

USER MANUAL SLK 4.1b (2D).

By: Dante Camargo.

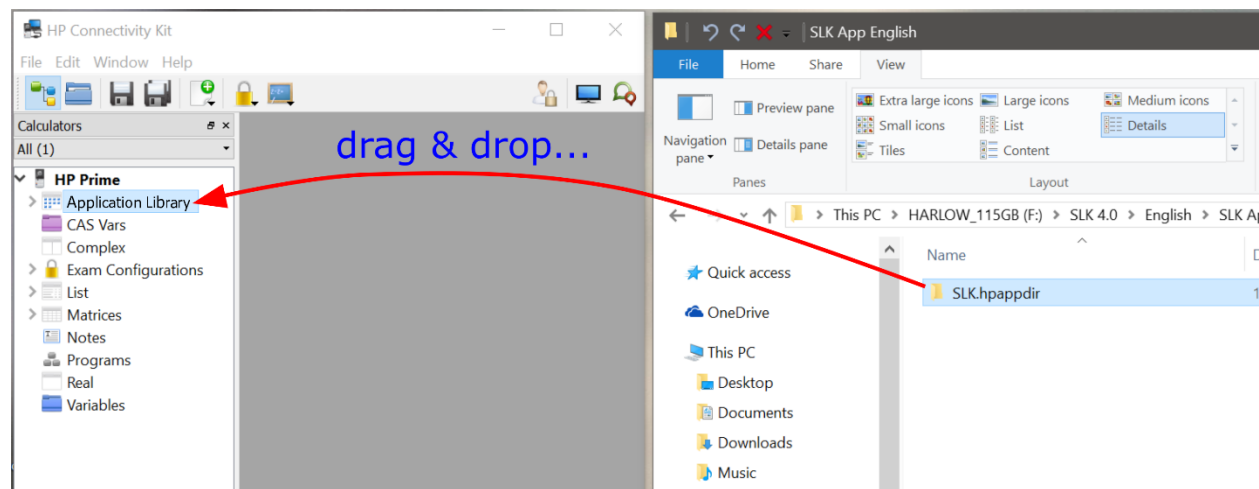
September 2018.

Morelia Michoacán, México.

SLK 4.1b Is mathematical Program originally written for the HP50G, now ported to the new HP Prime Graphing Calculator. **SLK 4.1b** Is specially focused in the Straight line, it is intuitive and user friendly. In order to work smoothly in **SLK 4.1b**... I suggest the User to have a conscious knowledge about how an equation of the line is formed in: **Standard** form: $(Ax+By+C=0)$ as well as in **Slope-intercept** form: $(y=mx+b)$.

IMPORTANT: The way you interpret your equations when it comes to entering the coefficients, some people make the mistake of entering the coefficients in the wrong way, because of the way their equations are written in their paper. Most sections within **SLK 4.1b** will expect you to enter your equations as: $Ax+By+C=0$ which is correct, and a totally different thing is to enter the coefficients as in: $Ax+By=C$, which in this case the coefficient **C** will change its sign and that will lead our results to be incorrect. **SLK 4.1b** will only ask you to enter your equations as: $Ax+By=C$ when working with the (Intersection Point program) other than that, you will always be asked to enter your coefficients as: $Ax+By+C=0$.

To install this App, Open the connectivity Kit, connect your calculator (or Emulator) and then just drag & drop the "SLK.hpappdir" folder into the connectivity Kit window on the [Application Library] section.



Now click save, then you may want to re-compile the program by opening the program manager (in your calculator or emulator) pressing **[SHIFT]+[1]**, This opens up the program menu, now look for the App called: **SLK (App)** or maybe called **Function (App)**, once you find it, open it by pressing **[ENTER]** now you should be able to view the source code of **SLK**, next you press the "check" button (In the soft menu area) to check for errors, if there's no errors, you can close the **program editor** by pressing the **[ESC]** key. That should re-compile **SLK**, so it'll run smoothly and correctly.

