*Turn on the Navigation Pane in Word when viewing this document:* **View tab > enable Navigation pane**

# Program Specification

* *Program title:* *Property Tax Estimator*
* *File name:* MIS3301-Ch5-HW-PropertyTaxes.py
* *Due Date:* see Canvas for due date
* *Main objective:* *Victorino County Tax Assessor* is the property tax office for Victorino Countywhich has a population of 100,000. They are requesting that you write a program that allows a county resident to enter their residential property information and estimate their property taxes. The system should be flexible and allow the user to enter any valid input values. The system should prompt the user for the property address, size of the building (i.e. house), the front & depth of their land, and a tax exemption code. Finally, a *Property Tax Estimate Report* should be generated.

# Read before starting

* As before:
* The expectation is that students have read the *entire* chapter and attended *all* lectures.
* Reminder: This is an *individual* assignment. Students are expected to complete this assignment independently and without the assistance of others. If you need help, contact your instructor via email or visit during office hours.
  + When emailing your instructor, please provide as much information as possible:
    1. State what the issue is and provide line numbers.
    2. Attach your \*.py file *zipped* as outlook does not permit attaching these files.
    3. Also, copy & paste the snippet of code where the issue is occurring within the email. If I am not at a computer when I see your email, I may be able to respond by seeing the snippet of code.
* I recommend that you read the entire assignment before beginning.
* Periodically review the ***Assignment Tips & Updates*** page in our module looking for tips as well as any ***required*** updates to this assignment.
* Refer to the sample *Execution Screenshot* for additional details.
* This code must be free of syntax errors, or it will not be graded. Thus, comment out any code that crashes, or for a better grade, improve the code and submit it one day late with a 10% deduction.
* Unique to this assignment:
  + This purpose of this assignment is to demonstrate your knowledge of functions. While the program could be written without functions, **no credit will be given if functions are not defined and called.**
  + **NO global variables** - do not use a concept discouraged in class called *global variables*.  
    **YES global constants** - do use *global constants*.
  + To avoid confusion, I recommend that you include the words *building (bldg)* or *land* in the name of variable names or functions. For example: Variables: **bldg**\_sqft**, land**\_sqft;   
    Functions: calc\_**bldg**\_value(), calc\_**land**\_value()

# Execution Screenshot

|  |
| --- |
| 1. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 2. Victorino County Tax Assessor 3. \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 4. Welcome to the Residential Property Tax Estimation Tool! 5. Enter property address: **123 Rolling Hills Ln** 6. Enter building size (sqft): **2450** 7. Enter land effective front (ft): **118.75** 8. Enter land effective depth (ft): **130.66** 9. Enter exemption code - (H)omestead, (O)ver 65, or (N)one: **H** 10. ------------------------------------------------------------ 11. Property Tax Estimate Report 12. ------------------------------------------------------------ 13. >>> Report for property located at: **123 Rolling Hills Ln** 14. Building: **2,450** sqft 15. Land Info 16. Front: **118.75** Depth: **130.66** 17. Sqft: **15,515.88** Acres: **0.3562** 18. Building market value: **318,500.00** 19. Land market value: **35,619.55** 20. Taxes w/Current Exemptions: **6,020.03** (**Homestead**) 21. Taxes w/o Exemptions: **7,082.39** 22. ------------------------------------------------------------ 23. Estimate Confirmation No: **402710** |

* Blue, bold text is data entry.
* Dark red, bold text is a calculated or determined value.

# Initial Steps

## Getting Started

* Open IDLE > open the *Editor Window* (File >NEW)
* Click File > Save As…
* Locate your Chapter 5 folder for this course
* Save your file as: **MIS3301-Ch5-HW-PropertyTaxes.py**

## Add Comment Lines

* Copy the comments lines below to your code.

|  |
| --- |
| #add required assignment header comment lines here  #Import Modules  #Global Constants  INDENT = ' '  **#MAIN--------------------------------------------------------------------------------------**  #Define the main() function here  #Display company & application header  #INPUT-----------------------------------------------------  #Get user input  #Test data  #PROCESS---------------------------------------------------  #OUTPUT---------------------------------------------------  #Display report title  #Display report results  #Display report footer  **#FUNCTIONS--------------------------------------------------------------------------------------**  #FN1: Company banner    #FN2: Application title  #FN3: Exemption name  #FN4: Building value  #FN5: Land sqft  #FN6: Land acres  #FN7: Land value  #FN8: Property taxes  #FN9: Confirmation number  **#Call main() here** |

## Constants

* Make sure to use the provided INDENT constant anywhere there is indentation in the output (as shown in the execution screenshot).
* Also, identify and properly name additional constants any time you have a value that will not change.

