1. Problem Definition:

Finding and displaying annual tuition at a certain college in a table format according to user’s inputted residency status (In-state, out-of-state or Graduate) using the letters I, O, or G respectively. Then calculating and displaying the tuition for the following 5 years at a 3% increase and the total increase from year 0 to year 5.

1. Analysis:

‘Residency’ will hold the string that has been put in by the user, should the user not put in an applicable string it will set to “0” in order to continue a loop until such that the user puts in an appropriate string.

‘TotalIncrease’ holds the running total of the increases each iteration of the calculation loop.

YearStart and YearEnd hold the academic year calculated for by adding one each iteration, starting values of 15 and 16 respectively and one is added to each during each counted iteration, displays in the table as “2016-17, 2017-2018, etc.” when combined with print function

‘Count’ holds the number 5 to start and is reduced each iteration to end the loop when 5 iterations of calculation and print are done.

YearIncrease holds the value of 3% of the Annual Tuition (Antuition \* 0.03)

AnTuition holds the initial Annual tuition amount pre-calculation loop as assigned by the input of the user’s residency status , then holds the new annual tuition as “Antuition = Antuition + yearIncrease” and is added to in each iteration of the calculation loop to display the next annual tuition.

runAgain set to ‘yes’ in order to begin the program’s full loop and will be set at the end of each total iteration by user input to determine if the calculation and table will be run again if set to “yes”

1. Design:

Set run value to enter while loop “RunAgain”

While statement to enter loop will always run at least once, and further times at user request

Initialize variable totalIncrease as 0

Initialize variable YearStart as 15

Initialize variable YearEnd as 16

Initialize variable count as 5

Ask user for type of residency by entering “I”, “O”, or “G” or the lowercase forms and store as residency

Assign AnTuition to appropriate value based on input (10000, 24000, or 40000 for I, O, or G respectively) and initialize variable “AnTuition

New prompt to ask again if input is not one listed above, loop to repeat as needed

Display table header “Undergraduate Tuition for next five years” if residency is “I”, or “O”

Display table header “Graduate tuition for next five years” if residency is “g”

Display separation lines using hyphens (“-“)

Begin a looping calculation and the 3 print functions 5 times

Multiply AnTuition by 0.03, for 3%, store as yearIncrease

Add YearIncrease to Antuition and store as AnTuition for next iteration and current print cycle

Add YearIncrease to totalIncrease and store as totalIncrease for a running total

Add 1 to YearStart and store as YearStart

Add 1 to YearEnd and store as YearEnd

Subtract 1 from count and store as count.

Display in line “20(YearStart)-(YearEnd) AnTuition and YearIncrease with proper spacing to align with headings to form a table (replace variable name in parenthesis with stored variable)

When the count equals 0 loop will be exited

Check if inputted value “residency’ is ‘G’, if it is, display total with certain amount of spacing, if not, use other spacing

Ask user if they would like to check another tuition projection, assign to runAgain.

Loop request until accepted form of “yes” or ‘No” is entered, assign to runAgain

If yes, repeat from first while statement.

If no, end program.

1. Implementation

The program was designed and tested within the IDLE environment on an AMD CPU based windows 7 PC. Designed to only enter/exit loops when certain conditions of input are satisfied. As such if an input that the program wouldn’t be able to enter is put in, the program will simply ask over and over until it’s satisfied with the answer for both inputs. Should any string that is not ‘I’, ‘O’,‘G’,‘I’, ‘o’, or ‘g’ be entered it will be rejected and the user will be asked for a proper input, this will happen regardless of type of string inputted as calculation is handled by assigning the value based on the input, calculations are unaffected by any messing around with attempted inputs. The same is true of the ‘Yes’ or “No’ prompt at the end, all variants of case sensitivity will be accepted as well as ‘y’ or ‘n’ as they are ubiquitous I decided to include as potential responses. Tested with the intended inputs as well as multiple strings varying from 1-10 characters consisting of all types of symbols found on a keyboard as well as some python functions to attempt. Should no proper “no” input be ever entered in the second request, the program will never properly end and exit though any prior requests will be visible. Also serves to keep open window when used with basic python shell only closing when a “no” response is entered.