

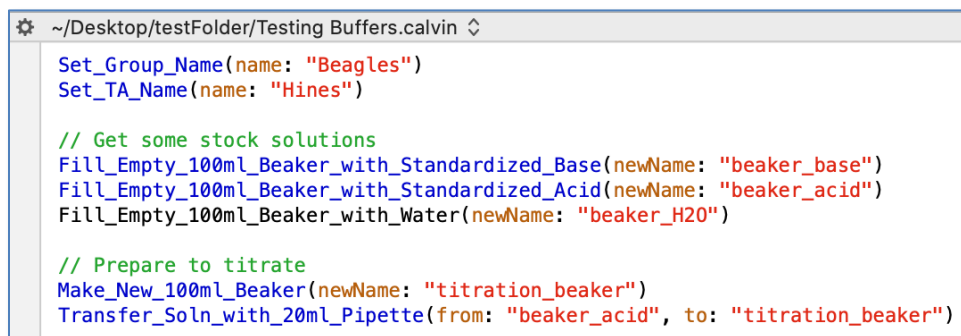
Working with Calvin: Your Online Lab Assistant

To help you complete the three acid-base experiments, the department has hired Calvin, your new online lab assistant. Your group will submit experiments to Calvin in a text file by e-mail. He will perform the experiments and upload his results to your group's folder in Box. Unfortunately, Calvin knows very little chemistry, so you have to tell him exactly what you want him to do.

Getting ready

Since Calvin only understands text files,¹ we highly suggest you write your commands to him in a text editor designed for programming, not a word processor. Please go to the link on Canvas or turn to the last page of this document for instructions on downloading and setting up a free text editor that works on your computer.

When writing to Calvin, add a `.calvin` extension to your filename to enable syntax coloring as shown below. The exact colors you see will depend on your text editor and how you set your preferences. In the example shown below, commands that Calvin understands are in blue, prompts are in brown, and literal names, which must be enclosed in quotes, are in red. (Literal names are often called “strings” in programming.) Notice that `Fill_Empty_100ml_Beaker_with_Water` is black, which implies that Calvin does not understand this command. The command should read `Fill_Empty_100ml_Beaker_with_H2O`. Comments, which Calvin ignores, always start with two slashes `//` and are colored in green below.



```
~/Desktop/testFolder/Testing Buffers.calvin ⌵
Set_Group_Name(name: "Beagles")
Set_TA_Name(name: "Hines")

// Get some stock solutions
Fill_Empty_100ml_Beaker_with_Standardized_Base(newName: "beaker_base")
Fill_Empty_100ml_Beaker_with_Standardized_Acid(newName: "beaker_acid")
Fill_Empty_100ml_Beaker_with_Water(newName: "beaker_H2O")

// Prepare to titrate
Make_New_100ml_Beaker(newName: "titration_beaker")
Transfer_Soln_with_20ml_Pipette(from: "beaker_acid", to: "titration_beaker")
```

Naming Things for Calvin

To help Calvin keep track of your experiment, you need to name all of your glassware. In the commands above, the beakers have names `beaker_base`, `beaker_acid`, and so forth. These names must be surrounded by quotes. If you forget the quotes, the text editor will turn the name black (or whichever color your editor uses) to indicate that Calvin will not understand the name.

Calvin only accepts names that start with a letter and contain letters, numbers, or the underscore. Calvin *does not* allow spaces in names. Calvin thinks capitalization is irrelevant, but it is good practice to improve the readability of your experiments by using namesInCamelCase or names_like_this.

Valid names: `myBeaker`, `Flask_1`, `The_splendiferous_buffer`

Invalid names: `1`, `3_Beaker`, `Beaker_15.2`, `!#@&*`

¹ Text files contain only plain text, no formatting or other information, and are stored as a sequence of characters. They can have many different file extensions, including `.txt` and `.calvin`. Word processors typically store their documents as binary files. Common binary file extensions are `.doc` and `.docx`. If you really want to use a word processor, you must save your file as plain text, which will typically lead to a `.txt` extension.

