CDEG 3.1 User's Manual

Nathaniel Miller

May 29, 2017

1 Introduction

Welcome to **CDEG**! **CDEG** stands for "Computerized Diagrammatic Euclidean Geometry." This program allows you to give formal proofs in Euclidean geometry that use diagrams and mirror the kinds of proofs found in Euclid's *Elements*.

2 What is it for?

CDEG is probably somewhat different from any other geometry software you may have previously encountered. You might have some questions about what it is and how it works. Here are some questions and answers. Some of these may be questions you already have; others, you may want to return to after playing with the program a bit.

- *What is* **CDEG** *for?*

  CDEG is a computer program that lets you give proofs in Euclidean Geometry, using diagrams and rules similar to those that Euclid used in the *Elements* 2300 years ago.

- *Proofs like the two-column proofs we did in my high school geometry class?*

  Yes, something like that. But since they are on a computer, the rules that are used are all completely precise and clearly specified in advance. Many other treatments of proof in geometry are not clear and precise about how all of their rules can be used and justified, especially when they use diagrams. Because **CDEG**'s proofs are implemented on a computer, there can be no ambiguity or disagreement about whether or not something is a complete and correct demonstration of a fact within **CDEG**.

- *I never totally understood what proofs were for. What is the point of giving geometric proofs?*

  Proofs have several different purposes and uses, but their main purpose is to demonstrate that something is true beyond any doubt. Mathematicians usually give arguments in ordinary language to demonstrate how they know that something is definitely true. Informal proofs in geometry typically use a mixture of words and diagrams to argue that some fact is known to be definitively true.

However, proofs of this type may sometimes contain subtle mistakes that are hard to notice. Giving a proof on a computer assures us that as long as we agree that all of the basic rules programmed into the system are sound, all of the conclusions demonstrated within the computer system will be valid.

- *How is* **CDEG** *different from other computer geometry programs like GeoGebra and the Geometer's Sketchpad?*

  Dynamic geometry programs such as these are like laboratories for doing geometric experiments. They allow you to make accurate pictures in the plane in which you can move around points, lines, and circles, take measurements, and make observations about what happens. They allow you to observe that something appears to happen in a particular case, but not to prove that it will always happen in every case.

- *How is* **CDEG** *able to make general arguments using diagrams, when any picture would seem to be an illustration of a specific case?*

  Unlike the pictures produced by dynamic geometry programs, which illustrate a particular case, each diagram produced by **CDEG** represents a class of possible situations in the plane. The key idea is that the information contained in a geometric diagram is contained in its topology—the features of the diagram that aren't changed by bending and stretching. Thus, a geometric diagram containing points, lines, and circles represents any collection of points, lines, and circles in the plane that intersect each other in the same way. For example, a diagram containing a single line segment represents all possible single line segments in the plane, since any such line segment can be stretched into any other.

  If you think about it, this is consistent with the ways that diagrams are typically used informally in geometry. A diagram sketched on a piece of paper is usually drawn so that its lines are not actually straight and its circles are not actually perfectly circular, but they still convey meaning. **CDEG** actually takes this a step further, drawing its diagrams solely to convey their topology and making no attempt to draw circular circles or lines without bends. Hopefully, this makes it easy to remember that its diagrams represent arrangements of geometric objects in the plane, rather than actually geometric objects.

- *How does* **CDEG** *handle geometric constructions?*

  In a dynamic geometry program, geometric constructions add new points, lines, and circles based on the current location of existing points, lines, and circles. For example, given two points, if they are connected by a line segment, the program will draw the particular line segment that actually connects those two particular points in their current location. By contrast, points in a **CDEG** diagram may represent an infinite collection of actual points in the plane. Different things may happen when we connect some of these points than when we connect others of these points. Thus, in **CDEG** the result of performing a geometric construction using some objects in a geometric diagram may be a collection of different diagrams showing all the possible ways the objects in the resulting figure may

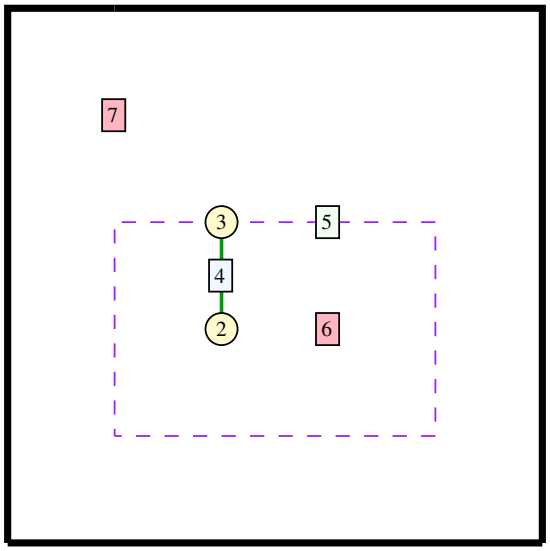


Figure 1: A **CDEG** diagram showing a single circle with one of its radii drawn.

be arranged. In order to distinguish them, we will refer a single diagram as a "primitive diagram" and a collection of possible diagrams as a "diagram array."

- *I'd like to hear more about how* **CDEG** *diagrams work.*

  Great! I'm glad you asked. Read on.

3 CDEG diagrams

An example of a **CDEG** diagram is shown in Figure 1. This diagram represents a single circle along with one of its radii. Note that the circle definitely does not look circular—it is drawn, in this case, as a rectangle. However, it has the same topology as a geometric circle—a single loop that comes back to where it started—and that is all we care about here.

In general, a **CDEG** diagram contains two kinds of objects: ***dots*** and ***segments***. The dots, which are shown as light yellow circles, represent points in the plane, while the segments represent pieces of lines and circles. There are actually two different kinds of segments that can occur in a diagram: ***solid segments***, which represent pieces of lines, and ***dotted segments***, which represent pieces of circles. Dotted segments are also sometimes referred to as ***arcs***. The dots and segments of a diagram are enclosed by a bold line called the ***frame***. The dots and segments of a diagram, along with the frame, break the diagram into a collection of ***regions***. Every dot, segment, region, and

piece of the frame in a **CDEG** diagram is labeled with a number, so that we can refer to it. Dots are labeled with a number inside the yellow circle; solid segments are labeled with a number in a light blue box along the segment; dotted segments are labeled with a number in a light green box along the segment; and regions are labeled with a number in a red box somewhere in the middle of the region.

