BIOS 10603 **Lab-2**

**An Introduction to Tcl and VMD, Part Two**

**NAME:**

**Important sections are written in blue. Questions to be answered are in green.**

Please insert all answers and figures into this Word document, and then save the completed lab as “Lastname\_Lab2.docx”. When you have completed the lab, submit it as an attachment to the Lab-2 assignment link on Canvas. The report is due by midnight the day after your lab period.

**Goals:**

In this week’s lab, you will learn about the basics of Tcl scripting that is used by VMD for analysis. You will also build your first system for a molecular dynamics simulation. **The main question that we will try to answer is: “can we create a realistic model of water?”.** We will create a water box using VMD this week and will simulate its dynamical behavior next week.

You will be asked to include screenshots of your VMD window and scripts. We are looking for pictures that are as uncluttered as possible, but still display what we are asking for. Please make sure your attached pictures are large enough to be clear, and that you follow these additional directives:

* Screenshots of code or of a script must be visible and legible
* Specific parts of the protein being focused on should be highlighted via coloring methods
* The entire protein should be included in the picture in order to give context to highlighted parts of the protein (unless it is explicitly listed to just show a specific part of the protein).
* **All of your pictures should be taken in orthographic mode with a white or other bright background colors**

In addition to changing the drawing method in VMD (New Cartoon, Licorice, Lines, etc.), it is also useful to use different materials (Opaque and Transparent) and change the coloring method:

* “Name” – color **red** is **oxygen**, **blue** is **nitrogen**, **cyan** is **carbon,** and **yellow** is **sulfur**
* “ResType” – color **white** is **non-polar**, **green** is **polar**, **red** is **acidic**, and **blue** is **basic**
* “Secondary Structure” – color **magenta** is **alpha-helix**, and **yellow** is **beta-sheets**

**Part 1 – Introduction to Tcl Scripting**

VMD includes support for Tcl scripting; anything that can be done in VMD’s graphical user interface can also be done in Tcl via the Tk Console. Under “Extensions”, select “Tk Console”; this opens a new window where you can type Python and Tcl commands. Code input lines in the lab will be marked with %, while output lines will be displayed in this font without anything preceding the line. **Do not enter the percent sign into the Tk Console!** Some examples are

% set variable value #assign value to variable

% puts $variable #output the value of $variable. Note the $ sign.

% expr {expression} #evaluate the mathematical expression. {} is not necessary.

You can use the up arrow to copy what you entered in a previous line to your current line.

Then you need to learn how to use different kinds of braces.

1. First and foremost, it important to remember that everything you type as a Tcl command is essentially a string, despite that they may look like numbers, etc.
2. All kinds of braces have the effect of giving the content inside priority upon evaluation – just what you imagine braces would do, like the braces in mathematical expressions.
3. Square brackets [] replace the embraced expression with the value it returns, i.e., force its evaluation.
4. Curly braces {} group the entire expression inside. It is necessary when your expression has spaces. Pay attention to the following example which shows that the only curly braces do is grouping its content.
5. Double quotes “” perform almost the same functionalities as curly braces, but with one thing more: they also substitute the value for the variable. Note that it is easy to expect curly braces to be a string-marker; however, because Tcl treats everything as a string, the true functionality of double quotes has nothing to do with “stringifying.” For example, the codes puts “Hello!”, puts {Hello!}, and puts Hello! have the exact same result. Nonetheless, it will make the code more readable if you enclose what you expect to be a string with double quotes.

For example:

% set myvar 5

5

% set x 3

3

% expr {$myvar \* $x}

15

% set result {7 + $myvar \*\* $x}

7 + $myvar \*\* $x ($result does not store a number!)

% set result [expr {7 + $myvar \*\* $x}]

132

% puts “The value of result is: $result”

The value of result is: 132

**Q1.** Suppose we type the following into the console:

% set mystr1 “fobar”

fobar

% puts “My string is: mystr1”

My string is: mystr1

Why does the command in the second line print out “mystr1” as opposed to the string we assign to the variable mystr1 (“fobar”)? Explain.

**Q2.** Write a few lines of code that print a **variable string** and the **number of bits** that string takes up in memory. Use the command string length to compute the length of a string. The number of bits that a string takes up in memory is equal to the **string length + 1, all multiplied by 8.** The **eval** command will be useful for doing this computation. The output should be in the following format:

*My string is: fobar. It takes up 48 bits in memory.*

Paste your code below:

**Navigating the directory structure**

The default directory that the Tk Console opens in may not be the directory your files are in. The current directory is listed in parentheses to the left of the percent sign.