Writing Experiments for Calvin

The easiest way to design experiments for Calvin is to have two files open at once: your experiment file and the file of Calvin command templates, `Command_File.calvin`. This will allow you to simply copy individual commands and fill them out with your instructions.

The current list of commands that Calvin understands are:

```
// Names start with a letter and can contain letters, numbers and _. No spaces!

// Identifying your experiments
Set_Group_Name(name: "groupName") // Determines where output goes
Set_TA_Name(name: "your_TAs_Name") // Determines where output goes

// Labeling fresh, clean glassware
Make_New_100ml_Beaker(newName: "beakerName")
Make_New_50ml_Volumetric_Flask(newName: "flaskName")

// Filling beakers with stock solutions
Fill_Empty_100ml_Beaker_with_H2O(newName: "beakerName") // pH varies realistically, not 7.00
Fill_Empty_100ml_Beaker_with_Standardized_Base(newName: "beakerName") // Reports exact concentration
Fill_Empty_100ml_Beaker_with_Standardized_Acid(newName: "beakerName") // Reports exact concentration
Fill_Empty_100ml_Beaker_with_Standardized_Buffer(newName: "beakerName") // Reports exact concentration
Fill_Empty_100ml_Beaker_with_Unknown_Buffer(buffer: "bufferName", to: "beakerName")

// Cleaning vessels after use
Clean_And_Dry(name: "vesselName") // After cleaning, the name is removed from the vessel

// Transferring solutions between vessels
Transfer_Soln_with_20ml_Pipette(from: "vesselName", to: "vesselName")
Transfer_Soln_with_Graduated_Cylinder(mL: volume, from: "vesselName", to: "vesselName")
Fill_Volumetric_Flask_with_H2O(name: "flaskName")

// Working with indicators
Add_One_Drop_of_Indicator(indicator: "indicatorName", to: "vesselName") // Don't mix indicators!

// Working with solid acids
Add_Solid_Acid_to_Vessel(grams: targetMass, acid: "acidName", to: "vesselName") // Reports exact mass

// Working with the buret
Fill_50ml_Buret(from: "vesselName") // Cleans buret before filling
Add_Soln_from_Buret(mL: targetVolume, to: "vesselName") // Must read volume before and after!
Read_Buret_Volume()

// Making measurements
Measure_pH(of: "vesselName")
Observe_Color(of: "vesselName") // reports single colored box
Take_Spectrum(of: "vesselName") // reports spectrum and exports spectrum as .csv
Observe_Color_Range(of: "indicatorName", at: relativeConcentration) // 1 drop/50 mL = 1.0 conc

// Performing a titration
Titrate_Beaker_from_Buret_until_Color_Change(into: "vesselName") // reports buret readings, color observed

// Other commands
Observe_Volume(name: "vesselName") // Reports rough volume measurement
```

Communicating with Calvin

First and most importantly every experiment file that you submit to Calvin must set your group name and your TA name using the commands `Set_Group_Name` and `Set_TA_Name`. I suggest making these the first two commands in every experiment. These names will be given to you by your TA. Second, give your experiment file an identifiable name, such as `Experiment_1.calvin`.

Important: Do not submit any experiments to Calvin until your TA gives you a group name and grants your group access on Box. You will not be able to access your output until then.

When you are ready to submit your file to Calvin, e-mail your file as an attachment to:

Chem_20.rpumm9z0d7ba31yz@u.box.com

The subject of the e-mail and any text in the body will be ignored. Only the attachment will be uploaded. You should receive an acknowledgment from Box within a few minutes of your submission.

Once Calvin completes your experiment, he will put his notebook and any generated spectra in a folder. He will compress (*i.e.*, zip) the folder and upload it to your group's folder on Box. For example, a typical output from Calvin would be named

Beagles_Testing Buffers_03-24_195948.zip

This is the output generated for the Beagles group from their `Testing Buffers.calvin` experiment which was performed on March 24 at 19:59:48. You can copy this file from Box to your own computer for analysis. On most computers, double clicking a zip file will decompress it.

