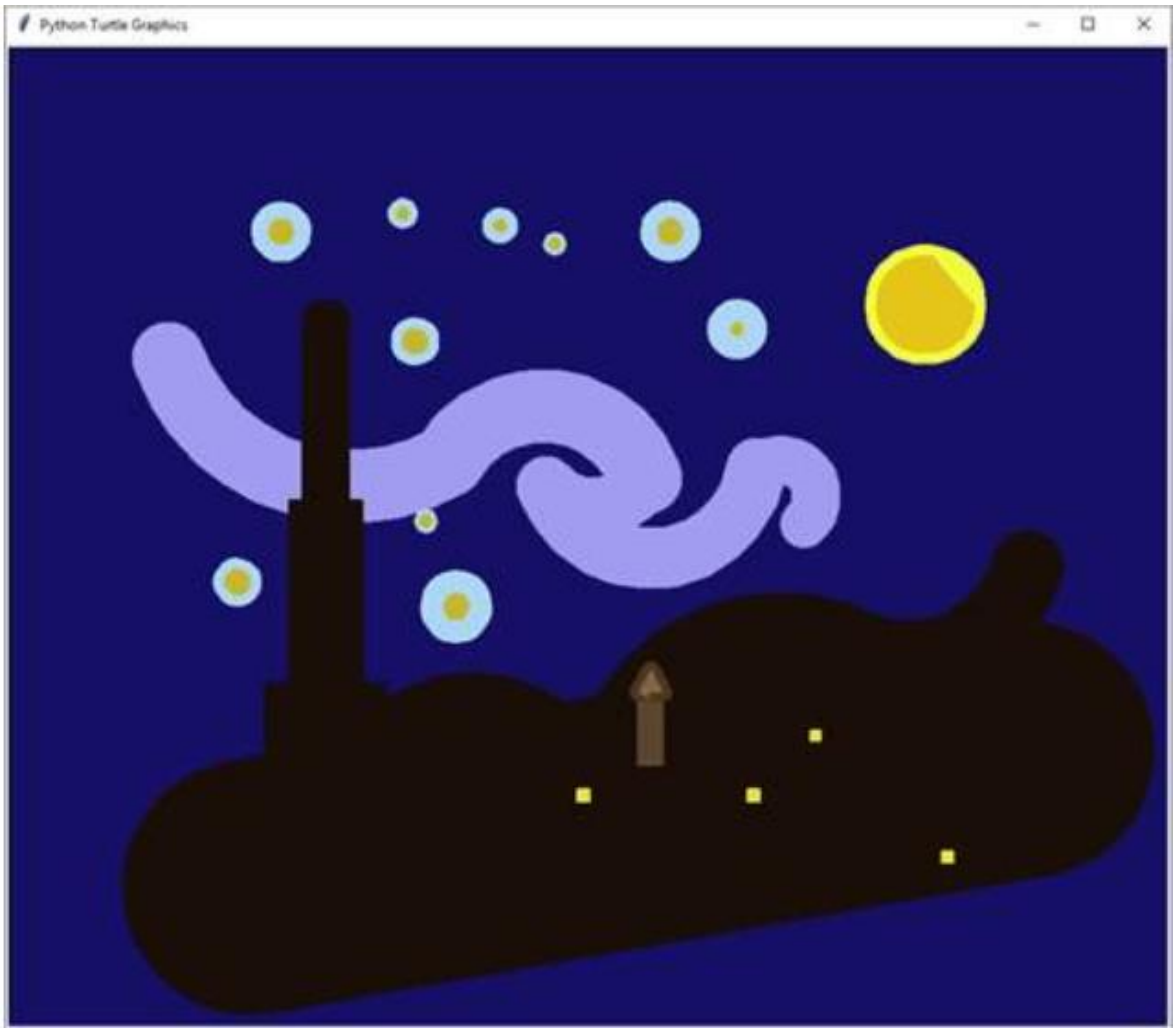
Assignment 3 – Mon *Chef-d'Oeuvre*!

Title: Assignment 3 - Mon *Chef-d'Oeuvre*!

Step 1 - Problem Statement:

Write a Python program, called **`MonChefDoeuvre1241.py`**, to create a drawing inspired from the work of a known painter (artist) using the Python Turtle module and its built-in functions. You must create and use at least one turtle from this module to create your drawing.

To illustrate what this assignment is asking you to do, here is the *chef d'oeuvre* created by a CMPT 120 student from a previous semester:



inspired by Van Gogh's famous painting **Starry Night**:

Awesome, isn't it! :)

Your program, for Assignment 3, must also satisfy the following requirements.

Requirements:

1. You must create your own Python 3 program using IDLE. You cannot use AI tools to create your program nor can you use someone else's program. For more details regarding Academic Honesty (or lack of) and what is permissible and what is not, please, read **A word about Academic Honesty** under **Lecture 1** on our course web site.
2. Your program must start when the menu option **Run** is selected or **F5** is pressed once we have opened your program file in IDLE.
3. Your program must not require input from the user in order to completely execute. This is to say that the user must not be prompted for any input data once your program has started executing, it must execute all on its own until it reaches its end.