Now I'll describe each section that comes bundled in *SLK 4.1b*:

- **Distance Between Two Points.**

The program assumes that you have 2 Points $P_1:(X_1,Y_1)$ & $P_2:(X_2,Y_2)$ and you want to find the distance between these. To do that, you'll need to enter the numeric values for: X_1 , Y_1 , X_2 and Y_2 , these numeric values can be integers or Real numbers. Once you're done entering your data, press [ENTER] to see the distance.

- **Slope Between Two Points.**

The program assumes that you have 2 Points $P_1:(X_1,Y_1)$ & $P_2:(X_2,Y_2)$ and you want to find the Slope between these. To do that, you'll need to enter the numeric values for: X_1 , Y_1 , X_2 and Y_2 , these numeric values can be integers or Real numbers. Once you're done entering your data, press [ENTER] to see the slope.

- **Two-Point Form Equation.**

The program assumes that you have 2 Points $P_1:(X_1,Y_1)$ & $P_2:(X_2,Y_2)$ and you want to find the Equation between these. To do that, you'll need to enter the numeric values for: X_1 , Y_1 , X_2 and Y_2 , these numeric values can be integers or Real numbers. Once you're done entering your data, press [ENTER] to see the equation.

- **Distance Between Point & Line.**

a). - Finding the Distance between a point $P:(x_0,y_0)$ & a line in Standard Form: $Ax+By+C=0$

The program assumes that you have an Equation of a line in **Standard Form: $Ax+By+C=0$** and a point $P_0:(x_0,y_0)$, and you want to find the shortest perpendicular distance between these. To find such distance, you'll need enter the numeric values for coefficients: **A**, **B**, and **C** as well as for: (x_0, y_0) , where:

A= the coefficient of the (X) variable, it has to be a **POSITIVE** Integer.

B= the coefficient of the (Y) variable, it has to be an Integer.

C= the constant, it has to be an Integer.

x_0 = the (X) coordinate of P_0 .

y_0 = the (Y) coordinate of P_0 .

The values for; x_0 and y_0 can be integers or Real numbers.

If your equation is missing one or more coefficients, you can always enter zeros instead, also if your equation contains decimal and/or fractional coefficients, it means it's not in Standard form, so you can always standardize it first, by going to the Main Menu and selecting the option "E" [Equation Fixer], then come back to this program and enter it. Once you're done entering your data press [ENTER] to see then distance.

b). - Finding the Distance between a Point $P:(x_0,y_0)$ and a line in Slope-Intercept Form: $y=mx+b$.

The Program assumes that you have a point $P_0:(x_0,y_0)$ and a line or segment in Slope-Intercept Form: $y=mx+b$. You want to find the shortest perpendicular distance between this point and the line. To find such distance you'll need to enter the numeric values for: x_0 , y_0 , **m** and **b**, where:

m= the Slope of your equation.

b= the y-Intercept (this is the 'Y' value where your line crosses the Y-Axis.)

x_0 = the (X) coordinate of P_0 .

y_0 = the (Y) coordinate of P_0 .

If your Equation is missing slope(**m**) or the Y-Intercept (**b**), you can always fill up the blanks with zeros instead. The values for **m**, **b**, **X₀** and **Y₀** can be integers or Real numbers. Once you're done entering your data, press [ENTER] to see the distance.

c).- Distance between point & line, when you are given $P:(x_0,y_0)$, $P_1:(x_1,y_1)$ & $P_2:(x_2,y_2)$.

The Program assumes that you have a line (segment) defined by two points: $P_1:(X_1,Y_1)$ & $P_2:(X_2,Y_2)$ and a third point $P_0:(x_0,y_0)$ which is not part of the line. You want to find the shortest perpendicular distance between the point (**P₀**) and the **line** that passes through: (**P₁**) and (**P₂**). To find such distance you'll need to enter the coordinate values for each one of the points, which are: **X₁**, **Y₁**, **X₂**, **Y₂**, **X₀**, and **Y₀**. These values can be integers or Real numbers. Once you're done entering your data, press [ENTER] to see the distance.