A Sample Experiment

The following is a simple experiment where Calvin titrates the stock acid solution with the stock base solution to make sure everything is working. Read through the experiment to get a feeling for Calvin operations.

```
// Identifying our experiments
Set_Group_Name(name: "Weimaraners")
Set_TA_Name(name: "Caserto")

// Getting stock solutions
Fill_Empty_100ml_Beaker_with_Standardized_Base(newName: "stock_base")
Fill_Empty_100ml_Beaker_with_Standardized_Acid(newName: "stock_acid")

// Making acid solution for titration
Make_New_100ml_Beaker(newName: "titration_beaker")
Transfer_Soln_with_20ml_Pipette(from: "stock_acid", to: "titration_beaker")
Add_One_Drop_of_Indicator(indicator: "Thymol_blue", to: "titration_beaker")

// Get the buret ready
Fill_50ml_Buret(from: "stock_base")

// Performing the titration
Titrate_Beaker_from_Buret_until_Color_Change(into: "titration_beaker")
```

Reading Calvin's Experiment Log

After Calvin performs your experiment, he will save his log (*i.e.*, notebook) as a rtf (rich text format) file in your group's folder on Box. **Do not try to read this file in a text editor, as you will not see Calvin's graphics.** Open the file in a word processor.

For example, the complete output from the sample experiment above is shown on the next page.

Calvin command file: Simple Titration.calvin**Output file: Weimaraners_Simple Titration_03-25_223827**

// Identifying our experiments

Set_Group_Name(name: "Weimaraners")

Set_TA_Name(name: "Caserto")

// Getting stock solutions

Fill_Empty_100ml_Beaker_with_Standardized_Base(newName: "stock_base")

A clean 100 ml beaker was named stock_base.

The beaker stock_base was filled with 100 ml of 0.1030 M standardized NaOH.

Fill_Empty_100ml_Beaker_with_Standardized_Acid(newName: "stock_acid")

A clean 100 ml beaker was named stock_acid.

The beaker stock_acid was filled with 100 ml of 0.1003 M standardized HCl.

// Making acid solution for titration

Make_New_100ml_Beaker(newName: "titration_beaker")

A clean 100 ml beaker was named titration_beaker.

Transfer_Soln_with_20ml_Pipette(from: "stock_acid", to: "titration_beaker")

Pipetted 20 ml of solution from stock_acid to titration_beaker.

Add_One_Drop_of_Indicator(indicator: "Thymol_blue", to: "titration_beaker")

Added one drop of Thymol_blue to titration_beaker.

// Get the buret ready

Fill_50ml_Buret(from: "stock_base")

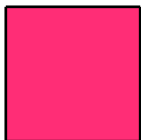
Added solution from stock_base to your 50 ml buret.

The volume of solution in the buret is now 50.00 ml.

// Performing the titration

Titrate_Beaker_from_Buret_until_Color_Change(into: "titration_beaker")

The color of the solution in titration_beaker is:

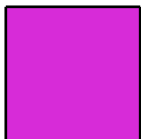


The volume of solution in the buret is now 50.00 ml.

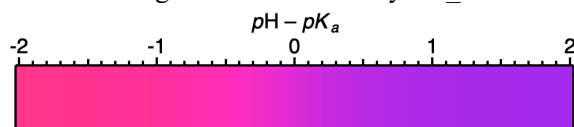
Performing titration now.

The volume of solution in the buret is now 30.50 ml.

The color of the solution in titration_beaker is:



The color range of the indicator thymol_blue is:



Good Programming Practice.

Calvin is the ultimate honey badger.² He don't care how you write your experiment or the names you choose.

You, on the other hand, are going to need to understand your experimental output, perhaps days or weeks after you write your experiment file. For this reason, we strongly suggest that you liberally comment your experiment, describing what you are trying to do. If you put a comment on its own line (*i.e.*, starting the line with //), Calvin will copy your comment in green into his log.

How Calvin Handles Errors

Calvin will try his best to follow your instructions, but he will write **red errors** in his log if your instructions lead to problems in the lab.