The segments in a diagram are part of diagrammatic lines and circles. Each line and circle is assigned a different color, so that all of the segments that make up a given line or circle with be the same color. In general, segments that are not part of the same line or circle have different colors, although once there are a lot of different lines and circles, the colors are assigned randomly, so occasionally different lines may have similar colors. Lines may continue to the frame, in which case we consider them to be infinite in that direction. Thus, infinite lines intersect the frame in two places, rays intersect it in one place, and line segments don't intersect it at all.

Just as with traditional informal geometric diagrams, pieces of **CDEG** diagrams may be marked equal. Markers can mark line segments, angles, or areas in the diagram. Rather than being marked with slash marks on the diagram, as would be done with some informal diagrams, the markings are printed as text in a separate "Markings" box. (Representations that combine text with diagrams are sometimes referred to as ***heterogenous*** representations.) Line segments are represented by collections of solid line segments; these are referred to as ***diagrammatic segments*** or ***dsegs***. If n solid segments all intersect a dot d, then they divide the 360° around d into n different smallest possible angles. We refer to these as ***primative angles***, or ***pangles***, and label them by a pair of numbers (nd, ns), where nd is the number of the dot at the vertex, and ns is the number of the segment on the counterclockwise side of the primitive angle. In general, angles are represented in diagrams by collections of these primitive angles, which we refer to as ***diagrammatic angles*** or ***di-angles***. Finally, areas of the graph are represented by collections of regions, which we refer to as ***region sets***. Markers in **CDEG** diagrams can mark dsegs, di-angles, or region sets to be equal to other dsegs, di-angles, or region sets.

Now that we've seen what **CDEG** diagrams look like, let's look at an example of how they can be used in a proof, specifically the proof of Euclid's Proposition 1.

4 Tutorial Part 1: Euclid's Proposition 1

Euclid's *Elements* starts with 23 definitions, 5 Postulates, and 5 Common Notions, which are reproduced in Appendix A. He then uses these as assumptions to prove his following Propositions. The first proof he gives is of Proposition 1, reproduced in Appendix B.1, in which he shows that an equilateral triangle can be constructed on any given base. In this section, we will see how to duplicate Euclid's proof within **CDEG**.

When we start up **CDEG**, we see the window shown in Figure 2.

The program is driven by a graphical user interface. It contains menu options to manipulate the diagrams and control the program, an "Instructions" window to communicate with the user, and a "Markings" window to list any collections of line segments, angles, or regions that have been marked as being equal. In the bottom part of the window is shown the current diagram. Since we have just started up the program, this is

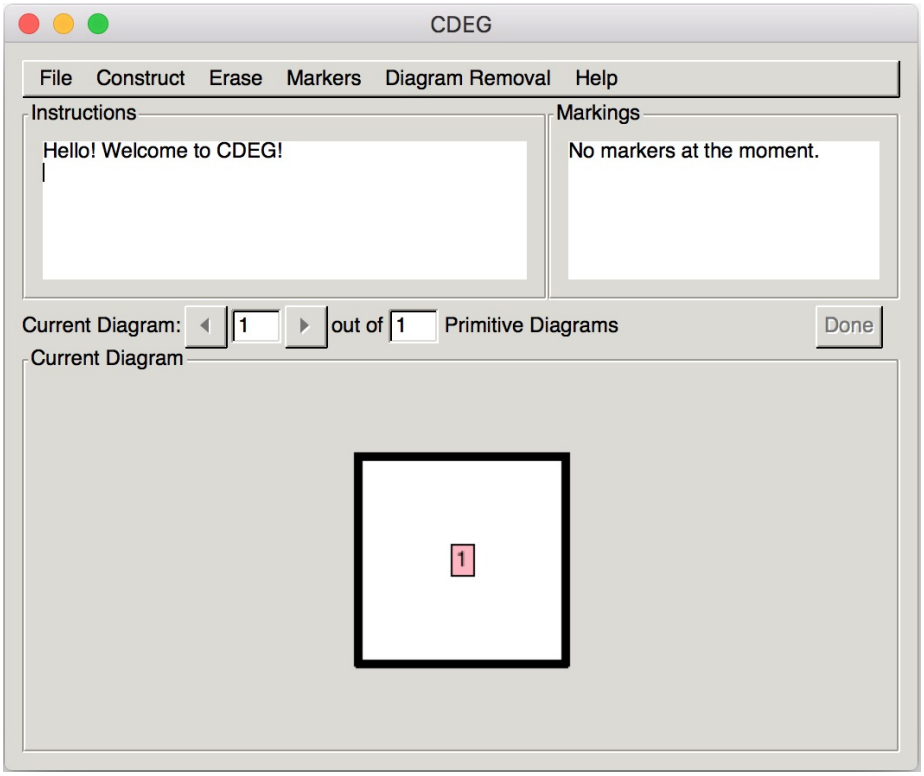


Figure 2: The **CDEG** starting window.

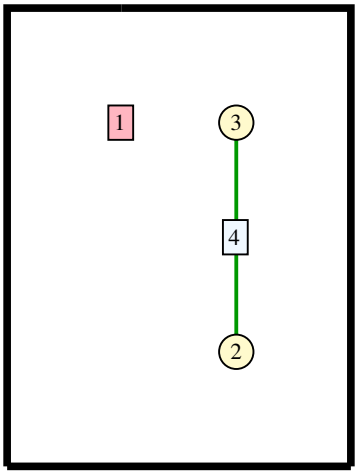


Figure 3: A **CDEG** diagram representing a single line segment.

the empty primitive diagram containing a single empty region that represents the whole plane, which is labeled as region 1.

Euclid's Proposition 1 starts with a line segment. We'll need to create a line segment in **CDEG** in order to replicate Euclid's proof. First, we'll need to add a point to region 1. We can do this by going to the "Construct" menu and selecting "Add point to region," then clicking on the region 1 label. This should result in a new diagram containing a point labeled as point 2. Once a menu option requiring further input is selected, the possible inputs (in this case, the region marker in the diagram) become clickable, and other menu options become grayed out. If, at any time, you want to abort an action that has been selected but not completed, you can do this by choosing "Abort Current Action" from the Help menu. This will reinstate all of the possible menu options. In order to complete the starting diagram for Euclid's proof, we will need to add another point to region 1, which will be labeled as point 3, and then connect points 2 and 3 with a line segment. We can do this by selecting the option "Draw segment" from the Construct menu, and then clicking on points 2 and 3 in turn. The resulting diagram is shown in Figure 3. This is the starting diagram for Euclid's proof. We can save it to come back to later by using the "Save Diagram" command found in the File menu.