- **Midpoint/ End-Point.**

Midpoint:

The program assumes that you have a segment formed by 2 Points: a starting Point: $P_1:(X_1,Y_1)$ and the End Point: $P_2:(X_2,Y_2)$. You want to find the coordinates of a Point that is right in the middle of **P₁** & **P₂**, this point is also known as Midpoint. To find the coordinates of the Midpoint, you'll need to enter the values for the **starting** and the **end** points:

X₁ and **Y₁**: are the coordinates for the starting point.

X₂ and **Y₂**: are the coordinates for the end point.

The values for: **X₁**, **Y₁**, **X₂** and **Y₂** can be integers or Real numbers. Once you're done entering your data press [ENTER] to see the coordinates of the Midpoint.

End-Point:

The program assumes that you have a segment between 2 Points, these points happen to be the Starting Point: $P_1:(X_1,Y_1)$ and the Midpoint: $P_2:(X_2,Y_2)$, you want to know what the coordinates of the Endpoint are, so enter the numeric values for the two Points that you already know, in this case:

X₁ and **Y₁**: the numeric values for the Starting Point.

X₂ and **Y₂**: the numeric values for the Midpoint.

The values for: **X₁**, **Y₁**, **X₂** and **Y₂** can be integers or Real numbers. Once you're done entering your data press [ENTER] to see the coordinates of the Endpoint.

- **Point–Slope Form Equation.**

The Program assumes that you have a Slope:(**m**) and a Point: $P_1:(X_1,Y_1)$, and you want to know the Equation that passes through this point and has the given slope. To find such equation, you'll need to enter numeric values for each one of the Input fields, where:

m= is the slope of your line.

X₁ and **Y₁**: are the coordinates for the a given Point that is already part of the line.

The values for: **m**, **X₁** and **Y₁** can be integers or Real numbers.

Once you're done entering your data, press [ENTER] to see the resulting Equation.

- $y=mx+b \Leftrightarrow Ax+By+C=0$ *Transforming equations between Standard & Slope-Intercept forms.*

$y=mx+b \Rightarrow Ax+By+C=0$

The program assumes that you have an Equation of a line in **Slope-Intercept Form**: $y=mx+b$ and you want to transform it to its **Standard Form**: $Ax+By+C=0$. To perform such transformation, you'll need to enter the numeric values for: **slope**:(m) and the **y-Intercept**:(b). Also the values for: m and b can be integers or Real numbers. If your equation is missing one of the values, you can always enter zeros instead, Once you're done entering your data, press **[ENTER]** to see the transformed Equation.

$Ax+By+C=0 \Rightarrow y=mx+b$

The program assumes that you have an equation of a straight line in **Standard Form**: $Ax+By+C=0$ and you want to transform it to its **Slope-Intercept Form**: $y=mx+b$, to perform such transformation, you'll need to enter the numeric values for each one of the coefficients: A , B , & C , where:

A = the coefficient of the (X) variable, it has to be a **POSITIVE** Integer.

B = the coefficient of the (Y) variable, it has to be an Integer.

C = the constant, it has to be an Integer.

If your equation is missing one or more coefficients, you can always enter zeros instead, also if your Equation contains decimal and/or fractional coefficients, it means it is not in Standard form, so you can always Standardize it first, by going to the main menu and select the "E" option [**Equation Fixer**], then come back and enter your equation. Once you're done entering your data, press **[ENTER]** to see the transformed Equation.

- *Equation \Rightarrow Coordinates.*

$Ax+By+C=0 \Rightarrow$ Computer generated Coordinates.

The program assumes that you have an Equation in **Standard form**: $Ax+By+C=0$ and you want to convert it to its **Coordinate Form** $P_1:(X_1,Y_1)$ & $P_2:(X_2,Y_2)$, in other words you want to know 2 arbitrary points on this line, In this case the Program generates this coordinates **RANDOMLY**, all you have to do is enter the values for Coefficients: A , B , & C Where:

A = the coefficient of the (X) variable, it has to be a **POSITIVE** Integer.

B = the coefficient of the (Y) variable, it has to be an Integer.

C = the constant, it has to be an Integer.