Calvin is a trooper. He will continue to perform your commands no matter how many errors are generated. This can be problematic if an error leads to a piece of glassware being cleaned and put away, as illustrated by the output from the unhappy experiment below. Notice that overfilling myFlask caused Calvin to clean and dry the spilled flask, which erased the name of the flask. As a result, myFlask did not exist when Calvin tried to measure its volume in the next step.

```
// Get a beaker of stock base solution
Fill_Empty_100ml_Beaker_with_Standardized_Base(newName: "base_beaker")
A clean 100 ml beaker was named base_beaker.
The beaker base_beaker was filled with 100 ml of 0.1001 M standardized NaOH.
// Giving two beakers the same name is bad
Make_New_50ml_Volumetric_Flask(newName: "myFlask")
A clean 50 ml volumetric flask was named myFlask.
Make_New_50ml_Volumetric_Flask(newName: "myFlask")
I'm sorry. A vessel by the name of myFlask already exists.
// Adding a negative mass is bad
Add_Solid_Acid_to_Vessel(grams: -1.2, acid: "KHP", to: "myFlask")
You cannot transfer a negative mass!
// Overfilling a vessel makes a mess
Transfer_Soln_with_Graduated_Cylinder(mL: 75, from: "base_beaker", to: "myFlask")
Used a 25 ml graduated cylinder to transfer 25 ml of solution from base_beaker to myFlask.
Used a 25 ml graduated cylinder to transfer 25 ml of solution from base_beaker to myFlask.
Oh, no! You put too much solution in myFlask. What a mess! Clean and dry myFlask.
// Check the volume of the overfilled flask
Observe_Volume(name: "myFlask")
I'm sorry. The vessel myFlask does not exist.
```

There is one type of error that you may not expect. Calvin will not allow you to name a beaker or flask after common programming terms (*e.g.*, print, exists, variable). This will generate a regular error as well as a “Bad Command Ignored” error as illustrated by the following:

```
The name exists conflicts with Calvin. Please use a different name.
Bad Command Ignored: Fill_Empty_100ml_Beaker_with_Standardized_Base(newName:
"exists")
```

² <https://www.youtube.com/watch?v=4r7wHMG5Yjg>, <https://www.youtube.com/watch?v=c36UNSoJenI>

Working with Colors and Spectra

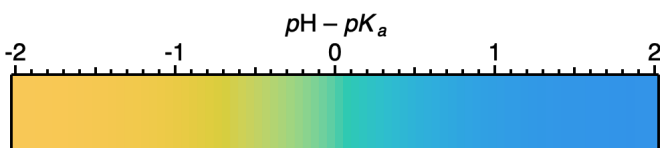
If your solution contains an indicator dye, Calvin can tell you what color your solution is, and he can acquire the solution's visible spectrum. In doing this, the concentration of the indicator should be approximately 1 drop of dye per 50 ml, although this varies somewhat by dye. If the solution is too concentrated, the colors will be too saturated, and the spectrum may be clipped. Conversely, if the solution is too dilute, the colors will be faint and the spectrum noisy.

You can see the effect of concentration most easily using the `Observe_Color_Range` command as shown by the following experiment which examines bromocresol green at three concentrations: just right, too dilute, and too concentrated.

```
// Examine the color range at the optimum concentration of 1 drop/50 ml
```

```
Observe_Color_Range(of: "bromocresol_green", at: 1.0)
```

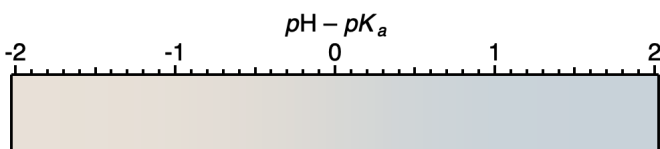
The color range of the indicator bromocresol_green is:



```
// Examine the color range at 1/10 of optimum concentration
```

```
Observe_Color_Range(of: "bromocresol_green", at: 0.1)
```

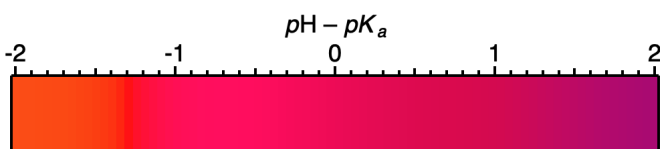
The color range of the indicator bromocresol_green is:



```
// Examine the color range at 10 times optimum concentration
```

```
Observe_Color_Range(of: "bromocresol_green", at: 10.0)
```

The color range of the indicator bromocresol_green is:



Why does the solution look red at high concentration? Simple! The solution is essentially black at all other wavelengths, so the tiny amount of light that is transmitted is red.