The first thing that Euclid does is to draw a circle centered at one of the endpoints of the circle and passing through the other endpoint. In **CDEG**, this is accomplished by selecting the "Draw Circle" option from the Construct menu, then clicking on points 2 and 3 in that order. This results in the diagram that we have previously seen as Figure 1. Next, we need to draw another circle centered on the second endpoint (point 3) and passing through the first one (point 2). This can be done by again selecting the

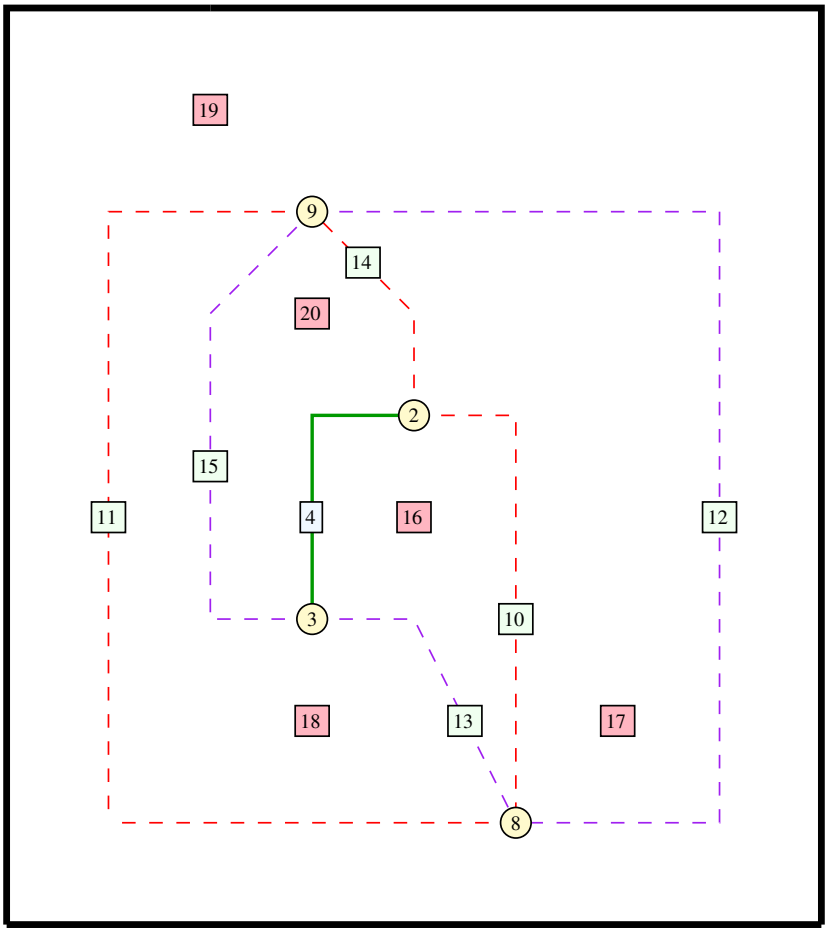


Figure 4: A **CDEG** diagram representing two intersecting circles.

"Draw Circle" option from the Construct menu, and then clicking on point 3 and then point 2. This gives us the diagram show in Figure 4. Finally, we need to connect both endpoints to one of the points of intersection of the two circles. We can do this using the "Draw segment" command again to connect point 3 to point 9, and then to connect point 2 to point 9. This gives us the diagram shown in Figure 5, which is the **CDEG** version of Euclid's diagram for Proposition 1 shown in Appendix B.1.

Now that we have constructed the triangle, we need to prove that it is equilateral. Euclid uses the fact that all radii of a circle are equal in length to mark segments 4 and 24 equal, since they are both radii of the purple circle. In **CDEG**, we can accomplish this by selecting the "Mark Radii of Circle" option from the Markers menu, and then clicking on any label on the circumference of the purple circle, such as the label for dotted segment 15. This doesn't change the displayed diagram, but adds the following text to the Markings window: "Marker 27 marks SegSetseg24 SegSetseg4 ." Notice that markers mark sets of geometric objects equal to other sets, which is why this refers to "SegSetseg24" rather than just Segment 24. Next, we can similarly mark segments 4 and 21 equal by using the "Mark Radii of Circle" command with the red circle, which adds "Marker 28 marks SegSetseg21 SegSetseg4" to the Markings window. To show that all three sides of the triangle are congruent, Euclid now uses the principle of transitivity, in the form of his Common Notion 1, to show that all three sides of the triangle are equal. In **CDEG**, we can mirror this by going to the Markers menu, highlighting the "Combine Markers" submenu, and then selecting "Combine Segment Markers." This command allows us to pick a set of segments, and then combine all the markers marking that set into one marker. In this case, Markers 27 and 28 both mark the bottom of the triangle, which is represented by the set containing the single element Segment 4. To select this set in **CDEG**, we first click on the label of Segment 4, and then click the "Done" button that appears on the right side of the window above the diagram to indicate that there are no other segments in this set. Doing this changes the display in the Markings window to "Marker 28 marks SegSetseg21 SegSetseg24 SegSetseg4." Thus, we now have a single marker that marks all three sides of the triangle as being congruent, showing that the triangle is equilateral.

Finally, we should clean up the diagram by erasing the superfluous pieces added in the course of our construction. We can do this using the commands in the Erase menu. If we use these commands to erase the two circles along with point number 8, we are left with the diagram shown in Figure 6. This diagrams shows an equilateral triangle constructed on the base segment that we started with, thus proving Euclid's Proposition 1.