4. The last statement in your program must call the turtle function (method) `exitonclick( )` as presented in our lectures.
5. Your drawing must completely fit within a Turtle canvas of default size, i.e., you cannot resize the canvas. This signifies that the viewer (the TA) must not have to make the Turtle canvas larger or scroll in order to view your drawing in its entirety. To help you visualize a Python canvas too small, download and execute the program **Turtle_Canvas_Too_Small.py**.
6. You cannot make use of Python modules that need to be downloaded on a computer before they can be used. You can only use Python modules that are already available on Python 3 IDLE (available by simply using the Python statement `import` in your Python program).
7. Your drawing must be composed of at least thirty (30) shapes. To help you determine what a "shape" is in a Turtle drawing, download and execute the program **Turtle_Shapes.py**. The drawing created by this program consists of nine (9) shapes:
    - 2 rectangles
    - 1 square
    - 3 circles
    - 2 semi circles (note that 2 semi circles drawn together count as 1 shape, but here, the 2 semi circles are not drawn together so they count as 2 shapes)
    - 1 quarter of a circle.
8. Some of the shapes your program draws must be filled in. For example, the rightmost rectangle in the drawing produced by the program **`Turtle_Shapes.py`** (the one with the "yellow" contour) is filled in with the colour called "hot pink".
9. Your drawing must include at least ten (10) colours. Looking at the drawing the program **`Turtle_Shapes.py`** generates, you can count eight (8) colours (nine (9) if you include "white").
10. Your program must include at least **five (5)** `for` loops. Looking at the program **`Turtle_Shapes.py`**, you can count two (2) `for` loops.
11. Your program must include at least four (4) **functions**:
    - Two (2) of these functions must be **void** functions, i.e., functions that do not return any values.
        - Our e-textbook calls these **void** functions **non-fruitful** and it says that they return the value **`None`**.
        - Note that these two (2) functions may or may not require parameters.
    - Three (3) of these functions must require parameters (the number of parameters is up to you).
        - Note that these three (3) functions may or may not be **void**.
    - Looking at the program **`Turtle_Shapes.py`**, we can count three (3) **void** functions that require parameters.

- You must apply the **Generalization Principle** when creating your functions as much as you can. This is to say that you must design your functions in a general fashion (perhaps using parameters and/or returned value) such that they can solve several similar problems, i.e., be called several times with different parameter values. Therefore, each of your (at least) four (4) **functions** must be called at least twice, either from the **Main** part of your program, and/or from your other functions.
- All your functions must terminate with a `return` statement, whether they are **void** functions or not.

12. Your program must not contain repeated code: code fragments that look very similar to each other and are repeated several times in your program.
13. You can use `while` loops in your program. However, your `while` loops must make use of a Boolean condition (i.e., Boolean expression) which must evaluate to `True` or **False** depending on the value assigned to the variable(s) contained in this Boolean expression. You cannot use `break` or `exit( )` function as part of the body of your `while` loop.
    - Also, you must use `while` loops only in situations when you do not know how many times your code needs to iterate.
14. You can use `for` loops in your program.
    - You must use `for` loops only in situations when you do know how many times your code needs to iterate.
15. You must include **a header comment block** at the top of your program containing the four (4) sections we talked about in our lectures:
    - filename
    - description
    - author
    - date
16. **For this assignment, you also need to add the following to your header comment block:**
    - **the name of the artist**
    - **the name of her/his *chef d'oeuvre* that inspired your drawing**
    - **whether we would like to have your *chef d'oeuvre* show cased (displayed anonymously) in class**
17. Your program must follow these **Good Programming Style (GPS) rules**:
    - **Functions** and **variables** must always be named in a descriptive fashion (describing their purpose).
    - All **import** statements must be placed at the top of the file, immediately after the header comment block, before the functions.
    - Each line of your code must not exceed 80 characters in length.

Step 2 - Design:

Before you start coding, make sure you design your algorithm using English (natural language) or using pseudocode. You must include your algorithm as comments in your program.

Step 3 – Implementation:

Once you have included your algorithm as comments in your empty program, translate each of the steps of your algorithm (each comment) into Python code. You must leave the comments in the code.

Incremental development: Implement and test your Python program in an incremental fashion, i.e., implement a feature at a time (perhaps a function), then test it. Move on to the implementation of the next feature only once the previous one has been successfully tested.

This **incremental software development strategy** is in contrast with the "Big Bang" approach where the whole program is written in one go and only tested at the end. This approach often leads to lengthy debugging sessions and a lot of frustration. ☹

Step 4 - Testing:

Incremental testing: Test as you go! This means that as you are implementing a feature at a time (perhaps a function), make sure your program executes as expected before to proceed to the next feature or function.

Art Gallery:

After the due date, we shall have **Art Gallery moments** at the beginning of our lectures in which we shall present *chef d'oeuvres*. So, if you are proud of what you created in this assignment and wish to have your *chef d'oeuvre* presented in class, please, let the instructor know by saying so on the very last line of your header comment block (see Requirements 16 above).

Submission:

- Submit your program on CourSys (https://coursys.sfu.ca/2024sp-cmpt-120-d3/). Click on the course activity called **Assignment 3**, then click on the option **Make Submission** on the left and finally, follow the instruction to browse for your program file.
- Note that you can submit your program as often as you wish. As long as your submissions are done before or on the due date and time, your assignment will be marked. CourSys will not stop you from submitting your program late, i.e., after the due date and time, but if your program is late, it will receive 0 marks.

How your Assignment 3 will be marked:

- When the TA marks Assignment 3, they will be looking at
    - whether your program solves the problem (10 marks),

    - whether your program satisfies **all the requirements** stated in this assignment (10 marks).
- The rubric for Assignment 3 is based on the above. Make sure your program satisfies the above before submitting your Assignment 3.

Since the drawing you create in this assignment is completely up to you (and your *muse*), no two drawings are expected to be similar!

Enjoy the creative process!

If you have any questions, drop by our office hours or post them on our Discussion Forum. Suggestion: Make sure you deal with your questions as you go. Do not wait until exam time.

Finally, there are no extension granted unless for medical reason once the Official Medical Certificate has been completed and submitted to the instructor.