Why does the solution look gray at low concentration? Equally simple. All colors are being transmitted (none absorbed), so the solution appears to be a neutral gray.

The output file on the next page illustrates the same problem, only this time with visible spectra instead of colors.

`Fill_Empty_100ml_Beaker_with_Standardized_Base(newName: "stock_base")`

A clean 100 ml beaker was named stock_base.

The beaker stock_base was filled with 100 ml of 0.0976 M standardized NaOH.

`// Make a solution at the optimum concentration`

`Make_New_100ml_Beaker(newName: "optimum_concentration")`

A clean 100 ml beaker was named optimum_concentration.

`Transfer_Soln_with_Graduated_Cylinder(mL: 50.0, from: "stock_base", to: "optimum_concentration")`

Used a 25 ml graduated cylinder to transfer 25 ml of solution from stock_base to optimum_concentration.

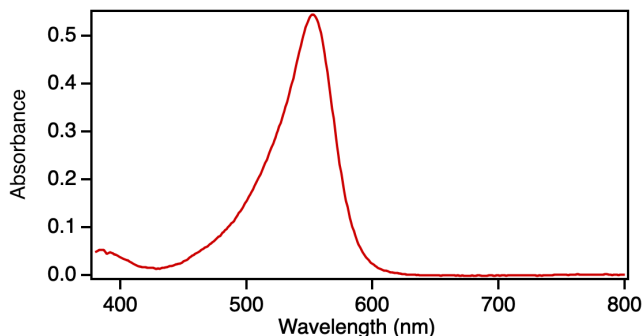
Used a 25 ml graduated cylinder to transfer 25 ml of solution from stock_base to optimum_concentration.

`Add_One_Drop_of_Indicator(indicator: "Phenolphthalein", to: "optimum_concentration")`

Added one drop of Phenolphthalein to optimum_concentration.

`Take_Spectrum(of: "optimum_concentration")`

The spectrum of the solution in optimum_concentration is:



This spectrum has been saved as Spectrum_1.csv

`// Make a solution at 10 times the optimum concentration`

`Make_New_100ml_Beaker(newName: "too_concentrated")`

A clean 100 ml beaker was named too_concentrated.

`Transfer_Soln_with_Graduated_Cylinder(mL: 5.0, from: "stock_base", to: "too_concentrated")`

Used a 10 ml graduated cylinder to transfer 5 ml of solution from stock_base to too_concentrated.

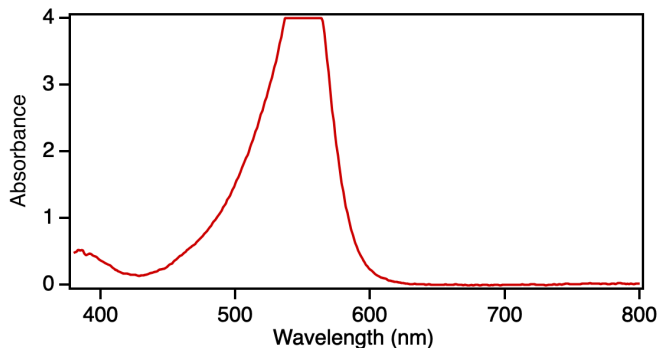
`Add_One_Drop_of_Indicator(indicator: "Phenolphthalein", to: "too_concentrated")`

Added one drop of Phenolphthalein to too_concentrated.

`Take_Spectrum(of: "too_concentrated")`

Your solution is too concentrated! The spectrum is being clipped at some wavelengths.