5 Transcripts

Although **CDEG** is menu driven, it also keeps track of the user's actions in the form of a text-based ***transcript*** file. Each menu option that can be used as part of a geometric proof also has a corresponding textual name that is recorded in the transcript file when that menu option is applied. The transcript file also records the label numbers of any other inputs to the action. When an action requires the input of a set of numbers, such as when we were inputing a set of segments in the previous derivation of Euclid's first

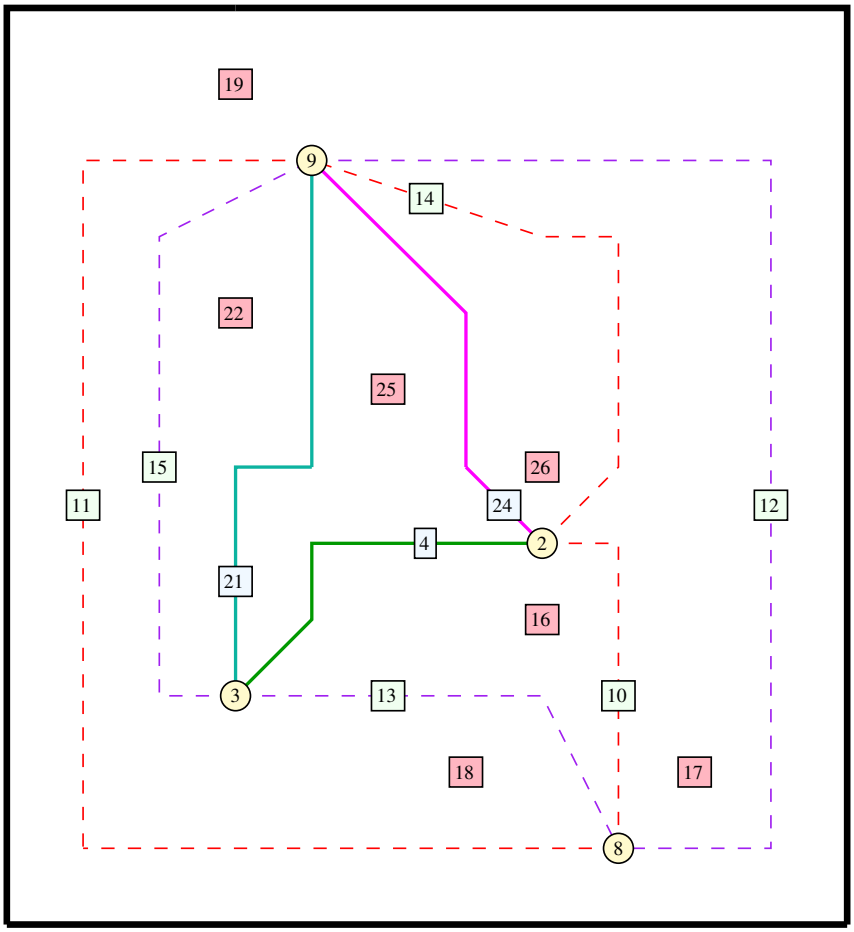


Figure 5: A **CDEG** diagram representing a triangle contained within two intersecting circles.

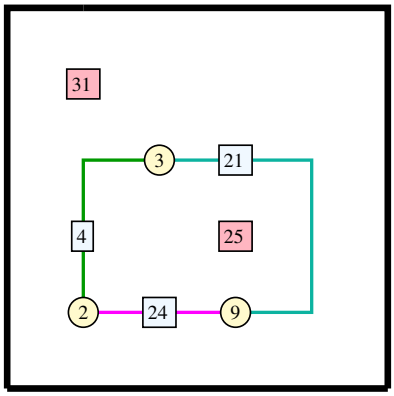


Figure 6: A **CDEG** diagram representing the triangle we are left with at the end of the proof of Euclid's first Propostion.

Proposition, these numbers are encased in curly braces. Each command, along with its inputs, is on a separate line of the file, and the last line of the file contains the single word "END." Menu items not affecting a proof, such as saving a diagram or opening a help window, are not recorded in the transcript.

As an example, the transcript file produced by our preceding derivation of Euclid's first proposition is as follows:

```
AddDotToReg 1
AddDotToReg 1
ConnectDots 2 3
DrawCirc 2 3
DrawCirc 3 2
ConnectDots 3 9
ConnectDots 2 9
MarkRadii 15
MarkRadii 11
SegMarkCombine { 4 }
EraseCircle 11
EraseCircle 15
EraseDot 8
END
```

CDEG automatically starts recording to a transcript as soon as it is started. The

transcript can be saved to a text file at any time by choosing "Save Transcript" from the File menu. This text file can also be edited by hand using any text editor. If the user wants a transcript that starts partway through a **CDEG** session, they can use the option "Reset Transcript" in the File menu. This discards everything in the current transcript and starts recording a new transcript from that point on.

CDEG can also "play back" a transcript, applying the commands found in the saved transcript to the current diagram in sequence. Thus, transcripts are an effective way for a user to save and replay a derivation. There are three ways to apply a transcript, all of which are found in the File Menu. If a transcript is applied "Stepwise Manually," a "Step" button appears, and one command from the transcript is applied each time it is pressed. If it is applied "Stepwise automatically," the commands found in the transcript file are executed in sequence with a fixed amount of time elapsing between each step. Finally, a transcript can be applied "All at once." In this case, all of the steps are applied in sequence immediately, and only the final resulting diagram is then displayed.

Transcript files edited by hand can also contain comments delimited by `(*` and `*)`, which will be ignored by **CDEG** when playing back the transcript.

6 Tutorial Part 2: Case Analysis

In this section, we will explore the way that **CDEG** handles a construction that results in an array of possibilities. To get the starting diagram, we will go back to the starting diagram from Euclid's first Proposition, which contained a line segment, and then add two more points to region 1. If you saved the starting diagram, you can reload it using the "Load Diagram" option from the File menu, and then add two more points using the "Add point to Region" option from the construct menu. The resulting diagram is shown in Figure 7.

Notice the line above the diagram that says "Current Diagram: 1 out of 1 Primitive Diagrams." That tells us that we are currently working with a single diagram. This diagram represents one line segment and two other points in the plane. What should happen if we connect the other two points? There are a variety of different options, depending on where the other two points actually lie with respect to the line segment and each other. If we use the "Draw Segment" command to connect points 5 and 6, the line above the diagram changes to say "1 out of 9 Primitive Diagrams" to indicate that we are now looking at the first diagram in an array of 9 diagrams. This is because there are 9 topolgically distinct cases that might occur when these dots are connected. The new line might intersect the original line at either endpoint or along the segment between them, and can do so in either of 2 possible orientations; or it could be collinear with the original line, in either of 2 possible orientations; or it might not intersect the original segment at all.

In order to view all of these different possibilities, we can move through them using the arrows on the "Current Diagram:" line, or else we can type the number of the diagram we wish to view directly into the box on this line that tells us which diagram we are looking at. One of the possibilities (the 6th diagram in the array) is shown in Figure 8.