If your equation is missing one or more coefficients, you can always enter zeros instead, but if your Equation contains decimal and/or fractional coefficients, it means it is not in Standard form, so you can always Standardize it first, by going to the main menu and select the "E" option [**Equation Fixer**]. Once you're done entering your data, press **[ENTER]** to see the computer generated coordinates.

$Ax+By+C=0 \Rightarrow$ User Defined Coordinates.

The program assumes that you have an Equation in **Standard form**: $Ax+By+C=0$ and you want to convert it to its **coordinate Form** $P_1:(X_1,Y_1)$ & $P_2:(X_2,Y_2)$, The program allows you to define the distances for: X_1 and X_2 , so the resulting coordinates will be located at the distances you specified. To find such coordinates, you'll need to enter the numeric values for coefficients: A , B , & C and the values for X_1 and X_2 , where:

A = the coefficient of the (X) variable, it has to be a **POSITIVE** Integer.

B = the coefficient of the (Y) variable, it has to be an Integer.

C= the constant, it has to be an Integer.

X₁= the numeric value on the **X-Axis** where you want the first coordinate to be generated.

X₂= the numeric value on the **X-Axis** where you want the second coordinate to be generated.

Values for: (**A**, **B** & **C**) can **only** be integers or Real numbers, also the values for (**X₁** and **X₂**) must be different from each other. If your equation is missing one or more coefficients, you can always enter zeros instead. Also if your Equation contains decimal and/or fractional coefficients, it means it is not in Standard form, so you can always Standardize it first, by going to the main menu and select the "E" option [**Equation Fixer**], then come back and enter it. Once you're done entering your data, press [**ENTER**] to see the User defined coordinates.

y=mx+b ⇒ Computer generated Coordinates.

The program assumes that you have an Equation in **Slope-Intercept form**: **y=mx+b** and you want to convert it to its **coordinate Form**: **P₁:(X₁,Y₁) & P₂:(X₂,Y₂)**. in other words you want to know 2 arbitrary points that lay on this line, In this case the Program generates this coordinates **RANDOMLY**. To find such coordinates you have to enter the numeric values for coefficients: (**m** & **b**), where:

m= the Slope of the line.

b= the y-intercept (this is the 'Y' value where your line crosses the **Y-Axis**).

The values for **m** and **b** can be integers or Real numbers. If your equation is missing one or more coefficients, you can always enter zeros instead. Once you're done entering your data press [**ENTER**] to see the computer generated coordinates.

y=mx+b ⇒ User Defined Coordinates.

The program assumes that you have an Equation in **Slope-Intercept form**: **y=mx+b** and you want to convert it to its **coordinate form** **P₁:(X₁,Y₁) & P₂:(X₂,Y₂)**, The program allows you to define the distances for: **X₁** and **X₂**, so the resulting coordinates will be located at the distances you specified. To find such coordinates, you'll need to enter the numeric values for coefficients (**m**, **b**, **X₁** & **X₂**), where:

m= the Slope of your line.

b= the y-intercept (this is the 'Y' value where your line crosses the **Y-Axis**).

X₁= a numeric value on the **X-Axis** where you want the first coordinate to be generated.

X₂= a numeric value on the **X-Axis** where you want the second coordinate to be generated.

The numeric values for: **m**, **b**, **X₁** and **X₂** can be integers or Real numbers, also the values for (**X₁** and **X₂**) must be different from each other. In case your equation is missing one or more coefficients, you can always enter zeros instead. Once you're done entering your data, press [**ENTER**] to see the User defined coordinates.

• **Parallel Lines.**

Parallelism between 2 straight lines (Equations) in Standard Form.

The program assumes that you have 2 Equations in Standard form: **Ax+By+C=0** and you want to find out if these lines are parallel to each other. To determine that, you'll need to enter the numeric values for coefficients: **A₁**, **B₁**, **C₁**, **A₂**, **B₂** and **C₂**, where:

A₁= the coefficient of the (**X**) variable from Equation # 1, it has to be a **POSITIVE** integer.