The spectrum of the solution in too_concentrated is:



This spectrum has been saved as Spectrum_2.csv

Asking Calvin to Use a Buret or Perform a Titration

Calvin can use a buret in two different ways. First, he can transfer a relatively precise amount of solution using a buret. Second, he can titrate a solution until a color change is observed, assuming of course that you have added an indicator to the solution. These tasks are illustrated by the output below, which we will examine in two parts. (Some of the output has been replaced by ... for brevity.)

In the first part of the experiment, Calvin tries to add a very small amount of base to 20 ml of H₂O, which generates some interesting results:

```
// Fill the buret with stock base solution and get some H2O
Fill_Empty_100ml_Beaker_with_Standardized_Base(newName: "stock_base")
...
Fill_50ml_Buret(from: "stock_base")
...
Fill_Empty_100ml_Beaker_with_H2O(newName: "pure_H2O")
...

// Fill a beaker with 20 ml H2O, measure pH, add tiny amount of base, measure pH
Make_New_100ml_Beaker(newName: "dilute_base")
A clean 100 ml beaker was named dilute_base.
Transfer_Soln_with_20ml_Pipette(from: "pure_H2O", to: "dilute_base")
Pipetted 20 ml of solution from pure_H2O to dilute_base.
Measure_pH(of: "dilute_base")
The pH of dilute_base is 6.14
Read_Buret_Volume()
The volume of solution in the buret is now 49.20 ml.
Add_Soln_from_Buret(mL: 0.0001, to: "dilute_base")
Read_Buret_Volume()
The volume of solution in the buret is now 49.15 ml.
Measure_pH(of: "dilute_base")
The pH of dilute_base is 10.26
```

The first surprise in this experiment is the first *pH* measurement. Even though there was only “pure” H₂O in the beaker, the *pH* was 6.14. This is typical of an actual lab, where the *pH* of pure H₂O can range from ~6 due to dissolved CO₂ which acidifies the H₂O to ~7.5 from trace contaminants in the pipes.

The second surprise is that even though you asked Calvin to add 0.0001 ml of base, he actually added 0.05 ml. There are two problems. First, there is a limit as to how small of an amount you can add with a buret, typically ~1/20 ml (~one drop). The second problem is that 50 ml burets can only be read to an accuracy of 0.05 ml. When you ask Calvin to add a tiny amount from a buret, he will always add something. That change in volume may or may not be too small to read on the buret.

Finally, even though you added a tiny amount of base to the H₂O, the *pH* skyrocketed by more than 4 *pH* units! This is to be expected, as a even a single drop of ~0.10 M base has a significant number of moles of OH⁻.

The experiment continues on the next page.


```
// Fill a beaker with 20 ml H2O, measure pH, add tiny amount of base, measure pH
```

```
Make_New_100ml_Beaker(newName: "titration_beaker")
```

A clean 100 ml beaker was named titration_beaker.

```
Transfer_Soln_with_20ml_Pipette(from: "pure_H2O", to: "titration_beaker")
```

Pipetted 20 ml of solution from pure_H2O to titration_beaker.

```
Add_One_Drop_of_Indicator(indicator: "thymol_blue", to: "titration_beaker")
```

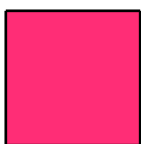
Added one drop of thymol_blue to titration_beaker.

```
Measure_pH(of: "titration_beaker")
```

The pH of titration_beaker is 6.14

```
Titrate_Beaker_from_Buret_until_Color_Change(into: "titration_beaker")
```

The color of the solution in titration_beaker is:

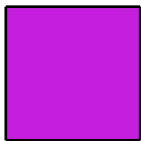


The volume of solution in the buret is now 49.15 ml.

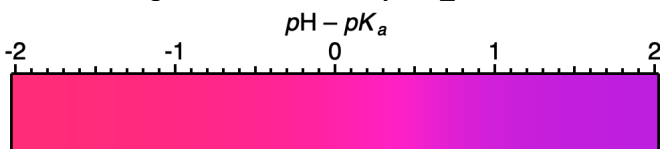
Performing titration now.

The volume of solution in the buret is now 49.10 ml.