Pwd shows the absolute path of the current directory

cd <path> changes the current working directory

ls lists the contents of the current working directory.

On Linux and OS X, the only argument is the path name to which you want to navigate. For example, if you are already in the directory jsmith” valid commands using Mac would include:

% cd /users/jsmith/documents/bio

or

% cd ~/documents/bio

On Windows, you need to enclose paths with quotation marks, as shown below:

% cd "C:\Users\John Smith\My Documents\bio"

**We strongly recommend changing your working directory to where your files are stored.**

**Writing to a File**

To write to a file, you first need to associate a variable with the file, and then you can use the **puts** command to write to it.

% set file1 [open "testfile.txt" w]

% puts $file1 "This is the first line."

% puts $file1 "This is the second line."

% close $file1

The first line opens a file named testfile.txt, and tells the console that we will be writing to the file (hence the **w** in the second argument). This file is then assigned to the variable **$file1**. The following two lines use the **puts** command to write to the file. By default, the puts command adds a newline to the end of each statement outputted. Finally, the **close** command will close and save the file. **Closing the file is critical**. Failure to close the file may result in lines not being written to the file, among other potential issues.

**For Loops**

For loops allow you to repeat an action until a pre-determined stopping point. In Tcl, the syntax is:

% for {initialization} {test} {increment} {commands}

A variable is initialized, and each time the loop is run, it is incremented. The loop repeats while the test condition remains true, or else it stops execution and moves on to the next line of code after the loop. The following example prints out each number from 1 to 15 on a separate line:

% for {set i 1} {$i <= 15} {incr i} {puts $i}

Notice that the variable passed into the **incr** command is NOT prefixed by a dollar sign**.**

**Q3.** Using a for loop, write a file that writes the numbers 25 through 47 (inclusive) on separate lines, and then print their sum.

Step 1: Open and set outfile

Step 2: Initialize a sum variable

Step 3: Create a for loop that writes number to file and updates the value of $sum

Step 4: Print to file: “the sum of the above numbers is: $sum”

Step 5: What is the important last thing you need to do when done working with a file?

Paste your commands below:

**Useful Tip**

VMD is a memory-intensive software, which might lead to the slowdown of operations and crashes in certain cases. A common issue is accidentally cluttering the Tk Console with large outputs, which can make your VMD very slow. To prevent this issue from occurring, type **clear** in your Tk Console.

clear: clears out the contents of the console

Let's practice utilizing the Tk console and Tcl to make some quick measurements on a protein. Load 2ER0\_autopsf.pdb into VMD and delete all other molecules you may have loaded.

[measure](https://www.ks.uiuc.edu/Research/vmd/current/ug/node138.html) bond {<atom1\_index> <atom2\_index>} measures the distance between two atoms

For example: measure bond {2 10} measures the distance between atom 2 and 10

Note that it may seem weird that curly braces both group math expressions and represents a list (of atoms indices here). Please remember that the only thing that curly braces do is to group the content within. Whether it is parsed into a list, a command, a math expression, or so on depends on its context.

We can obtain the list of atom indexes of an atom selection:

set sel [[atomselect top “all”] list]

Since sel is already a list, there is no need to use {} in measure bond.

**Q4.** Measure the distance between the α-carbons of ASP32 and ASP215.

Step 1) create a list of atom indices containing the α-carbons of ASP32 and ASP215 (you can always verify atom selections by visualizing them using a graphical representation.)

Step 2) measure the distance between the 2 atoms

Paste your commands below:

Report the distance:

There are many other measure commands that perform a similar function to measure bond mentioned above, which includes measure angle and measure dihed. Using the [documentation](https://www.ks.uiuc.edu/Research/vmd/current/ug/node138.html) for these commands and the measure bond example above, answer the following questions:

**Q5**. Measure the dihedral angle ϕ of residue 152. Recall a dihedral angle is an angle between 4 atoms, and the ϕi angle is defined as the angle between C (of residue i-1) – N – Cα – C. Hint: Try using the atom selection keyword name to differentiate between C, N, and CA (aka Cα). Also, just show the backbone of residues 151, 152, and 153 on your screen and zoom in (use the "=" sign) as a guide for your atom selection.

