اولین رأس در ردیف آخر برای راحتی محاسبات در ادامه‌ی برنامه) به دلیل اشاره شده برای خط ۴۲، حداکثر برابر $2 \times m$ رأس باشد. به علاوه، متغیر $cnvhull$ به صورت آرایه‌ی سلولی تعریف شده، چون ممکن است تمام پوش‌ها، تعداد اضلاع برابری نداشته باشند.

خطوط ۸۰ تا ۹۴: این خطوط مسئول محاسبه‌ی زاویه‌ی جهت‌دار مطابق شکل ۵ هستند. صرفاً توجه شود که اگر یک زاویه‌ی ساعت‌گرد (مقدار منفی) در $-\hat{k}$ ضرب داخلی شود، نتیجه‌ی حاصل مقداری مثبت خواهد داشت. شروط نوشته‌شده در خطوط ۸۸ تا ۹۴ علاوه بر توجه به این نکته، رئوس پوش محدب را به ترتیب و به صورت ساعت‌گرد در کنار یک‌دیگر تعریف می‌کنند.

خطوط ۱۳۷ تا ۱۶۰: در این خطوط، تمام اقطار پوش‌های محدب تولیدشده و شماره‌ی دو رأس هر قطر، به ترتیب افزایش شماره‌ی رئوس، در متغیر $diagonals$ ذخیره خواهند شد.

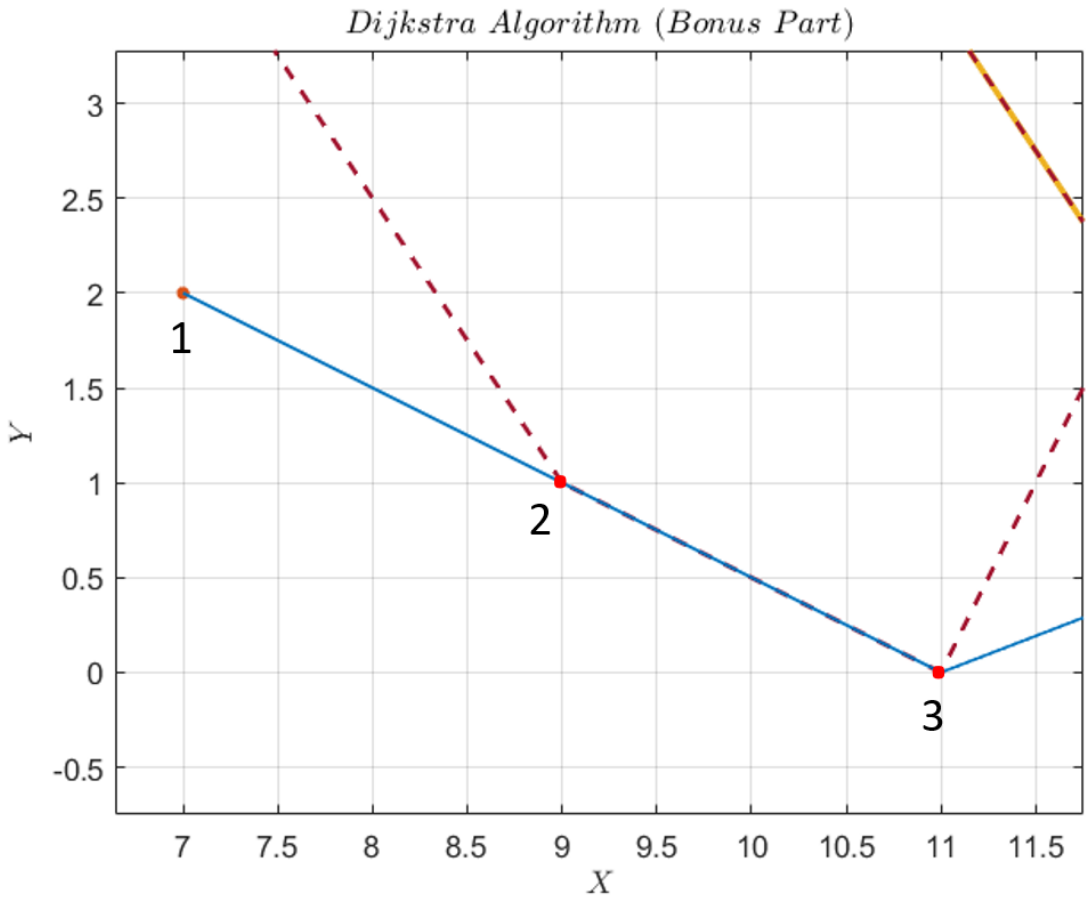
خطوط ۱۶۲ تا ۱۷۰: نکته‌ی ۵ در صفحه‌ی ۱ در این خطوط پیاده‌سازی می‌شود.

خطوط ۱۷۸ تا ۱۸۳: این خطوط اقطار پوش‌های محدب را از مجموعه یال‌های گراف حذف می‌کنند؛ زیرا ربات نمی‌تواند از مانع و در نتیجه یال‌های داخلی بین رئوس آن (همان قطرها) عبور کند.

خطوط ۱۸۸ تا ۲۱۹: این خطوط به کمک دو تابع $polyxpoly$ (برخورد دو چندضلعی) و $inpolygon$ (بررسی وجود مجموعه‌ای از نقاط در داخل یا روی یک چندضلعی) از نرم افزار $MATLAB$، دو شرط زیر را در خطوط ۱۹۶ تا ۲۱۰ و خطوط ۲۱۱ تا ۲۱۴ بررسی می‌کنند.

خطوط ۱۹۶ تا ۲۱۰: این خطوط به بررسی وجود نقطه‌ی پایانی روی یکی از اضلاع پوش‌های محدب می‌پردازند. بدیهی است اگر نقطه‌ی پایانی روی یکی از اضلاع باشد، تنها می‌توان از دو رأس آن ضلع، یالی به سمت آن تعریف کرد. به عبارت دیگر از باقی رئوس آن پوش، نمی‌توان به سمت نقطه‌ی پایانی حرکت کرد، زیرا این کار به معنی ورود نقطه‌ی نماینده‌ی ربات به مانع رشدیافته می‌باشد. خطوط ۲۰۳ تا ۲۰۶ به کمک تعریف ضرب خارجی، به بررسی روی یک یال بودن (روی یک خط بودن) نقطه‌ی پایانی و دو رأس ضلعی که نقطه‌ی پایانی بر روی آن قرار گرفته است می‌پردازند.