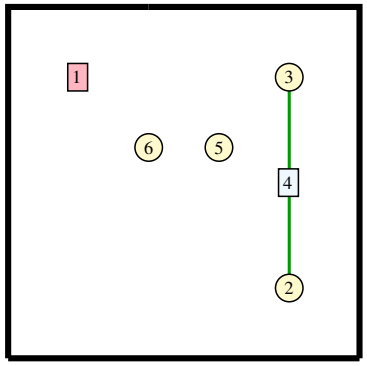


Figure 7: The starting diagram for a branching of several cases.

7 CDEG commands

In this section, we list all of **CDEG**'s menu commands (along with their transcript abbreviations in parentheses) for reference. After picking a menu item, any additional inputs are specified by clicking on the appropriate labels in the diagram. The "Instructions" window generally explains what input is next needed. If a command requires a set of objects to be selected, then they are selected by clicking on them one at a time and then clicking the "Done" button. Angles are selected by clicking on their vertex followed by their sides in clockwise order. Labels in the diagram are only clickable when it is appropriate to the current command, and most menu items become greyed out and disabled while the inputs for the current command are being input. A command that has been selected but not yet executed because not all of its inputs have been selected yet can be aborted by selecting "Abort Current Action" from the Help menu, which will return **CDEG** to the state it was in before the current action was selected.

File Menu commands:

- **Load Diagram:** Loads a previously saved diagram.
- **Save Diagram:** Saves the current diagram array to a file.
- **Save As EPS format:** Saves the current primitive diagram as a graphics file in .eps (encapsulated postscript) format. This is useful for exporting graphics. Most of the figures in this User Manual were created using this command.

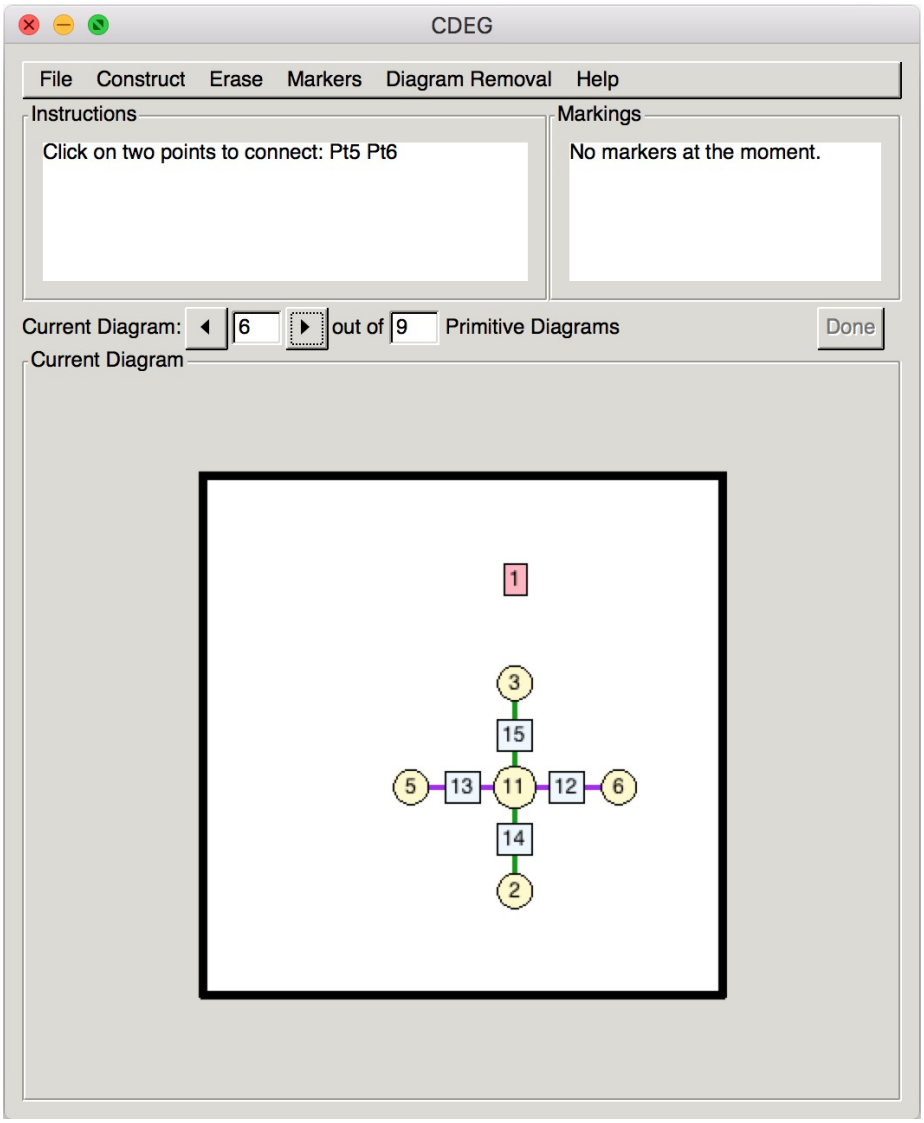


Figure 8: One of 9 possible results.

- **Reset Transcript:** Clears the transcript of the current session. If a transcript file is subsequently saved, it will start from here, excluding any previous actions.
- **Save Transcript:** Saves the current transcript to a text file.
- **Apply Transcript Stepwise Manually:** Performs the commands found in a transcript file one at a time, in order. One command is performed every time the "Step" button above the diagram display is pressed.
- **Apply Transcript Stepwise Automatically:** Performs the commands found in a transcript file one at a time, in order, waiting a fixed time between each step. This is useful for automatically showing a whole derivation.
- **Apply Transcript All at Once:** Performs the commands found in a transcript file one at a time, in order, then immediately displays the diagram resulting from the final step.
- **Quit:** Quits the **CDEG** program.

Construct Menu commands:

- **Add Point to Region:** (`AddDotToReg`) This command adds a new dot to the specifed region.
- **Add Point to Segment:** (`AddDotToSeg`) This command adds a new dot to the given segment, usually breaking it into two segments.
- **Add Point to Arc:** (`AddDotToArc`) This command adds a new dot to the given arc of a circle, usually breaking it into two arcs.
- **Draw Segment:** (`ConnectDots`) This command corresponds to Euclid's first Postulate. It allows any two given dots to be connected by a line segment. (The two dots cannot be on the frame, though.) It outputs an array of possible diagrams; this array can be quite large if the starting diagram has a lot of objects in it. The running time of this command can be exponential in the number of objects already in the diagram, so it may take a long time to run. However, it generally doesn't take as long to run as drawing a circle.
- **Extend Segment to Ray:** (`ExtendtoRay`) This command allows you to extend a line segment to a ray. To use it, you must first choose one segment already on the line, and then the dot at the end of the line segment on the side you want to extend.
- **Extend Segment to Infinite Line:** (`ExtendSeg`) This command allows you to extend a line segment to an infinite line. To use it, you must click on one segment already on the line. This command corresponds to Euclid's second Postulate.