Paste your commands below:

Report angle:

**Q6**. Measure the dihedral angle ψ of residue 152. Recall ψis defined as N – Cα – C – N (of residue i+1).

Paste your commands below:

Report angle:

**Q7**. Do these angles agree with the ranges given by the Ramachandran plot and this residue’s secondary structure? Use the [Godzilla](https://godzilla.uchicago.edu/cgi-bin/rama_all.cgi) website to check for this.

**Q8**. Measure the hydrogen bond distance between residues 227 and 231. Hint: To determine the atom selection, you can visualize the hydrogen bond by using the HBonds drawing method, setting the distance cutoff to 3.5 and angle cutoff to 30 (Hint: refer to lab 1).

Paste your commands below:

Report angle:

Lists in Tcl are similar to lists in R, as they are both ordered collections of items that can be of varying data type, including other lists. Some common functions you will use on lists include:

**llength**: returns length

% llength {1 2 3}

3

**lindex**: returns item at given index

lindex {1 2 3} 1

2

**Q9.** Explain why the example above returned 2 and not 1.

To learn more about lists in Tcl, you can [visit](https://www.tutorialspoint.com/tcl-tk/tcl_lists.htm)

Another measurement we might be interested in is the number of hydrogen bonds. We can find hydrogen bonds within selections using **measure hbonds**. This will return 3 lists; read the documentation [here](https://www.ks.uiuc.edu/Research/vmd/current/ug/node138.html) to understand what each list represents.

Since each element in one of the lists corresponds to a hydrogen bond, we can take the length of one of the lists to calculate the number of hydrogen bonds present.

**Q10**. Count the number of hydrogen bonds in the entire protein by taking the length of the first list. Use a cutoff distance of 3Å and angle of 20°.

Paste your commands below:

Report number of hydrogen bonds:

**Q11.** Why might we be interested in counting the number of hydrogen bonds in the protein?

**Part 2 – Creating and Setting up a Water Box for Simulation**

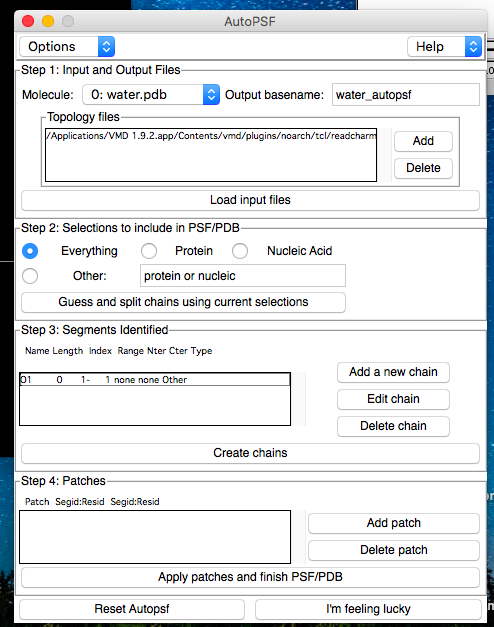
To run molecular dynamics simulations, scientists surround their macromolecules with water and ions to model a real cell as much as possible. Therefore, we first need to make sure that we have a realistic model for water. Download the file water.pdb from Canvas and open this file in VMD. You can load molecules through the VMD Main window, or by using Tcl:

% mol new water.pdb

You should notice that there is only a single atom in this PDB. You can confirm this by opening the PDB file in a text editor. In the default view of “Lines” in VMD, you will not be able to see what this atom is. Change the drawing method to “VDW”, which stands for “Van Der Waals”, a space-filling representation that approximates the atomic radius of each atom.

**Q12.** This water molecule was selected from an actual protein structure solved with X-ray crystallography.

1. What atom of water is visible?
2. Why are the other atoms not visible?

We will simulate the dynamics of water molecules as part of a molecular dynamics experiment using the software NAMD (Not Another Molecular Dynamics simulator, which humorously refers to the field’s propensity throughout the ‘80s and ‘90s for having inordinate amounts of software that accomplish the same thing). After loading the water file, the next step in an MD simulation is to fill in any missing information and supply further parameters about the system in the form of a **Protein Structure File (PSF)**, which contains the system’s charges, bonding patterns, and other structural information. **This process also adds the missing Hydrogen atoms to the water molecule.**

We can use VMD to automatically create PSF files. From the VMD main menu, select Extensions > Modeling > Automatic PSF Builder. This will open a new dialog box. Click **Load Input Files**, **Guess and split chains using current selection**, and finally **Create Chains.** When this is done, wait a short moment while the file is generated. You should now see a completed water molecule in the display window.