خطوط ۲۱۱ تا ۲۱۴: گاهی پیش می‌آید که یک یال گراف (به عنوان مثال ۳-۱) مطابق شکل زیر موازی با یکی از اضلاع پوش محدب (۳-۲) می‌شود. در این حالت باید بررسی کرد که این یال، پوش محدب را قطع نکند. برای تفکیک این یال از ضلع پوش محدب، توجه می‌شود که یال گفته‌شده، پوش محدب را در بیش از ۱ نقطه قطع می‌کند، اما هر دو رأس آن یال بر روی پوش قرار ندارند.

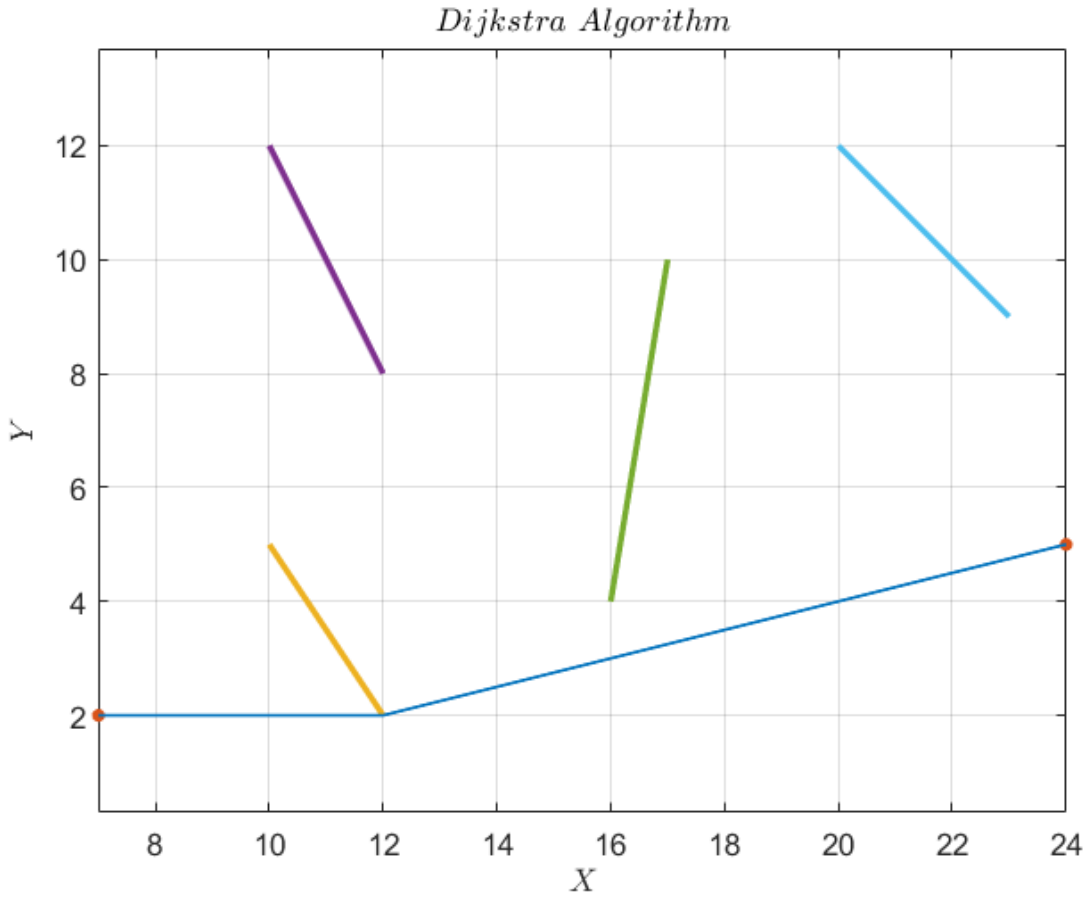


شکل ۶) توضیح خطوط ۲۱۱ تا ۲۱۴

آزمودن برنامه‌ی بخش اجباری)

ورودی ۱:

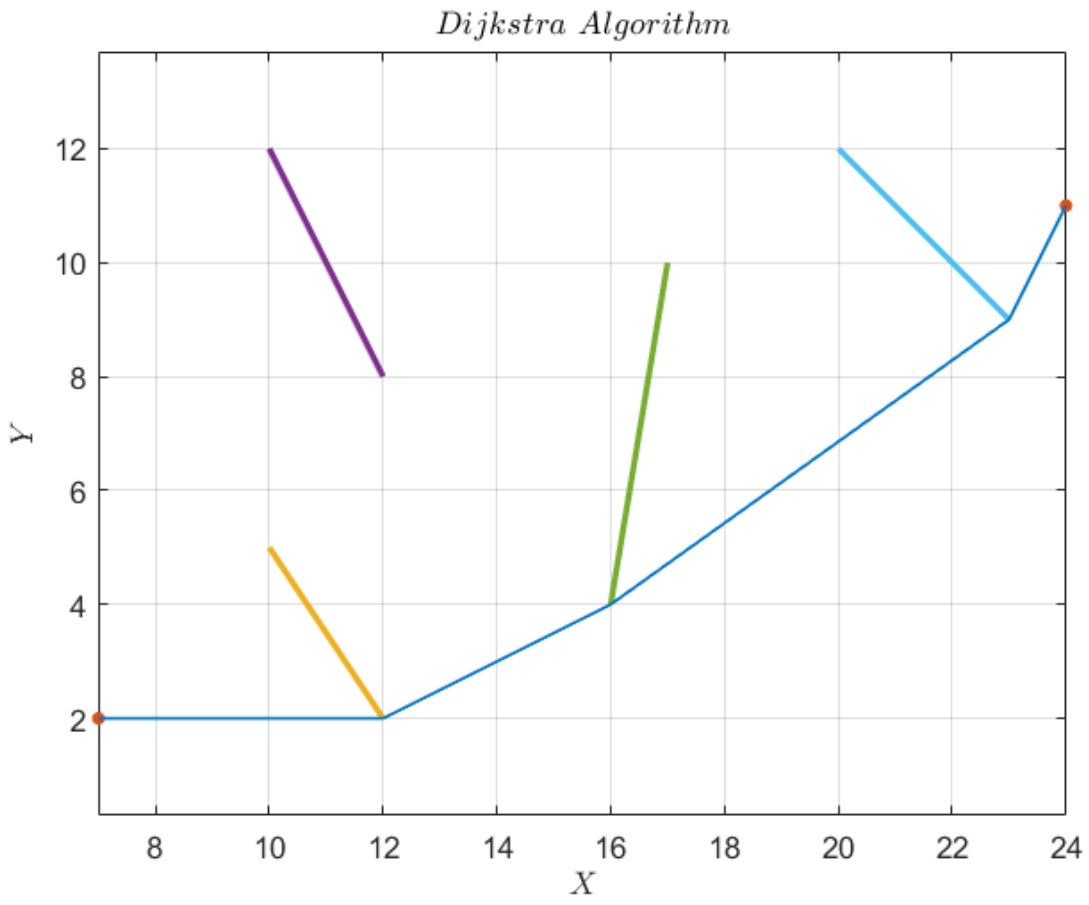
```
4
12,2,10,5
12,8,10,12
16,4,17,10
20,12,23,9
7,2
24,5
```