The color of the solution in titration_beaker is:



The color range of the indicator thymol_blue is:



```
Measure_pH(of: "titration_beaker")
```

The pH of titration_beaker is 10.58

In the second experiment, we added an indicator, thymol blue ($pK_a = 9.20$), and had Calvin perform a titration. In principle, Calvin should have added just enough base to reach a pH of ~ 9.2 , but you can see that he overshot the mark by over one pH unit and ended up at a pH of 10.58. Again, the problem wasn't Calvin. He was working at the limit of his glassware and his eyesight. The problem is that the pH of a titration changes drastically when the titration is near the equivalence point.

Is Calvin a Good Lab Assistant?

Calvin almost assuredly performs your experiments more accurately than you could. Nevertheless, he is limited by the accuracy of his glassware and his equipment. Even the best of chemists would struggle to transfer one drop of solution accurately and reproducibly.

You should also realize that the pH of "pure H₂O" — the stuff that comes out of real faucets — fluctuates with time. The pHs of standardized solutions also change when new carboys are prepared. As a result, the precise pH values of H₂O and the standardized solutions will typically change with each experiment file.

Calvin's Glassware Supply

Calvin has an essentially endless supply of 50 ml volumetric flasks and 100 ml beakers. He also has a 20 ml pipette and both a 10 and 25 ml graduated cylinder. (The 10 ml graduate cylinder is more accurate than the 25 ml graduated cylinder, but Calvin will automatically use the larger cylinder for volumes exceeding 10 ml.) Calvin has one 50 ml buret.

Calvin washes and dries his measuring glassware and buret between uses, so you don't need to tell him to do this. If you would like to refill a vessel with fresh solution, `Clean_and_Dry` the vessel, then make a new vessel with the old name.

Calvin's Supply of Known and Unknown Chemicals

Indicators: Calvin has four known indicators with properties described in your textbook: `Methyl_orange`, `Thymol_blue`, `Phenolphthalein`, and `Bromocresol_green`. Your TA will also assign you a named unknown indicator. Unknown indicators have names of the form `Dye_of_noun`.

Solid acids: Calvin has one known solid acid which can only be used for the Unknown Solid Acid experiment. That acid is named KHP, which is an abbreviation of potassium hydrogen phthalate. Your TA will also assign you a named unknown acid. Unknown acids have names of the form `Acid_of_DisneyCharacter`.

Buffers: Calvin has one known buffer which can only be used for the Unknown Buffer experiment. That buffer is accessed through the `Fill_Empty_100ml_Beaker_with_Standardized_Buffer` command. Calvin will inform you of the concentration of the standardized buffer when you run your experiment. Your TA will assign you a named unknown buffer. Unknown buffers have names of the form `Buffer_of_FictionalCharacter`.

TA Names and Group Names

Your TA will tell you what form of their name to use. Some TAs will use their first name, others their last. Your TA will also assign you to a group. Group names are always a plural dog breed, such as the Weimaraners.

Problems with Calvin?

If your group name and/or TA name are set incorrectly or misspelled, your output will be sent to `_Lost & Found`. Your TA may be able to find it, but it would be much faster for you to just spell things correctly and resubmit.

If you think that Calvin may be sick or sleeping, submit a two-line test file that sets your group name and TA name. For example:

```
// Identifying your experiments
Set_Group_Name(name: "Weimaraners")
Set_TA_Name(name: "Caserto")
```

If you are sure everything is OK with your group and TA names and Calvin has not responded within a reasonable period of time, please send an e-mail to Melissa.Hines@cornell.edu.

For the Cognoscenti

Calvin is a custom interpreted language implemented in *Igor Pro* and running on an iMac. All of the data used by Calvin are real and were collected in the undergraduate labs by Prof. Hines shortly before Cornell went on lock down. All of the colors generated by Calvin are calculated from actual spectra of indicators using the CIE 1931 color space. A general introduction to this conversion can be found at <https://www.fourmilab.ch/documents/specrend/>.