B₁= the coefficient of the (**Y**) variable from Equation # 1, it has to be an integer.

C₁= the constant of equation # 1, it has to be an integer.

A₂= the coefficient of the (**X**) variable from Equation # 2, it has to be a **POSITIVE** Integer.

B₂= the coefficient of the (**Y**) variable from Equation # 2, it has to be an integer.

C₂= the constant of equation # 2, it has to be an integer.

In case your equation is missing one or more coefficients, you can always enter zeros instead. But if your Equation contains decimal and/or fractional coefficients, it means it is not in Standard form, so you can always Standardize it first, by going to the main menu and select the "E" option [[Equation Fixer](#)], then come back and enter it. Once you're done entering your data, press [[ENTER](#)] to see if the lines are parallel or not.

Parallelism between 2 straight lines (Equations) in Slope-Intercept Form.

The program assumes that you have the Equations of 2 straight lines in [Slope-Intercept form](#): $y=mx+b$ and you want to find out if these lines are parallel to each other, to determine that... you'll need to enter the numeric values for: m_1 , b_1 , m_2 & b_2 Where:

m_1 = the Slope of your line.

b_1 = y-intercept (this is the 'Y' value where your line crosses the Y-Axis).

m_2 = the Slope of your line.

b_2 = y-intercept (this is the 'Y' value where your line crosses the Y-Axis).

The numeric values for m and b can be integers or Real numbers. If your equation is missing one or more coefficients, you can always enter zeros instead. Once you're done entering your data, press [[ENTER](#)] to see if the lines are parallel or not.

- **Perpendicular Lines.**

Perpendicularity between 2 Lines in Standard Form ($Ax+By+C=0$)

The program assumes that you have 2 Equations in [Standard form](#): $Ax+By+C=0$ and you want to find out if these lines are perpendicular to each other. To determine that, you'll need to enter the numeric values for coefficients: A_1 , B_1 , C_1 , A_2 , B_2 and C_2 . Where:

A_1 = the coefficient of the (X) variable from Equation # 1, It has to be a **POSITIVE** Integer.

B_1 = the coefficient of the (Y) variable from Equation # 1, It has to be an Integer.

C_1 = the constant of equation # 1, it has to be an Integer.

A_2 = the coefficient of the (X) variable from Equation # 2, it has to be a **POSITIVE** Integer.

B_2 = the coefficient of the (Y) variable from Equation # 2, it has to be an integer.

C_2 = the constant of equation # 2, it has to be an Integer.

In case your equation is missing one or more coefficients, you can always enter zeros instead. But if your Equation contains decimal and/or fractional coefficients, it means it is not in Standard form, so you can always Standardize it first, by going to the main menu and select the "E" option [[Equation Fixer](#)], then come back and enter it. Once you're done entering your data, press [[ENTER](#)] to see if the lines are perpendicular or not.

Perpendicularity between 2 Lines in Slope-Intercept Form: $y=mx+b$

The program assumes that you have 2 Equations in [Slope-Intercept form](#): $y=mx+b$ and you want to find out if these lines are perpendicular to each other. To determine that, you'll need to enter the values for:

m_1 , b_1 , m_2 and b_2 where:

m_1 = the Slope of the equation # 1.

b_1 = the y-intercept (this is the 'Y' value where the equation # 1 crosses the Y-Axis).

m_2 = the Slope of the equation # 2.

b_2 = the y-intercept (this is the 'Y' value where the equation # 2 crosses the Y-Axis).

The numeric values for: m_1 , b_1 , m_2 and b_2 can be integers or Real numbers. If your equation is missing one or more coefficients, you can always enter zeros instead. Once you're done entering your data, press [[ENTER](#)] to see if the lines are perpendicular or not.

- **(X) & (Y) Intercepts.**

Finding the (X) & (Y) Intercepts for an Equation in Standard Form ($Ax+By+C=0$).

The program assumes that you have an Equation in **Standard form**: $Ax+By+C=0$ and you want to find out what the (X) & (Y) Intercepts for this particular line are, to determine that, you'll need to enter the numeric values for coefficients: **A**, **B** and **C**, Where:

A= the coefficient of the (X) variable, it has to be a **POSITIVE** Integer.