شکل ۷) بخش اجباری – ورودی ۱

ورودی ۲:

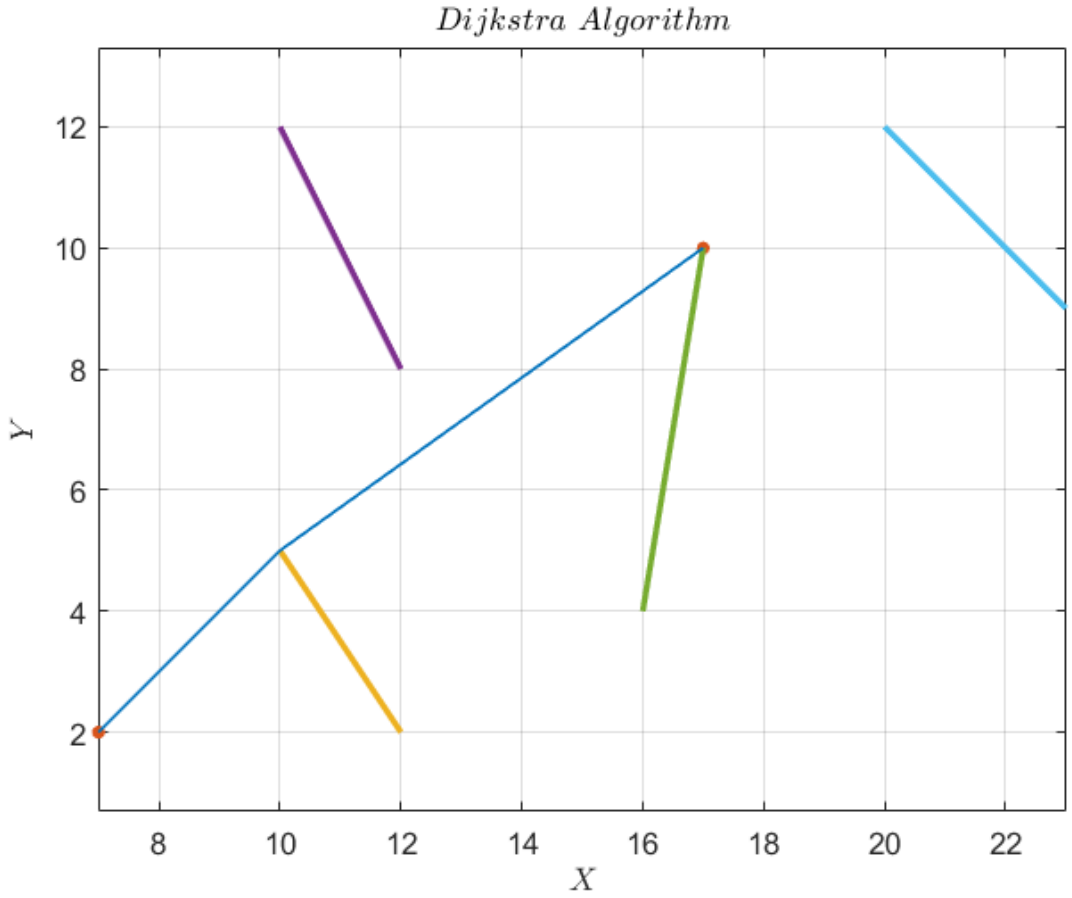
```
4
12,2,10,5
12,8,10,12
16,4,17,10
20,12,23,9
7,2
24,11
```



شکل ۸) بخش اجباری – ورودی ۲

ورودی ۳:

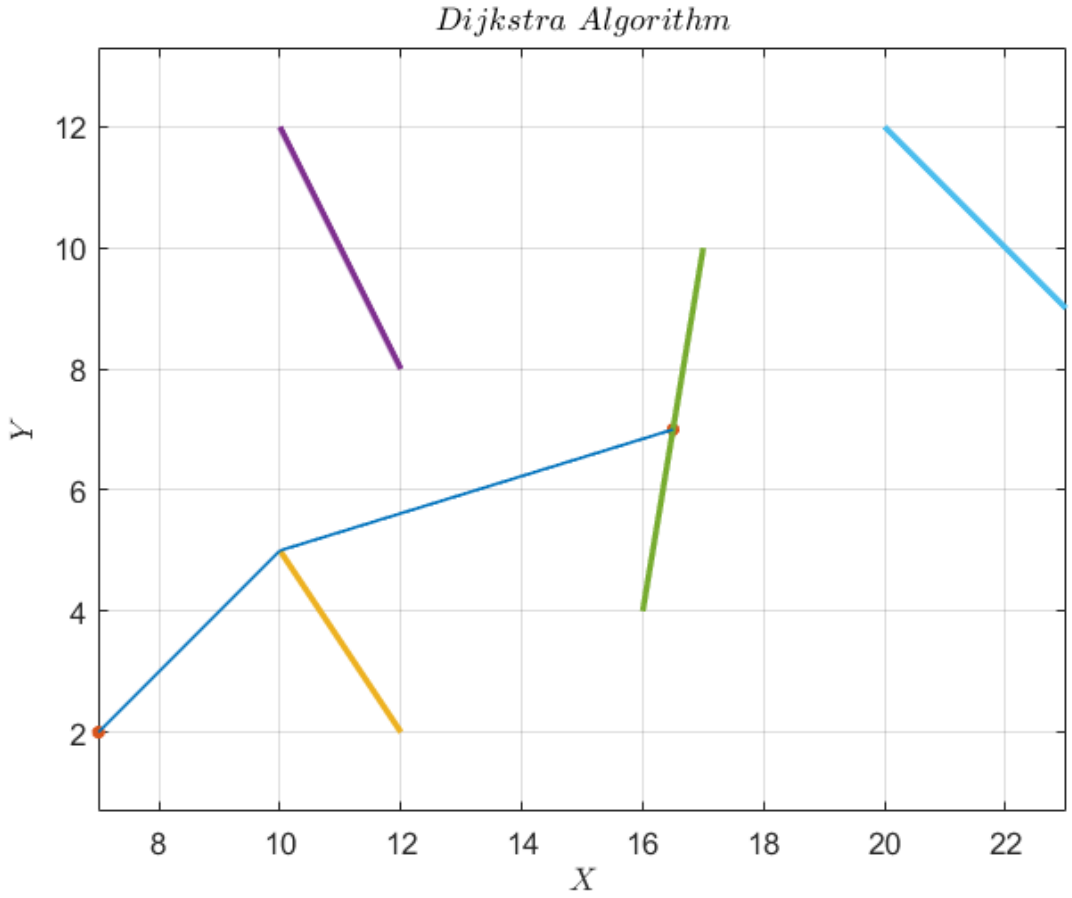
```
4
12,2,10,5
12,8,10,12
16,4,17,10
20,12,23,9
7,2
17,10
```



شکل ۹) بخش اجباری – ورودی ۳

ورودی ۴:

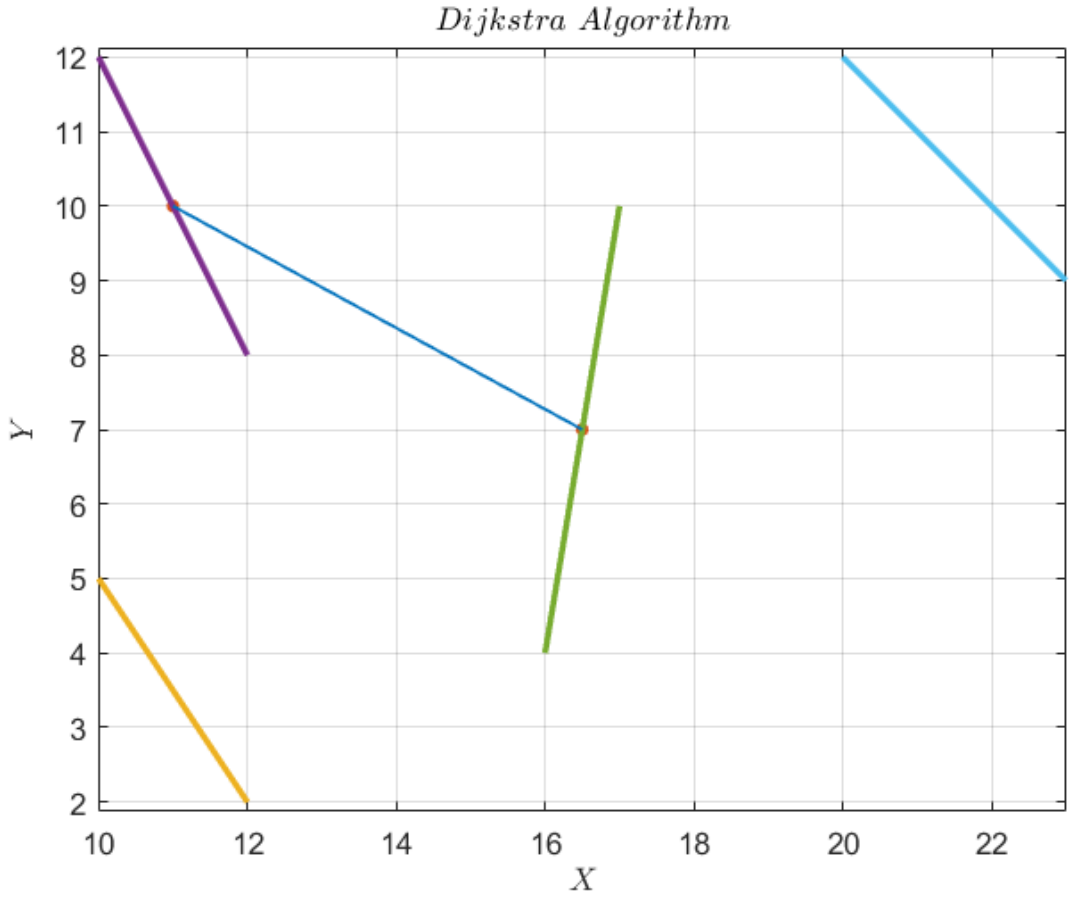
```
4
12,2,10,5
12,8,10,12
16,4,17,10
20,12,23,9
7,2
16.5,7
```