- **Draw Circle:** (`DrawCirc`) This command allows you to draw a new circle, and corresponds to Euclid's third Postulate. It takes the center dot and a dot that will be on the circumference as inputs. It outputs an array of possible diagrams; this array can be quite large if the starting diagram has a lot of objects in it. In general, this is the command that takes the longest to run and produces the largest number of output diagrams. Its running time can be exponential in the number of objects already in the diagram, and so it can take intractably long to run for even moderately complicated diagrams.

Erase Menu commands:

- **Erase Point:** (`EraseDot`) This allows you to erase a dot from the diagram by clicking on it. The dot cannot be at an intersection point of two objects or the endpoint of a line. To delete a point in these situations, you must first delete the other objects.
- **Erase Line:** (`EraseLine`) This allows you to delete a whole line by clicking on one of its segments.
- **Erase Line Ends:** (`EraseLineEnds`) This allows you to delete part of a line, leaving only a single connected segment. To use it, first click on one segment of the line, and then on the two dots that will be the endpoints of the segment that is left.
- **Erase Circle:** (`EraseCircle`) This allows you to delete a circle by clicking on one of the dotted segments on its circumference.
- **Erase Marker:** (`EraseMarker`) This command pops up a window listing all of the markers in your diagram. Selecting one causes that marker to be deleted.
- **Erase All Unused Markers:** (`EraseUnusedMarkers`) This command removes any makers that are in the diagram that no longer mark any objects in the diagram. These unused markers can arise when the pieces of the diagram that they marked have been removed.

Markers Menu commands:

- **Mark Radii of Circle:** (`MarkRadii`) This allows you to mark all of the radii of a given circle as being equal by clicking on one of the dotted segments on the circumference of the circle.
- **New Marker:** (`MarkOneSeg/MarkOneAng/MarkOneReg`) This allows you to mark a single set of segments, angles, or regions with a new marker.

- **Marker Sum:** (`SegAdd/AngAdd/RegAdd`) This rule corresponds to Euclid's Common Notion 2, which says that "If equals be added to equals, the wholes are equal." If you have one pair of objects A_1 and B_1 (segment, angle, or region sets) that are marked equal in a diagram, and another pair of objects A_2 and B_2 of the same type are also marked equal, then this rule allows you to mark $A_1 + A_2$ equal to $B_1 + B_2$. A_1 and A_2 must be disjoint, and likewise, B_1 and B_2 must be disjoint.

- **Marker Difference:** (`SegSubtract/AngSubtract/RegSubtract`) This rule corresponds to Euclid's Common Notion 3, which says that "If equals be subtracted from equals, the remainders are equal." If you have one pair of objects A_1 and B_1 (segment, angle, or region sets) that are marked equal in a diagram, and another pair of objects A_2 and B_2 of the same type are also marked equal, and A_1 is a proper subset of A_2, and B_1 is a proper subset of B_2 then this rule allows you to mark $A_2 - A_1$ equal to $B_2 - B_1$. The objects must be entered in the order A_1, A_2, B_1, B_2.

- **Combine Markers:** (`SegMarkCombine/AngMarkCombine/RegionMarkCombine`) This rule corresponds to Euclid's Common Notion 1, which states that "Things which are equal to the same thing are also equal to one another." If there is one object O (segment, angle, or region set) in the diagram that is marked by more than one marker, this rule combines the markers, so that all of the objects marked by any of the markers that mark O are now marked equal with a single marker.

- **Apply Side-Angle-Side Triangle Congruence:** (`ApplySAS`) Hopefully, **CDEG** will eventually include the means to use the method of superposition in proofs. This is the method of proof that Euclid uses to prove the Side-Angle-Side and Side-Side-Side triangle congruence rules. It is possible to incorporate this method into a formal system like this, as described in [2]. However, because this method will be non-trivial to program and will require a huge number of cases to be considered, for the moment **CDEG** instead directly incorporates these two triangle congruence rules as inference rules. This rule asks the user to select the corner vertices of two triangles that have two pairs of corresponding sides and the corresponding contained angles marked equal, and marks equal the remaining corresponding sides and angles of the triangles, along with the region sets enclosed by the triangles. The corners of the triangles must be entered in the corresponding order, with the second corner given being the one with the equal corresponding angles.

- **Apply Side-Side-Side Triangle Congruence:** (`ApplySSS`) This rule asks the user to select the corner vertices of two triangles that have all three pairs of corresponding sides marked equal, and marks equal the corresponding angles of the triangles, along with the region sets enclosed by the triangles. The corners of the triangles must be entered in the corresponding order.

- **Free Marking (UNSAFE!):** (`FreeMarkSeg/FreeMarkAng/FreeMarkReg`) This option allows you to mark any two segment, angle, or region sets equal.

This is not a sound inference rule in derivations, which is why it is marked as being unsafe, but it can be used in creating starting diagrams or for marking objects equal to one another when you know that they can be marked equal using previous derivations that you don't want to take the time to repeat.

Diagram Removal Menu The commands in this menu allow you to erase a single primitive diagram from a diagram array. A primitive diagram can be erased if there is a reason that the given arrangement is not possible. The menu contains the following commands:

- **Remove because of Segment Contradiction:** (`ApplyCS`) This rule applies when the diagram contains two segment sets which are marked equal, and such that one is properly contained in the other one. This rule, along with the next two rules, corresponds to Euclid's Common Notion 5, which states that "The whole is greater than the part."
- **Remove because of Angle Contradiction:** (`ApplyCA`) This rule applies when the diagram contains two angle sets which are marked equal, and such that one is properly contained in the other one.
- **Remove because of Region Contradiction:** (`ApplyCR`) This rule applies when the diagram contains two region sets which are marked equal, and such that one is properly contained in the other one.
- **Remove for Contradiction of Euclid's Fifth Postulate:** (`ApplyEFP`) This allows you to erase a diagram that satisfies the hypotheses of Euclid's fifth postulate but not its conclusion. That is, it allows you to erase a diagram if it contains two infinite lines crossed by an infinite transversal, such that the interior angles on one side of the transversal are properly contained in a set of angles that is marked equal to a straight angle, in which the two lines don't intersect each other on that side of the transversal. This command requires the user to select one segment on the transversal; one segments from each of the two lines that intersect the transversal on the given side, the angle set containing the interior angles which is marked equal to a straight angle, and the vertex and counter-clockwise side segment of the straight angle.
- **Free Removal (UNSAFE!):** (`EraseFree`) This allows you to erase any diagram. It is not a sound inference rule in derivations, which is why it is labeled as being unsafe, but it can be used in creating starting diagrams, for erasing diagrams when there is another identical (isomorphic) diagram elsewhere in the array, or for erasing diagrams that you know can be eliminated using previous derivations that you don't want to take the time to repeat.