B= the coefficient of the (Y) variable, it has to be an Integer.

C= the constant, it has to be an Integer.

In case your equation is missing one or more coefficients, you can always enter zeros instead, But if your Equation contains decimal and/or fractional coefficients, it means it is not in Standard form, so you can always Standardize it first, by going to the main menu and select the "E" option [**Equation Fixer**], then come back and enter it. Once you're done entering your data, press [**ENTER**] to see if to see the intercepts.

(X) & (Y) Intercepts for an Equation in Standard Form: $y=mx+b$

The program assumes that you have an Equation in **Slope-Intercept form**: $y=mx+b$ and you want to find out what the (X) & (Y) Intercepts for this particular line are, to determine this, you'll need to enter the numeric values for coefficients: (**m**) and (**b**), where:

m= the Slope of your line.

b= the y-intercept (this is the 'Y' value where your line crosses the **Y-Axis**).

The values for **m** and **b** can be integers or Real numbers. If your equation is missing one or more coefficients, you can always enter zeros instead. Once you're done entering your data, press [**ENTER**] to see the intercepts.

- **Intersection Point.**

Intersection Point between 2 Equations in Standard form: $Ax+By=C$ (Notice the form of the equation, this time is different from the usual $Ax+Bx+C=0$).

The program assumes that you have 2 Equations in **Standard form**: $Ax+By=C$ and you want to find out if these lines intersect each other, to determine that, you'll need to enter the numeric values for coefficients: **A₁**, **B₁**, **C₁**, **A₂**, **B₂** and **C₂**, where:

A₁= the coefficient of the (X) variable from Equation # 1, It has to be a **POSITIVE** Integer.

B₁= the coefficient of the (Y) variable from Equation # 1, It has to be an Integer.

C₁= the constant of equation # 1, It has to be an Integer.

A₂= the coefficient of the (X) variable from Equation # 2, It has to be a **POSITIVE** Integer.

B₂= the coefficient of the (Y) variable from Equation # 2, It has to be an integer.

C₂= the constant of equation # 2, It has to be an Integer.

If your equation is missing one or more coefficients, you can always enter zeros instead, but if your Equation contains decimal and/or fractional coefficients, it means it is not in Standard form, so you can always Standardize it first, by going to the main menu and select the "E" option [**Equation Fixer**], then come back and enter it. Once you're done entering your data, press [**ENTER**] to see if the coordinates of the intersection point (if there's any).

Intersection Point for 2 Equations in Slope-Intercept form: $y=mx+b$

The program assumes that you have 2 equations in **Slope-Intercept form**: $y=mx+b$ and you want to find out if these lines intersect each other, to determine that, you'll need to enter the numeric values for coefficients: m_1 , b_1 , m_2 and b_2 , where:

m_1 = the Slope of the equation # 1.

b_1 = the y-intercept (this is the 'Y' value where the equation # 1 crosses the **Y-Axis**).

m_2 = the Slope of the equation # 2.

b_2 = the y-intercept (this is the 'Y' value where the equation # 2 crosses the **Y-Axis**).

The numeric values for: m_1 , b_1 , m_2 and b_2 can be integers or Real numbers. If your equation is missing one or more coefficients, you can always enter zeros instead. Once you're done entering your data, press **[ENTER]** to see the coordinates of the intersection point (if there's any).