**Q13.** Render the completed water molecule using the “Licorice” drawing method.

1. Insert a screenshot here. Be sure to delete the original water molecule (with the single atom) in order to see the complete molecule clearly.
2. Measure the H-O-H bond angle and report it below.
3. How does our angle compare to the TIP3P water model? (Hint: Google the TIP3P angle).

Our next step is to “solvate” our structure to create a water box. Generally, we solvate a molecule (such as a protein) with water molecules; here we are solvating water with water. VMD does this automatically – select Extensions > Modeling > Add Solvation Box. Make sure you have the output from your AutoPSF as the input PSF and PDB files for the completed water molecule, change **Output** to some obvious name (like H2O-solvate), and check **Rotate to Minimize Volume, Use Molecule Dimensions,** and enter **35** in all the xyz-padding boxes. Finally, click **Solvate.**

**Q14.**

1. How many atoms are in this solvated box?
2. This corresponds to how many water molecules?

**Q15.** The above procedure created a new PSF file. What is the difference between this file and the first PSF file you created? (Hint: open the two files using any text editor)

**Q16.** What is the volume of this solvated box in Angstroms3? In cm3?

**Q17.** If the molar mass of water is 18.0 grams, and there are 6.02×1023 molecules per mole, what is the density in g/cm3 of this water box? Show your calculations.

Next, we are going to set up our water box for our simulation by centering the box using Tcl and taking a few measurements. Although there is nothing inherently wrong with an uncentered box, **centering the system box allows for easier analysis in the future.** As previously mentioned, the default directory that Tk Console opens in is likely not the directory your files are in. The current directory is listed in parentheses to the left of the percent sign.

Open the Tk Console and use the **cd** command to navigate to the correct directory.

Make sure your water box is selected as the “top” molecule in VMD. The Top molecule is the molecule with the “T” in the VMD Main screen. You can double-click on the space where the “T” should be to change the top molecule. Having set the water box as the top structure, we are now going to center the water box and measure its dimensions.

The centering must be done for every solvation box for all MD simulations that we will create in this class, including the protein setup for your final project.

Run the following command to select **all** the atoms corresponding to the **top** molecule and assign it to the variable **$sel** (If we wanted, we could replace all with any valid selection, such as resid 1 to 5, in order to select specific residues).

% set sel [atomselect top "all"]

atomselect0

Next, we need to measure the center of the selection that we made and move it to the origin. To do so, we will move every atom’s position by the additive inverse of the current center. These actions can be performed in Tcl; make sure you understand why this code works:

% set coords [measure center $sel]

% $sel moveby [vecinvert $coords]

**Q18.**

1. What does the vecinvert command do?
2. What happened to the water box in the VMD window when you ran the above commands?
3. Repeat the measure center command. What coordinates are printed?
4. Why would you expect this result? Link your observations to your answers in parts (a) and (b).

Finally, we need to save the coordinates of our new centered water box; the TcL command is

% $sel writepdb water\_box\_centered.pdb

Note that this pdb file will save to your current working directory (use pwd). Make sure you are in your intended directory so can find your centered pdb file later.

Now, measure the dimensions of the system, using the command:

% measure minmax $sel

**Q19.**

1. What are the six values that you get? What data structure are the values in?
2. Interpret the values.
3. Thinking back to when we solvated the original water molecule in VMD, why are these values not perfectly Å3?

**Save these numbers plus all of the water files you created, as they will be used in the NAMD configuration files next week.**

It is the first time you have encountered a load of files generated for modeling, but it is just the beginning. Throughout the course, you will have tens, if not hundreds, of new files, generated by either the VMD or MD simulation. It is important to keep the files well sorted so that you will be able to access them when you need to. I recommend that for each model, you create a folder and save all its files inside. Now, you can begin doing this by creating the, for example, model\_water\_box folder and put the following files in:

water.pdb

water\_autopsf.pdb

water\_autopsf.psf

water\_autopsf.log

water\_autopsf\_formatted.pdb

H2O-solvate.pdb

H2O-solvate.psf

H2O-solvate.log

water\_box\_centered.pdb

Note that it is perfectly fine if you want to use different file names, as long as you find your naming reasonable and helpful for future use.