شکل ۱۰) بخش اجباری – ورودی ۴

ورودی ۵:

```
4
12,2,10,5
12,8,10,12
16,4,17,10
20,12,23,9
11,10
16.5,7
```

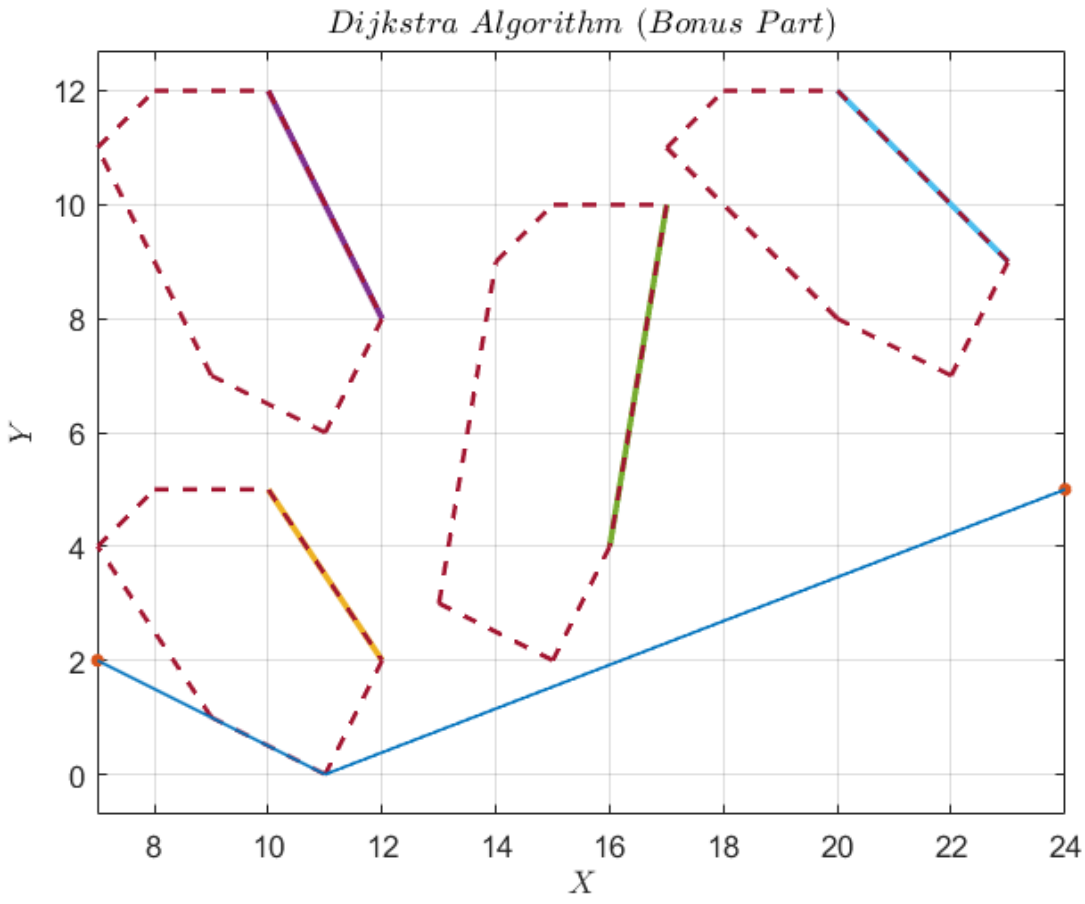


شکل ۱۱) بخش اجباری – ورودی ۵

آزمودن برنامه‌ی بخش امتیازی)

ورودی ۱:

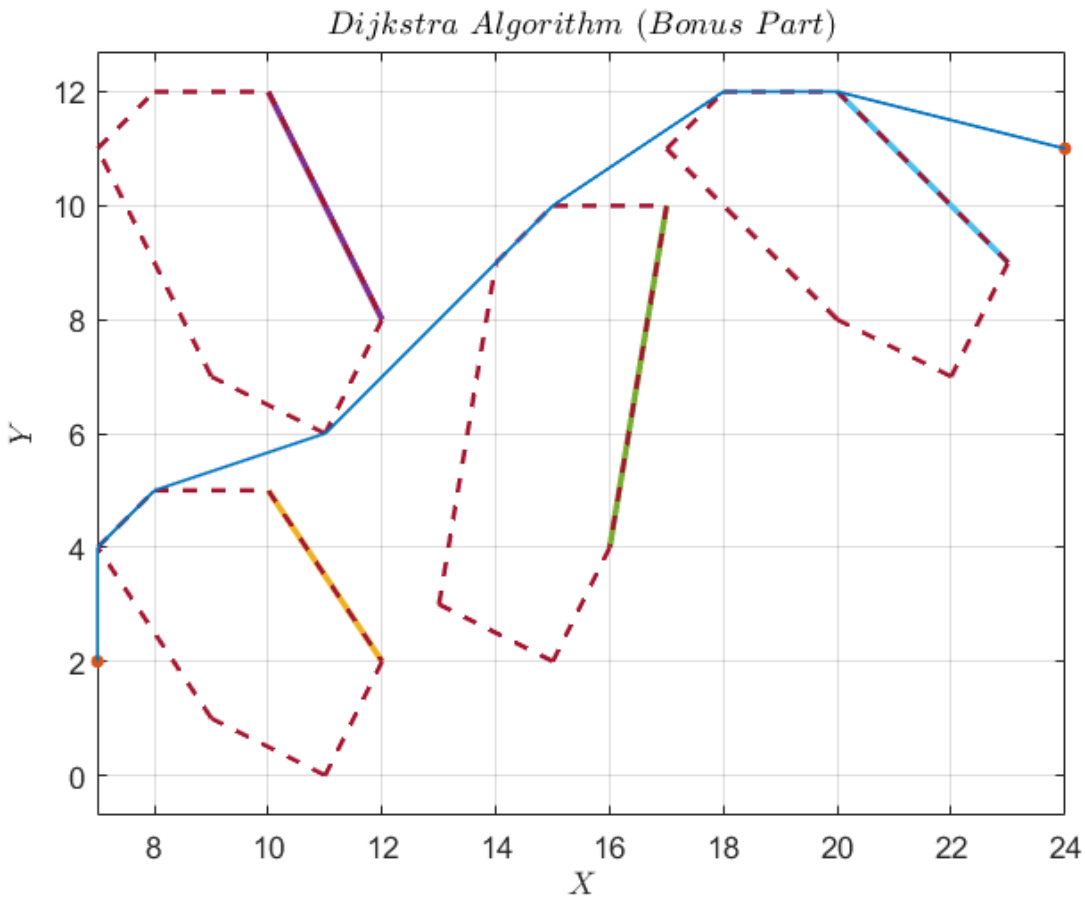
4 12,2,10,5 12,8,10,12 16,4,17,10 20,12,23,9 7,2 24,5	4 7,2 9,2 10,3 8,4