Help Menu commands:

- **Abort Current Action:** This command aborts an action that has been started by selecting its menu item, but has not been completed by clicking on all of the appropriate inputs. It is useful when you have mistakenly selected the wrong command. In some cases, it is possible to select a command which is impossible to complete because the necessary inputs are not even present in the current diagram; in this case, this is the only way to get out of this situation.
- **About:** This brings up a window with information about **CDEG**.
- **Help:** This brings up a window with some helpful information.
- **License/Legal:** This brings up a window with legal disclaimers and information about **CDEG**'s license. **CDEG** is free software, and is distributed under the terms of the Gnu General Public License, version 3.

In addition to the menu options, if the current diagram is an array, the current primitive diagram can be changed using the up and down arrows on the line above the drawn diagram, or by typing the number of the desired primitive diagram directly into the box between them. The corresponding transcript command is `SetPD` followed by the number of the desired primitive diagram.

8 CDEG vs. Euclid's Elements

CDEG should be theoretically able to prove versions of all of Euclid's Propositions from the first four books of the *E*lements, which is the part that deals purely with planar geometry. We have already seen that Euclid's definitions, Propositions, and Common Notions are reproduced for reference in Appendix A, while his proofs of Propositions 1 and 5 are given in Appendix B. In Section 4, we saw how to duplicate Proposition 1 in **CDEG**. A good next step in learning to use **CDEG** might be to try to likewise duplicate Euclid's proof of Proposition 5.

However, anyone who tries to actually use **CDEG** to duplicate all of the proofs in these books will quickly realize that in practice it will be very difficult to use **CDEG** to duplicate all of Euclid's *Elements*. One issue is that as the diagrams become more complicated, the amount of time required for one step in the proof can grow exponentially. As noted above, the commands that draw circles and lines can take an amount of time that is exponential in the number of objects in a diagram. Thus, some computations may take impractically long. Furthermore, the number of new diagrams produced by these commands can also be exponential in the number of objects in the diagram. So the number of cases that need to be considered can also grow very quickly.

Another issue that exacerbates this problem is the lack of lemma incorporation in **CDEG**. Lemma incorporation refers to the use of previously derived Propositions and Lemmas in proving new Theorems. Of course, these previously derived Propositions and Lemmas can always be rederived in the course of a proof. However, the additional objects in the diagram in the course of a later proof normally necessitate considering even more cases. Thus, the lack of Lemma Incorporation here can lead to a huge blowup in the length of a given proof. For more discussion about Lemma incorporation,

see [2]. Lemma Incorporation will hopefully be included in some future version of **CDEG**.

In the meantime, one way to use **CDEG** is to try to duplicate one of Euclid's proofs, but to only complete the proof for one branch of the many possible cases that arise. This is, in fact, Euclid's normal practice. You can also use the Free Marking and Free Removal commands to incorporate previously proven results without entirely rederiving them.

9 Technical Details

If you are interested in more technical details behind **CDEG** and the formal system that it is based on, they can be found in my book, *Euclid and his Twentieth Century Rivals: Diagrams in the Logic of Euclidean Geometry* [2].

This system is based on a precisely defined syntax and semantics of Euclidean diagrams. What does this mean? To say that it has a precisely defined syntax means that all the rules of what constitutes a diagram and how we can move from one diagram to another have been completely specified. The fact that these rules are completely specified is perhaps obvious since you have been using the formal system on a computer, and computers can only operate with such precisely defined rules. However, it was commonly thought for many years that it was not possible to give Euclidean diagrams a precise syntax, and that the rules governing the use of such diagrams were inherently informal.

To say that the system has a precisely defined semantics means that the meaning of each diagram has also been precisely specified. In general, as we have seen, one diagram drawn by **CDEG** can actually represent many different possible collections of lines and circles in the plane. What these collections all share, and share with the diagram that represents them, is that they all have the same topology. This means that any one can be stretched into any other, staying in the plane. So, for example, a diagram containing a single line segment represents all possible single line segments in the plane, since any such line segment can be stretched into any other.

CDEG allows you to manipulate diagrams through the use of construction and inference rules. These rules are meant to be ***sound***. This means that if you start with a diagram D that represents a collection $\mathcal{C}$ of points, lines, and circles in the plane, and you apply a rule to D, then at least one of the diagrams in the resulting diagram array should still represent $\mathcal{C}$ (or, in the case of a construction rule that added a line or circle to the diagram, $\mathcal{C}$ with the appropriate line or circle added). For more details, see [2].

10 Bug reports

The current version of **CDEG** is still a beta version, and there are likely still bugs in the program. If you find a bug, *please* let me know about it! You can send the transcript file from the **CDEG** session in which the bug appeared to `nathaniel.miller@unco.edu`. If you have other comments or suggestions about the program, I would love to hear them as well.

A From Book I of Euclid's *Elements*

Euclid wrote the thirteen books of the *Elements* around 300 B.C. The first four books are about elementary planar geometry; books five through ten are largely about the theory of ratio and proportion and about what would now be considered number theory; and books eleven through thirteen are about three dimensional geometry.

The excerpt reprinted here is taken from the edition translated and edited by Thomas Heath [1].

A.1 Definitions

1. A ***point*** is that which has no part.
2. A ***line*** is breadthless length.
3. The extremities of a line are points.
4. A ***straight*** line is a line which lies evenly with the points on itself.
5. A ***surface*** is that which has length and breadth only.
6. The extremities of a surface are lines.
7. A ***plane*** surface is a surface which lies evenly with the straight lines on itself.
8. A ***plane angle*** is the inclination to one another of two lines in a plane which meet one another and do not lie in a straight line.
9. And when the lines containing the angle are straight, the angle is called ***rectilineal***.
10. When a straight line set up on a straight line makes the adjacent angles equal to one another, each of the equal angles is ***right***, and the straight line standing on the other is called a ***perpendicular*** to that on which it stands.
11. An ***obtuse*** angle is an angle greater than a right angle.
12. An ***acute*** angle is an angle less than a right angle.
13. A ***boundary*** is that which is an extremity of anything.
14. A ***figure*** is that which is contained by any boundary or boundaries.
15. A ***circle*** is a plane figure contained by one line such that all the straight lines falling upon it from one point among those lying within the figure are equal to one another;
16. And the point is called the ***centre*** of the circle.
17. A ***diameter*** of the circle is any straight line drawn through the centre and terminated in both directions by the circumference of the circle, and such a straight line also bisects the circle.