## Begin the main() function

* First, replace the comment line "#Define the main() function here" with an appropriately written **function definition** for the main() function.
* Next, replace the last comment line "#Call main() here" with an appropriately written **function call** for the main() function. Note: if your forget to do this, when you execute your code, **nothing will occur** because functions must be called to run.

## Code the Company & Application Header

* Write the code to display lines 1-7 in the execution screenshot directly in the main() function under the comment "Display company & application header". Run & test it.

## Code the User Input

* Write the code to prompt the user for all required input in the main() function under the comment "Get user input". Run & test it.

## Code the Test Data

* Speed up your testing! Comment the user input code in the previous step so you are not prompted each time to enter the data; instead test your program using the following test data. Copy the code below to the main() function under the comment "Test data". Replace all question marks with the same variable names that you used in your "Get user input" code.

|  |
| --- |
| #Test Data  #??? = '123 Rolling Hills Ln'  #??? = 2450  #??? = 118.75  #??? = 130.66  #??? = 'H' |

* **Important:** during final testing (and before submitting your assignment), make sure to uncomment the "Get user input" lines (and comment out your hardcoded sample data) in order to test various sample values. When your assignment is graded, we will start by manually entering our own test data.

# Review on Functions

**A function is a block of code that has one specific purpose**; thus, its code should not do anything else but what is stated in the function name. Creating functions allow us to organize our program better and in some cases allows use to create reusable code.

|  |  |
| --- | --- |
| Text  Description automatically generated with medium confidence | **FN0:** Calculate and return the total pay. Total pay is the product of hours and the pay rate. |

## How do I write a **Function Definition**?

When defining each **function definition**, you will need to know (or decide) 4 pieces of information:

* *Function name**-* a well-written function name tells everyone the purpose of the function (i.e. what it is responsible for producing). Always begin a function name with a verb such as: calc, display, generate, etc. The function name appears after the keyword *def*.
* *Function description**–* read the description provided for each function as they explain important details such as calculations required.
* ***Parameters*** – some functions require values to accomplish its purpose. Parameters are shown visually in the image above as *input arrows*. Functions can have 0 or many parameters. These values are received and then used within the function. Parameters are coded within parentheses.
* ***Return values*** – functions sometimes return value(s) to the calling program and are shown here as *output arrows*. Functions that return a value are called *value-returning functions*, and those that do not are called *void functions* which typically display output directly to the output window (i.e. Shell window). In Python, functions can return many values (but typically, only returns one value). Return values must be listed after the keyword *return*.

Text

Description automatically generated

## How do I write a **Function Call**?

* A **function call** is a line of code that invokes/executes the function. For our assignment, we will *call* (or invoke) our functions only within the main() function.
* To call a function, you are required to send (i.e. pass) the function any values that are defined in the function header. We refer to these values as **arguments**. Thus, we are "passing arguments".
* The *function call* should also receive any values that are returned from the function by adding a variable name to the left of the function call.

Text

Description automatically generated

# Coding the Functions

Next, you will code functions to produce most of the values used later in the report. There are 2 ways to work through the steps below:

* ***Approach A:*** code & test **one** function at a time – i.e. code the function definition, then code the function call in main(), and test it (before coding the next function).
* ***Approach B:*** code **all** function definitions first. Finally, go to main(), and write and test each funcall & test one function at a time. Either approach is fine!

## Code FN1: Company Banner

We have decided that since the company banner will be used in all future applications, it will be best to move that code to a function so that it can be reused by other programs.