شکل ۱۲) بخش امتیازی – ورودی ۱

ورودی ۲:

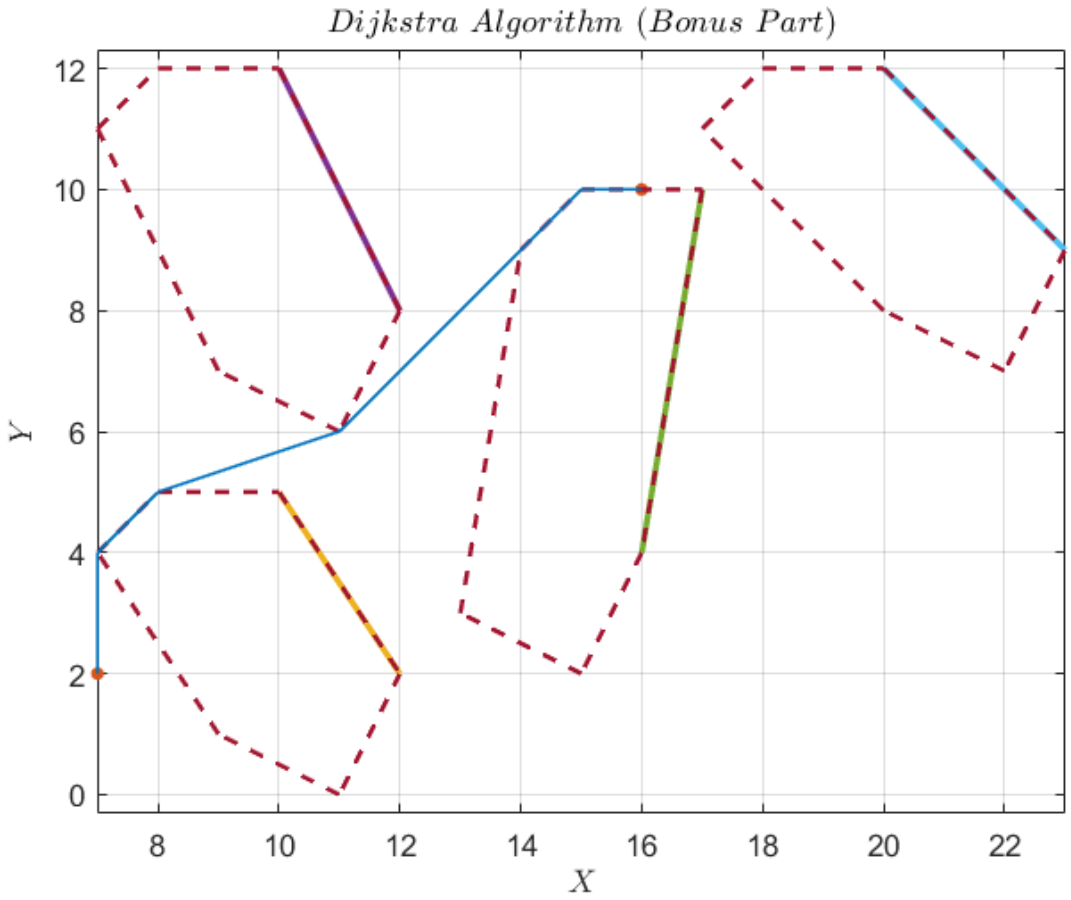
4 12,2,10,5 12,8,10,12 16,4,17,10 20,12,23,9 7,2 24,11	4 7,2 9,2 10,3 8,4



شکل ۱۳) بخش امتیازی – ورودی ۲

ورودی ۳:

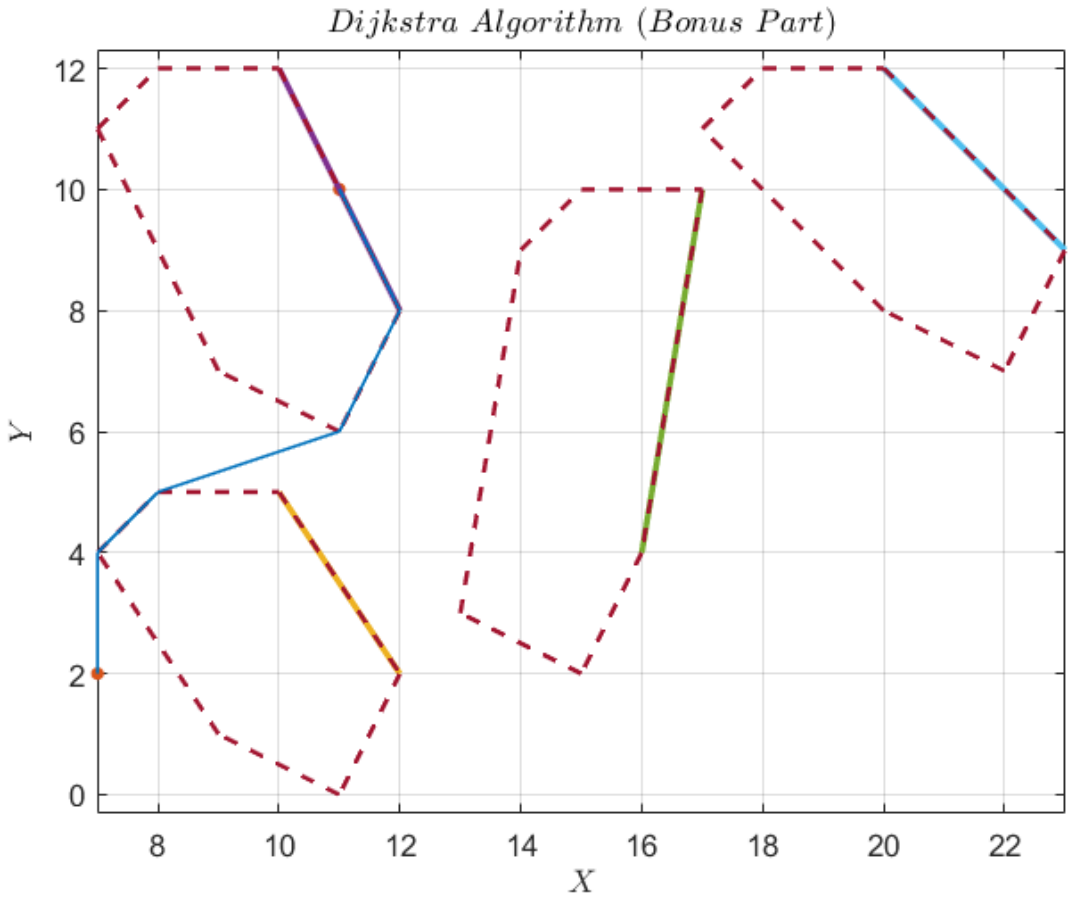
4 12,2,10,5 12,8,10,12 16,4,17,10 20,12,23,9 7,2 16,10	4 7,2 9,2 10,3 8,4