The name Calvin is an homage to Python, which is an interpreted language named after *Monty Python's Flying Circus*. Python was written by engineer Guido von Rossum out of boredom while his research lab was closed over Christmas week in 1989. Boredom was not our primary motivator in writing Calvin, but we are nevertheless cooped up at home.

Igor Pro is an outstanding graphing and data analysis package for scientists and engineers. Igor is, of course, Dr. Frankenstein's lab assistant and also Calvin's supervisor. For more information about *Igor Pro*, visit WaveMetrics at <https://www.wavemetrics.com/>. Any piece of software that can enable a chemistry professor to write and deploy a new programming language around the world in a couple of weeks is wicked cool in my book.

— Melissa, the programmer, and Hobbes, the golden retriever

Copyright 2020, Melissa A. Hines

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Finding a Text Editor and Setting It Up for Calvin

Mac users: We suggest you download the *free version* of BBEdit from the link below. Do not purchase a license, as you do not need the extra bells and whistles.

<https://www.barebones.com/products/bbedit/download.html>

After downloading and installing BBEdit, go to BBEdit > Folders > Language Modules. This will open a folder in Finder. Put the file `CalvinLanguageModule.plist` in this folder, then restart BBEdit.

When writing to Calvin, you can either add a `.calvin` extension to your filename or set the popup menu at the bottom of your editing window to Calvin.

Windows users: We suggest you download Sublime Text from the link below. It is free for evaluation purposes, but will nag you about purchasing after some period of time.

<https://www.sublimetext.com/>

After downloading and installing Sublime Text, go to Preferences > Browse Packages... which will open a folder on your computer. Put the file `calvin.sublime-syntax` in the User folder, then restart Sublime Text.

You can change the color scheme using Preferences > Color Scheme... If you don't care for the installed color schemes, go to Open Tools > Command Palette... Start typing Package Control: Install Package. When that option appears in the palette, click it. Wait for the available packages to show up, and then start typing Github Color Theme. When that option appears in the palette, click it to install the theme. Go to Preferences > Color Scheme... to select the scheme.

When writing to Calvin, add a `.calvin` extension to your filename to enable syntax coloring.

Sublime Text is also available for Mac and Linux.

Ubuntu/Linux users: We suggest you use Gedit, which is the standard GUI text editor on Ubuntu and comes pre-installed with most installations. You can find more information about Gedit as well as installation instructions at the following link:

<https://help.ubuntu.com/community/gedit>

You should then add `calvin.lang` to `/usr/share/gtksourceview-3.0/language-specs`.

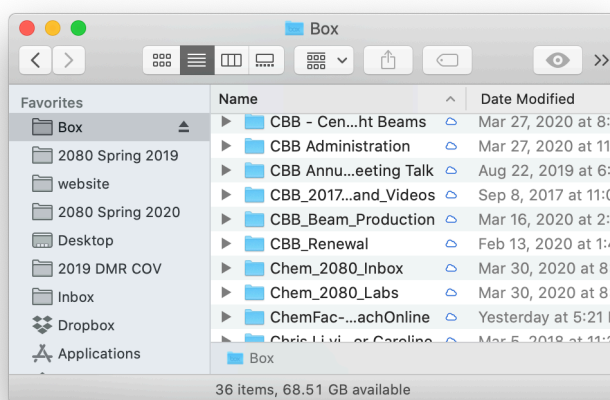
When writing to Calvin, add a `.calvin` extension to your filename to enable syntax coloring.

Using Box

Box is a cloud-based storage system. All Cornell faculty, staff, and students have an unlimited amount of storage on Box. Folders and files can be shared across multiple users. There are three ways of accessing your files on Box.

Browser: Go to <https://cornell.box.com/>

Computer: If you install Box Drive (not Box Sync, which is no longer recommended!), you can access your Box files like any other file on your computer. Go to <https://www.box.com/resources/downloads/drive> and download the installer for Mac or Windows. See the image below for an example.



Box in China: We think that Box is accessible in China at <https://app.boxcn.net/>, but we (obviously) have no way to test this. If you are having difficulty, this may be a good place to start:

<https://community.box.com/t5/Archive-Forum/Accessing-Box-from-China/td-p/56552>