* **Function Definition:** Under the #FN1 comment line, code the function definition header  
  to match the screenshot below 🡪 **def display\_company\_banner():** – however, do not change the comment line as we want to keep the function number - "#FN1: Company banner". Move the code that you had previously written to display the company banner (i.e. lines 1-5 on the execution screenshot) to this function. This function will not include the application title. As described above in the section titled "Review on Functions", you should recognize that this is a *void* *function* – i.e. it does not return a value. This function is a very basic function as it also has no parameters.

|  |  |
| --- | --- |
| Graphical user interface  Description automatically generated with low confidence | **FN1:** Display the company banner as shown in lines 1-5 in the execution screenshot. This function will include print() functions. |

* **Function Call:** Remember that functions do not execute until they are called. Write the code to call this function within the main() function at the top, under the comment “#Display company & application header”. Run your program and verify that your banner still works.

## Code FN2: Application Title

We have also decided to create a reusable function that displays *any* application title.

* **Function Definition:** Name this function as shown in the image below and code it to achieve its purpose as stated in the description. Move the code that you had previously written to display the application title (i.e. lines 6-7 on the execution screenshot) to this function. Notice that this function has one parameter (input) to accept any application title. Make sure to use this parameter so that this function is dynamic – i.e. it display any title sent to it.

|  |  |
| --- | --- |
|  | **FN2:** Using an application title that is received, display the application title as shown in lines 6-7 in the execution screenshot. This function will include print() functions. |

* **Function Call:** Write the code to call this function within the main() function at the top, under the comment “#Display company & application header”. Make sure to pass it the application title – you may hardcode the title in the argument (or use a variable). Test your code.

## Code FN3: Exemption Name

* **Function Definition:** Name this function as shown in the image below and code it to achieve its purpose as stated in the description. Notice that this function has one parameter (input) and produces one return value (output). As described above in the section titled "Review on Functions", you should recognize that this is a *value returning* *function* – i.e. it returns a value to the calling program.

|  |  |
| --- | --- |
|  | **FN3:** Using the exemption *code* that is received (H, O, N), return the appropriate (and longer) exemption *name*. This function must NOT have a print() function. |

* **Function Call:** Next, call this function within main() under the comment line “Process” and test it. Recall, that since this function returns a value, when this function is called, the function call must handle/accept the returned value by storing it in a variable – just like we do when we write an input() function. See the "Review on Functions" section above on how to do this.
  + **Important note:** when you test any function that returns a value, you will not see anything displayed to the output window. To see if the returned value is correct, add a temporary line after your function call to display the returned value (i.e. exemption name). Remove this temporary line once tested.

## Code FN4: Building Value

* **Function Definition:** Name this function as shown in the image below and code it to achieve its purpose as stated in the description.

|  |  |
| --- | --- |
|  | **FN4:** Calculate and return the market value of a building/home. Assume that the average home value in this county is $130.00 per square foot. This function must NOT have a print() function. |

* **Function Call:** Next, call this function within main() under the comment line “Process” & test it.

## Code FN5: Land Sqft

* **Function Definition:** Name this function as shown in the image below and code it to achieve its purpose as stated in the description.

|  |  |
| --- | --- |
| Diagram  Description automatically generated | **FN5:** Calculate and return the total square footage of a piece of land. The square footage is the front of the land (in feet) multiplied by the depth (in feet).  This function must NOT have a print() function. |

* **Function Call:** Next, call this function within main() under the comment line “Process” & test it.

## Code FN6: Land Acres

* **Function Definition:** Name this function as shown in the image below and code it to achieve its purpose as stated in the description.

|  |  |
| --- | --- |
| Diagram  Description automatically generated | **FN6:** Calculate and return the total acres for a piece of land. Note: one acre is equal to 43,560 square feet.  This function must NOT have a print() function. |

* **Function Call:** Next, call this function within main() under the comment line “Process” & test it.

## Code FN7: Land Value

* **Function Definition:** Name this function as shown in the image below and code it to achieve its purpose as stated in the description.

|  |  |
| --- | --- |
|  | **FN7:** Calculate and return the market value of a piece of land. Assume the market land value is $100,000.00 per acre. This function must NOT have a print() function. |

* **Function Call:** Next, call this function within main() under the comment line “Process” & test it.