شکل ۱۴) بخش امتیازی – ورودی ۳

ورودی ۴:

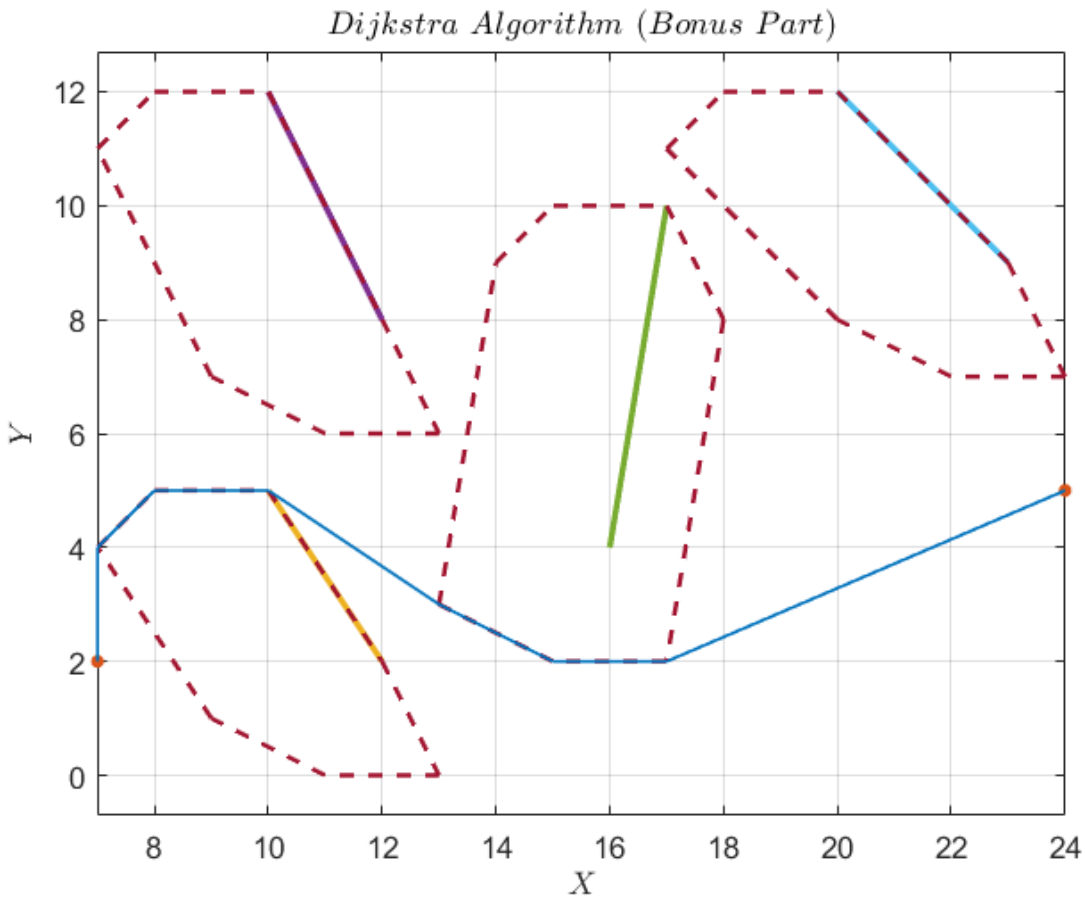
4 12,2,10,5 12,8,10,12 16,4,17,10 20,12,23,9 7,2 11,10	4 7,2 9,2 10,3 8,4



شکل ۱۵) بخش امتیازی – ورودی ۴

ورودی ۵:

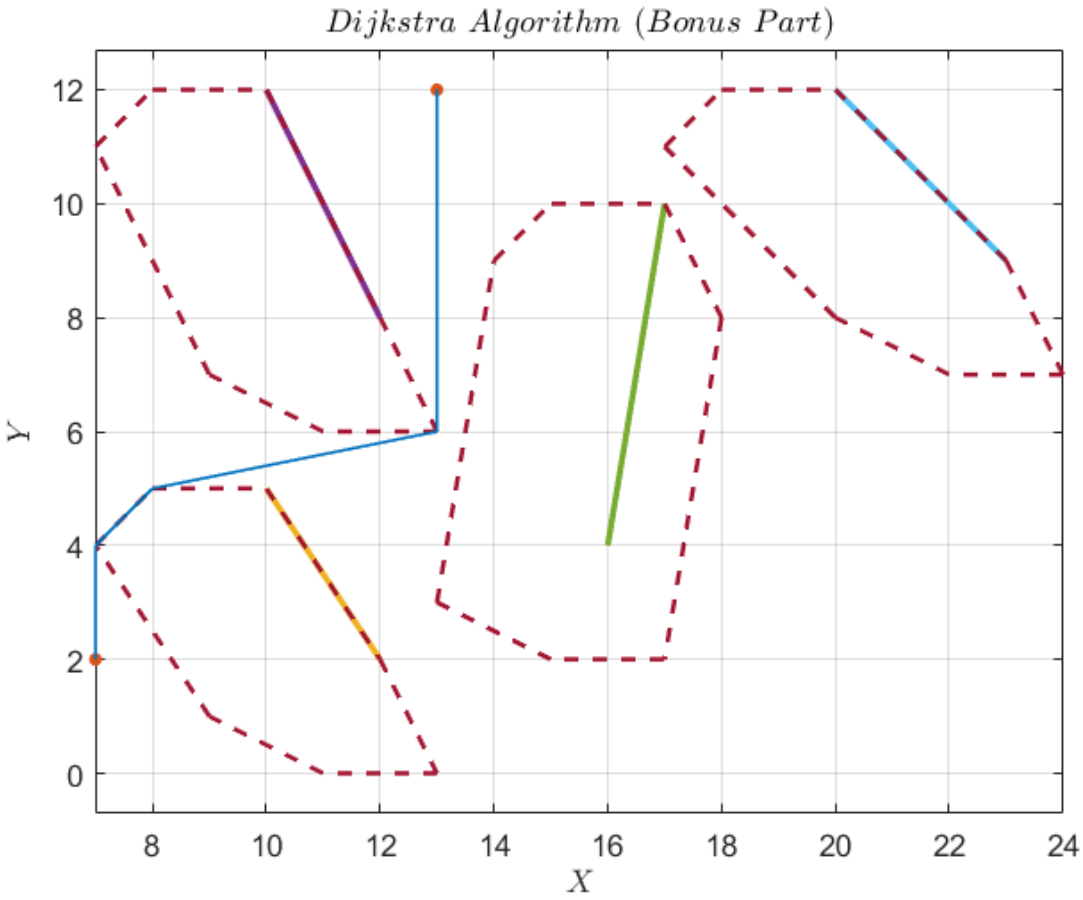
4 12,2,10,5 12,8,10,12 16,4,17,10 20,12,23,9 7,2 24,5	5 7,2 9,2 10,3 8,4 6,4



شکل ۱۶) بخش امتیازی – ورودی ۵

ورودی ۶:

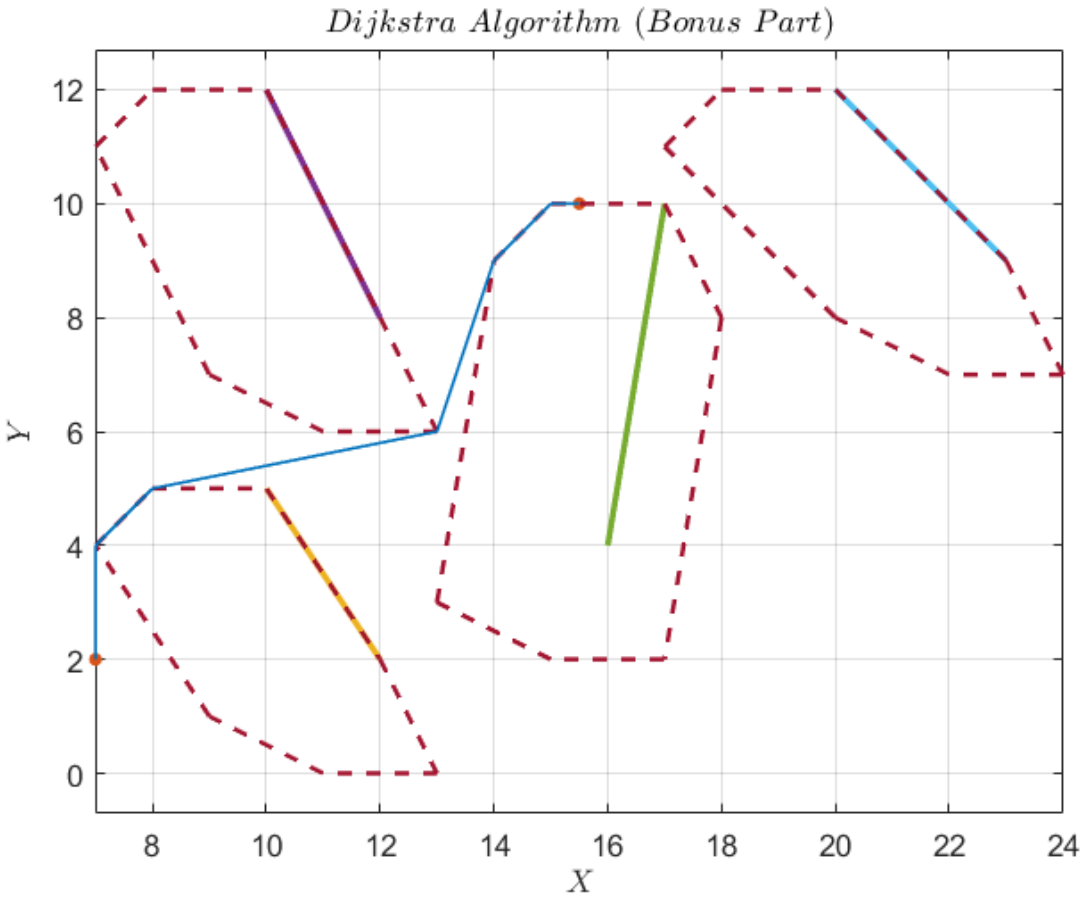
4 12,2,10,5 12,8,10,12 16,4,17,10 20,12,23,9 7,2 13,12	5 7,2 9,2 10,3 8,4 6,4



شکل ۱۷) بخش امتیازی – ورودی ۶

ورودی ۷:

4 12,2,10,5 12,8,10,12 16,4,17,10 20,12,23,9 7,2 15.5,10	5 7,2 9,2 10,3 8,4 6,4



شکل ۱۸) بخش امتیازی – ورودی ۷