18. A ***semicircle*** is the figure contained by the diameter and the circumference cut off by it. And the centre of the semicircle is the same as that of the circle.

19. ***Rectilineal figures*** are those which are contained by straight lines, ***trilateral*** figures being those contained by three, ***quadrilateral*** those contained by four, and ***multilateral*** those contained by more than four straight lines.

20. Of trilateral figures, an ***equilateral*** triangle is that which has its three sides equal, an ***isosceles*** triangle that which has two of its sides alone equal, and a ***scalene*** triangle that which has its three sides unequal.

21. Further, of trilateral figures, a ***right-angled*** triangle is that which has a right angle, an ***obtuse-angled*** triangle that which has an obtuse angle, and an ***acute-angled*** triangle that which has its three angles acute.

22. Of quadrilateral figures, a ***square*** is that which is both equilateral and right-angled; an ***oblong*** [rectangle] that which is right-angled but not equilateral; a ***rhombus*** that which is equilateral but not right-angled; and a ***rhomboid*** [parallelogram] that which has its opposite sides and angles equal to one another but is neither equilateral nor right-angled. And let quadrilaterals other than these be called ***trapezia***.

23. ***Parallel*** straight lines are straight lines which, being in the same plane and being produced indefinitely in both directions, do not meet one another in either direction.

A.2 Postulates

Let the following be postulated:

1. To draw a straight line from any point to any point.

2. To produce a finite straight line continuously in a straight line.

3. To describe a circle with any centre and distance.

4. That all right angles are equal to one another.

5. That, if a straight line falling on two straight lines make the interior angles on the same side less than two right angles, the two straight lines, if produced indefinitely, meet on that side on which are the angles less than the two right angles.

A.3 Common Notions

1. Things which are equal to the same thing are also equal to one another.

2. If equals be added to equals, the wholes are equal.

3. If equals be subtracted from equals, the remainders are equal.

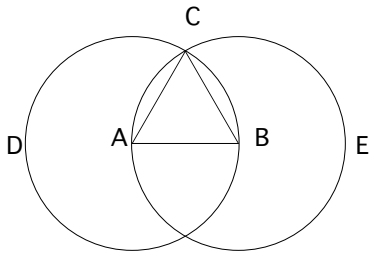
4. Things which coincide with one another are equal to one another.

5. The whole is greater than the part.

B Propositions

B.1 Proposition 1.

On a given finite straight line to construct an equilateral triangle.



Let AB be the given finite straight line.

Thus it is required to construct an equilateral triangle on the straight line AB.

With centre A and distance AB let the circle BCD be described; [Post. 3] again, with centre B and distance BA let the circle ACE be described; [Post. 3] and from the point C, in which the circles cut one another, to the points A, B let the straight lines CA, CB be joined. [Post. 1]

Now, since the point A is the centre of the circle CDB, AC is equal to AB. [Def. 15]

Again, since the point B is the centre of the circle CAE, BC is equal to BA. [Def. 15]

But CA was also proved equal to AB; therefore each of the straight lines CA, CB is equal to AB.

And things which are equal to the same thing are also equal to one another; [C.N. 1] therefore CA is also equal to CB.

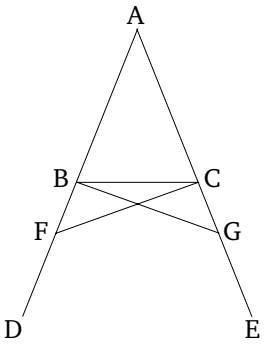
Therefore the three straight lines CA, AB, BC are equal to one another.

Therefore the triangle ABC is equilateral; and it has been constructed on the given finite straight line AB.

(Being) what it was required to do.

B.2 Proposition 5.

In isosceles triangles the angles at the base are equal to one another, and, if the equal straight lines be produced further, the angles under the base will be equal to one another.



Let ABC be an isosceles triangle having the side AB equal to the side AC; and let the straight lines BD, CE be produced further in a straight line with AB, AC. [Post. 2]

I say that the angle ABC is equal to the angle ACB, and the angle CBD to the angle BCE.

Let a point F be taken at random on BD; from AE the greater let AG be cut off equal to AF the less; [I. 3] and let the straight lines FC, GB be joined. [Post. 1]

Then, since AF is equal to AG and AB to AC, the two sides FA, AC are equal to the two sides GA, AB, respectively; and they contain a common angle, the angle FAG. Therefore the base FC is equal to the base GB, and the triangle AFC is equal to the triangle AGB, and the remaining angles will be equal to the remaining angles respectively, namely those which the equal sides subtend, that is, the angle ACF to the angle ABG, and the angle AFC to the angle AGB. [I. 4]

And, since the whole AF is equal to the whole AG, and in these AB is equal to AC, the remainder BF is equal to the remainder CG.

But FC was also proved equal to GB; therefore the two sides BF, FC are equal to the two sides CG, GB respectively; and the angle BFC is equal to the angle CGB, while the base BC is common to them; therefore the triangle BFC is also equal to the triangle CGB, and the remaining angles will be equal to the remaining angles respectively, namely those which the equal sides subtend; therefore the angle FBC is equal to the angle GCB, and the angle BCF to the angle CBG.

Accordingly, since the whole angle ABG was proved equal to the angle ACF, and in these the angle CBG is equal to the angle BCF, the remaining angle ABC is equal to the remaining angle ACB; and they are at the base of the triangle ABC. But the angle FBC was also proved equal to the angle GCB; and they are under the base.

Therefore etc.

Q. E. D.

References

[1] Euclid. 1956. *The Elements*. New York: Dover, 2nd edn. Translated with introduction and commentary by Thomas L. Heath.

[2] Miller, Nathaniel. 2007. *Euclid and his twentieth century rivals: Diagrams in the logic of Euclidean geometry*. Stanford, CA: CSLI Press.