## Code FN8: Property Taxes

* **Function Definition:** Name this function as shown in the image below and code it to achieve its purpose as stated in the description. Notice that this function has multiple parameters (inputs) and produces ***multiple*** return values (outputs). The latter is unique to Python as most programming languages do not allow functions to return more than one value. See class notes on how to write a *function with multiple return values*.

|  |  |
| --- | --- |
| Graphical user interface, text, application, chat or text message  Description automatically generated | **FN8:** Calculate and return both values for property taxes owed: 1) property taxes (i.e. full taxed amount), and 2) property taxes with exemptions (i.e. taxed amount after the discount). See the additional details below.  This function must NOT have a print() function. |
| * *Property Taxes* - the tax rate is a flat 2% per the total value of the property (i.e. the building’s value and the land’s value). * *Property Taxes with Exemptions* - If the resident has an exemption, the appropriate exemption *percentage* is deducted from this amount to yield the "property taxes with exemptions". There are two exemptions in this county: “Homestead” which requires you to be living in the home as your primary *residence*, and “Over 65” which is for any resident over 65 years old. Typically, counties allow for more than one exemption; however, assume that this county only allows one exemption. A *Homestead* exemption reduces the property taxes owed by 15% and an *Over 65* exemption reduces the taxes by 20%. * If there is no exemption, the values for property taxes with and without exemptions will be the same. | |

* **Function Call:** Next, call this function within main() under the comment line “Process” & test it. Make sure to also review class notes on how to *handle the 2 returned values properly*.

## Code FN9: Confirmation Number

* Assume that the county plans in the future to store the generated *Property Tax Estimate Report* in a database table. The report must contain a unique estimate confirmation number in the event they ever need to pull the report from the database. Right now, we will temporarily use a random number generator as taught in class and shown in the textbook to generate this number; in the future, it will be easy to update the code to enforce uniqueness by simply changing the code in this sigle function.
* **Function Definition:** Name this function as shown in the image below and code it to achieve its purpose as stated in the description.

|  |  |
| --- | --- |
| Graphical user interface, text, application, chat or text message  Description automatically generated | **FN9:** Calculates and returns a *randomly* generated 6-digit estimate confirmation number. *Tip:* you will need to *import* the **random** module to use any of the random functions. Add the import statement at the top of your code under the "Import Modules" comment line.  This function must NOT have a print() function. |

* **Function Call:** Next, call this function within main() under the comment line “Process” & test it.

# Coding the Property Tax Estimate Report

Finally, write the code to display the *Property Tax Estimate Report*. This code will be written in main(). Review the execution screenshot before starting. Notice that the values in lines 18-23 came directly from the user and the values in lines 24-31 were calculated by the functions. Thus, you are simply displaying the values stored in these variables.

## Code the Estimate Report Title

* In the OUTPUT section of main(), under “#Display report title”, write the code to generate lines 13-19 in the execution screenshot. Run & test it.

## Code the Estimate Report

* In the OUTPUT section of main(), under “#Display report results”, write the code to generate lines 20-29 in the execution screenshot. You should already have the data in variables to produce this. Run & test it.

## Code the Report Footer

* In the OUTPUT section of main(), under “#Display report footer”, write the code to generate lines 30-32 in the execution screenshot. You should already have the data in a variable to produce this. Run & test it.

# Final Steps

## Finalize your code & testing

* Review the execution screenshot to finalize anything that is still missing. You should not need to write any additional functions. Make sure to handle all formatting as shown.
* Make sure to uncomment the "Get user input" lines that you had previously commented out; and comment the code for "Test Data". Test all 3 scenarios – Homestead, Over 65 age, and no exemption.

# Submitting the Assignment

* First, review the ***Assignment Tips & Updates*** page to see if there are any additional requirements to this assignment – denoted with the tag: **[required].**
* To submit the assignment, go to Canvas and open the assignment link.
* Upload your \*.py file**.** You may upload more than once as long as it is before the due date & time. Only the final submission will be graded. Assignments will be accepted one day late albeit with a 10% deduction.