• **Equation FIXER.**

The equation Fixer program was designed to transform something like this: $-0.5x - \frac{3}{5}y + 1.33 = 0$

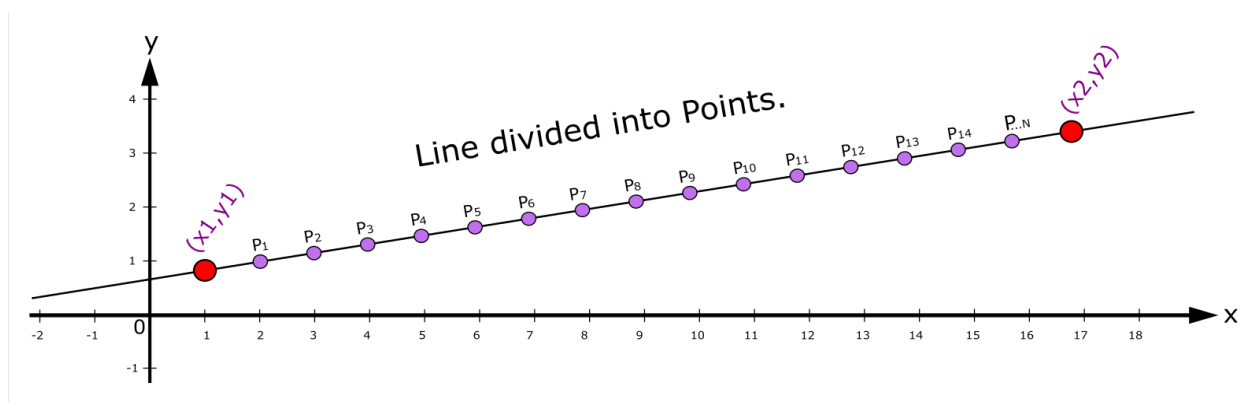
into something like this: **$50x+75y-133=0$** , where both equations are equivalents. So all you have to do is enter the numeric values for coefficients: **A**, **B** and **C** considering that your equation is written as: **$Ax+By+C=0$** . The program will standardize a non-standard form equation into a standardized version of it, this time any coefficient is allowed to be **negative**, **positive**, **decimal** or **fractional**. Once you're done entering your data, press **[ENTER]** to see the resulting equation.

• **Segments.**

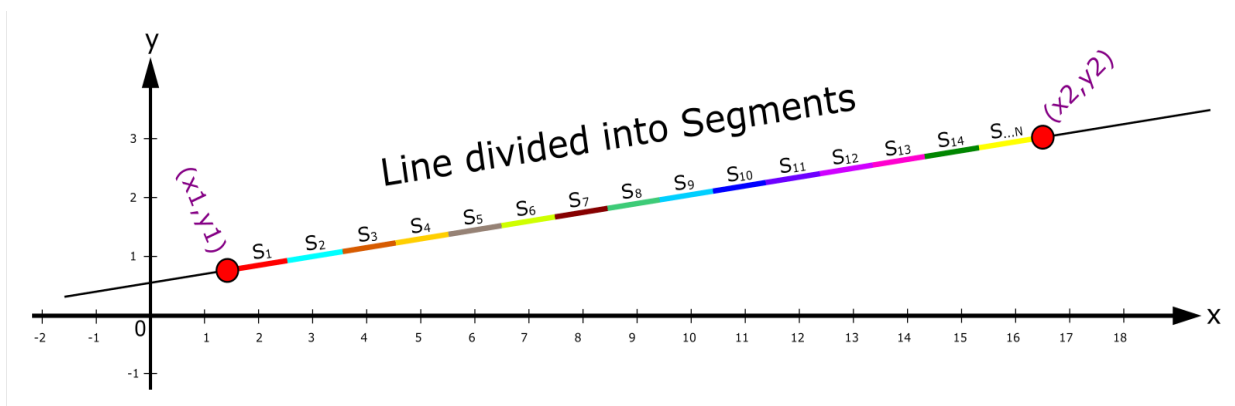
This program assumes that you have 2 points and these points are part of a Straight Line, and you want to divide this line into segments or points equally distributed along the line.

Here you are presented with 2 choices:

- 1.- Divide the Line in "**x**" **number** of points, where both end-points will be included.
- 2.- Divide the Line in "**x**" **number** of points, where both end-points will **NOT** be included.



If you want to divide the line into segments, then you have to enter the number of segments you want to divide the Line into, then press [ENTER] to see the list of coordinates for the segments.



