

CPSC 231: Mini-Assignment 5

Due Friday June 23 at 5 PM

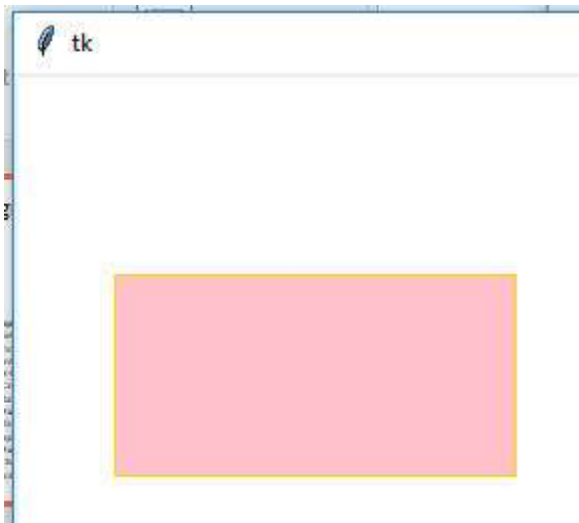
New Concepts to be applied for the assignment

- **Part I:**
 - Writing a program that reacts to button events using the `tkinter` module
 - Drawing basic shapes and setting graphical properties using the `tkinter` module
- **Part II:**
 - Defining classes, declaring references, creating objects and calling methods

Part I Description:

Modify the existing code in the program `'button_events.py'` program to draw the graphic of a rectangle that appears in a new Window when the button is pressed. The window is already created and appears when the button is pressed. You need to add a rectangle to the popup window with the following properties:

- Coordinates of top left hand corner (50,100) worth **0.4 GPA**
- Coordinates of bottom right hand corner (250,200) worth **0.4 GPA**
- The rectangle is draw in 'pretty': filled in 'pink', outline in 'gold' worth **0.6 GPA**



Part II Description:

Your program must consist of two classes (as described below). Aside from the `'start()'` function you are not to define any other functions for this assignment. If you do implement other functions methods then you will receive few, if any, marks for features listed below. Also to make it easier to mark ensure that your method carries out the prescribed tasks exactly as specified (so the the person knows precisely what results should come out at what times).

Details

- Defines a class `'Adventurer'` which **MUST** be in it's own module file (otherwise no marks for the features below)

- Attributes: `name` (String), and `health` (int) (**0.4 GPA**)
- Methods:
 - A no argument constructor that sets the `name` and `health` attributes to obvious default starting values (e.g., "nameless" for the name and -1 for the health (**0.4 GPA**))
 - A '`gainLevel()`' method that displays a message onscreen ("Congratulations!") and increases health by 5. (**0.4 GPA**)
- Defines a '`start()`' function which MUST be module (otherwise no marks for the features below) which will contain the following instructions
 - Declares a reference and calls the default constructor (**0.2 GPA**)
 - Displays the health and name of the adventurer, the default values should appear (**0.4 GPA**)
 - Sets the name of the adventurer to 'Balin' (**0.2 GPA**)
 - Calls the '`gainLevel()`' method of the adventurer (**0.2 GPA**)
 - Displays the health and name of the adventurer, the new values should appear 'Balin' and a non-default value for the health (increased by 5 over the default) (**0.4 mark**)

D2L configuration:

- Multiple submissions are possible for each assignment: You can and should submit many times before the due date. *For this assignment D2L will keep older versions of your submissions.*
- **Important!** Multiple files can be submitted for each assignment. I am allowing you to submit multiple files for each assignment so **don't archive/compress multiple files using a utility such as zip**. However, this means that **TAs will only mark the latest versions** of each file submitted via D2L. Even if the version of a document that you want marked has been uploaded into D2L if it isn't the latest version then you will only get marks for the latest version. (It's unfair to have the TAs check versions or to remark assignments because marking is enough work as-is).

Marking

- Assignments will be marked by your tutorial instructor (the "Teaching Assistant" or "TA"). When you have questions about marking this is the first person that you should be directing your questions towards. If you still have question after you have talked to your TA, then you can talk to your course (lecture) instructor.
- Here is the [[marking checklist](#)] that specifies the grade breakdown for specific parts of the assignment. Besides seeing your grade point in D2L you can also see the detailed feedback that your TA will enter for each student. You can access the grading sheet in D2L under **Assessments -> Dropbox** and then clicking on the appropriate assignment link. If you still cannot find the grading sheet then here is a [[help link](#)] To speed up marking: assignments that are awarded full credit will just show the numeric show in the D2L Dropbox (no feedback is needed), assignments awarded partial credit will include the marking sheet.

Points to keep in mind:

1. **Due time:** All assignments are due at 5 PM on the [due dates](#) listed on the course web page. Late assignments or components of assignments will not be accepted for marking without approval for an extension beforehand. Alternate submission mechanisms (non exhaustive list of examples: email, uploads to cloud-based systems such as Google drive, time-stamps, TA memories) cannot be used as alternatives if

you have forgotten to submit work or otherwise have not properly submitted into D2L. **Only files submitted into D2L by the due date is what will be marked**, everything else will be awarded no credit.

2. **Extensions** may be granted for reasonable cases by the course instructor with the receipt of the appropriate documentation (e.g., a doctor's note). Typical examples of reasonable cases for an extension include: illness or a death in the family. Example cases where extensions will not be granted include situations that are typical of student life: having multiple due dates, work commitments etc. Tutorial instructors (TA's) will not be able to provide extension on their own and must receive permission from the course instructor first.
3. **Method of submission:** You are to submit your assignment using D2L [[help link](#)]. Make sure that you [[check the contents of your submitted files](#)] (e.g., is the file okay or was it corrupted, is it the correct version etc.). It's your responsibility to do this! (Make sure that you submit your assignment with enough time before it comes due for you to do a check).
4. **Identifying information:** All assignments should include contact information (full name, student ID number and tutorial section) at the very top of your program in the class where the 'start()' function resides (or the starting execution point for assignments that don't require functions to be defined). (Note other documentation is also required for most full assignments).
5. **Collaboration:** Assignments must reflect individual work; group work is not allowed in this class nor can you copy the work of others. For more detailed information as to what constitutes academic misconduct (i.e., cheating) for this course please read the following [[link](#)].
6. **Execution:** programs must run on the computer science network running Python 3.x. If you write your code in the lab and work remotely using a remote login program such as Putty or SSH. If you choose to install Python on your own computer then it is your responsibility to ensure that your program will run properly here. It's not recommended that you use an IDE for writing your programs but if you use one then make sure that you submit your program in the form of text ".py" file or files.
7. **Use of pre-created Python libraries:** unless otherwise told you are to write the code yourself and not use any pre-created functions. For this assignment the usual acceptable functions include: `print()`, `input()` and the 'conversion' functions such as `int()`, `float()`, `str()`. Look at the particular assignment description for a list of other classes that you are allowed to use and still get credit in an assignment submission.