Important. - After you get the list of coordinates, SLK 4.1b will close itself, this is normal... this is the way it was programmed, because it was the only way I found to display large amounts of coordinates and still be able to scroll the list up & down in the terminal screen. Since scrolling in the terminal screen is only available when a program is finished and there's no more code to execute. This is the reason why SLK closes itself after presenting the results.

• **Linear Interpolation/Extrapolation.**

In this section we can do both Interpolations and Extrapolations, just enter the numeric values that you were given and a name for the unknown variable you want to find its value. The numeric values for: x_2 , y_2 , x_1 , y_1 , x_3 and y_3 can be integers or real numbers.

If you have problems naming your unknown variable (it should be lower case though), it may be probably already occupied, so you may want to delete its contents by going to the memory manager [SHIFT]+[B], then come back to this program and now it should be available. If you don't want to delete its contents, you can also use another name for your variable just to get the result.

Linear Interpolation / Extrapolation 22:27

| | |
|-------------|------------|
| x_1 : 0 | y_1 : 2 |
| x_2 : abc | y_2 : 6 |
| x_3 : 8 | y_3 : 10 |

Middle boxes for interpolations:

Real Number or a Lowercase Variable name.

Edit Cancel OK

CAS Terminal

The missing value is:
abc = 4

Linear Interpolation / Extrapolation 22:25

| | |
|------------|-------------|
| x_1 : 2 | y_1 : 4 |
| x_2 : 6 | y_2 : 8 |
| x_3 : 10 | y_3 : xyz |

these boxes for extrapolations:

Real Number or a Lowercase Variable name.

Edit Cancel OK

CAS Terminal

The missing value is:
xyz = 12

Special greetings to:

- [Erwin Ried](#) the creator of PrimePad an application that I found so useful when developing SLK in a smooth way, otherwise it would have taken me much more time to achieve such task.
- [Ing. Carlos Ismael Campos Guerra](#) the creator of “**TECLAS DEDICADAS**” and many more programs, thanks for helping me out to understand the HP Prime graphics environment and all his great support.
- [Ing. Pedro](#) from Estructurapps for sharing his programs and tutorials, they have helped me a lot.
- [Han](#) the creator of **GRAPH 3D v2.422** for creating very useful tutorials I used to improve SLK.

VERSION HISTORY:

September 2018 (SLK 4.1b English) 279kb. Tested **ONLY** on Firmware version: **13865**.

- *Fix some graphical issues with the intro and other sections due to the last 2 FW updates.*
- *A couple of corrections to this manual.*
- *New color theme.*

January 2017 (SLK 4.1 English) 278kb. Tested on Firmware versions: **10637** and **10638**.

- A new section called: **Linear Interpolation/Extrapolation** was added!
- A bug in the **$y=mx+b \Rightarrow Ax+By+C=0$** section was fixed.
- SLK 4.1b User manual minor additions.

December 2016 (SLK 4.0 English) 274kb. Tested on Firmware versions: **10637** and **10638**.

- Now SLK 4.1b comes as an App.
- Graphic Help was added to each section.
- A new section called: **SEGMENTS** was added!
- A new welcome screen was added.
- A bug in the **Equation Transformation** section was fixed.
- SLK 4.1b User manual minor additions.

April 2016 (SLK 3.3 English) 173kb. Compatible with Firmware version: **10077**.

- SLK3.3 **NO longer** works on earlier versions of firmware.
- More redundant code was removed.
- Minor corrections to SLK 3.3 User manual.

February 2016 (SLK 3.2 English) 173kb [It works only on FW: 8151 or earlier].

- File size was optimized.
- A bug regarding Parallel Lines [Fixed].
- Redundant texts were removed.
- Minor corrections to SLK 3.2 User manual.

November 2015 (SLK 3.1 English) 187kb.

- A little optimization on the file size (about 2.6%)
- A couple of messages were modified for better understanding.
- Code has been modified to improve reliability on the results.
- No more garbage in the terminal screen.
- No more messing with User settings, now User settings are kept untouched.

May 2015 (SLK 3.0 English) 192kb.

- Introducing (SLK 3.0).

For comments & bug reports please *E-mail* me to: dantecamargosilva01@gmail.com