Homework 15– Project Homework

Introduction

In this homework, we are going to do something a little different than the rest of the homework assignments that you have done so far. Many times in Computer Science, you will receive a strict set of criteria that tell you exactly how you should solve the problem– just like we have done all semester. In other cases, (the more fun cases) you will only be given general guidelines to follow, and it is up to you to use your creativity and (newly developed) problem solving skills to get the job done! In homework 14, we want to tap into your creativity and allow you to pick your own project that you can make your own.

In addition to making the project your own, we also want to introduce you to doing your own research while trying to solve computer science problems. Outside of the classroom, you will not always be told which functions to use and how to use them, and that is exactly what this project is all about. We are not going to tell you how to solve the problem – that is up to you. **We may give a few hints and guidelines, but the implementation is completely up to you. There are no banned functions.**

So get out there, be creative, and have fun with this project!

Grading

Each project will be graded out of 100 points. There is a total of 100 extra credit points available for each project (yes, 100 points). The extra credit is more challenging and will push your problem solving skills! **Note that this project will not be autograded. It will be hand-graded by the TAs, so worry less about exact formatting and worry more about impressing the TAs with your awesome projects.**

What to turn in

Each project below lists the specific things that you must turn in if you choose that project. **However, each project must also include a text document with answers to the following questions:**

1. What project did you choose? Why did you choose that project? (1-2 sentences)
2. How did you solve the problem? Why did you solve it that way?
3. What did you learn?

Your responses to these questions don't have to be very long. Focus on answering the question succinctly and accurately. **Note that the text document is not graded directly, but your project will not be graded if you do not submit the text document**. The text document should be named <gt_username>_analysis.txt where "<gt_username>" is replaced with your GT username, not your GTID.

Extra Credit Notes

As stated above, each project includes extra credit options. However, we are very interested in how creative you can be with these projects. If you think of your own improvement or cool feature to add and want it to be extra credit, contact your TA to get it approved! Let them know about what you are improving, and how many points you think that new extra credit option should be worth.

Table of Contents

There are a total of 7 projects to choose from. Check out the links below to find the description of each. **You only have to choose one!**

For some of these projects, there are extra resources that you need that can be downloaded at cs1371.gatech.edu/projectHW/project.zip

Project Option #1: Write your own autograder

You are nearing the end of a wonderful Fall semester at Georgia Tech, when a terrible catastrophe happens – Georgia Tech servers catch fire and are irreparably destroyed. All Computer Science classes, including your beloved CS 1371, are in chaos. Since the TAs can't possibly grade all the remaining homework submissions by hand, they tell the class that everyone will get a 0, *unless* someone can come up with a quick fix. You realize that, given all your MATLAB knowledge, you yourself can create a rudimentary autograder!

Create a function, `autograder.m`, that takes in a rubric structure, and is placed in the same directory as student code, and autogrades their homework. You can assume that you're in a folder that contains the following files and directories:

- A folder with solution files (.p)
- A folder for each student
    - In each folder, the homework files that the student turned in (ie, `rectangleMath.m`)

Your job is to generate feedback for each student. The student folders are named based on the student's name; for example, if the student's name is Alexander Rao, their folder is "Alexander Rao". You can assume that each student has a folder and their code won't error. You can assume that all students have a folder.

The rubric is a structure that contains information about each problem:

Field:	Value:
`name`	The name of the problem.
`testCases`	The test cases to run for that problem
`grades`	The point value assigned for this problem
`bannedFunctions`	The banned functions. This is for extra credit only!

`name` is a string that represents the name of the problem; for example, if the problem is called `getFeedback`, then the name field of rubric will have `getFeedback` for that problem.

`testCases` is a cell array where each value is the complete set of inputs for the given test case. For example, if the first test case has three inputs (1, 2, 3), then `testCases` would have `{{1, 2, 3}}` as its value. The number of elements in `testCases` is the number of test cases, and each test case is guaranteed to be a cell array; that is, `testCases{1}` will be the complete set of inputs for the first test case.

`grades` is a vector of numbers, the same size as `testCases`. Each value in `grades` is the number of points to assign for that specific test case; for example, if the first test case was worth 15 points, and the second test case was worth 5 points, `grades` would be `[15 5]`. You are guaranteed that all points will always add up to 100 percent.

`bannedFunctions` is a cell array of strings that represent functions the student should not be able to use. For example, if you shouldn't be allowed to use the `numel()` function, then `bannedFunctions` would contain `{'numel'}`.

`rubric` is a structure array; below is an example of what `rubric` might look like:

```
rubric(1):
      name: drinkWater
      testCases: {{1, 2, 3}, {55, 6, -7}, {45, 44, 1}}
      grades: [10 10 20]
      bannedFunctions: {'imread', 'why'}
rubric(2):
      name: theRest
      testCases: {{[1 2 3]}, {'helloWorld1'}, {}}
      grades: [10 15 35]
      bannedFunctions: {'size', 'numel'}
```

Your autograder should create some kind of feedback file (whether it be text or HTML or anything else you can come up with) that contains the following information:

- Student name
- List of each problem and test case
- Whether they got the test case right or wrong
- The points the problem was worth
- Some kind of summary of their grade for each problem
- Some kind of summary of their overall grade

Each student should have a feedback file.

Additionally, we also need to upload these grades to TSquare, so you should generate an Excel file that summarizes each student and their grade for the homework. The exact structure of this file is not important. Think about what would be most useful given the task!

Notes:

- You are guaranteed that each student's code will work for the given test case, and that the same goes for the solution code.
- The function name will always be exactly the same as the entry in the `rubric`.
- There will never be any external files used in the code
- The zip file attached for this problem gives an example of what the folder directory might look like

Point Breakdown:

- 40 points: Your code successfully loops through all the student folders
- 30 points: You correctly write the feedback or each student
- 15 points: You correctly write the gradebook file
- 15 points: The whole thing works correctly

There is a chance for extra credit:

- 25 points: You successfully detect and catch all extra outputs (for example, if the function outputs two outputs, you successfully find this and check both outputs for correctness. You are guaranteed that both functions will output the same number of outputs, even if the outputs are wrong).
- 10 points: You account for if the student code errors. (your autograder does not crash if a student function crashes)
- 20 points: You correctly implement banned functions.
- 10 points: You account for the case where the student did not submit the function
- 35 points: You account for infinite loops in the student’s code

Hints:

- You should look at the functions `cd()`, `dir()` `feval()`, and `exists()`.
- For the inputs, see what happens when you do this:
  - `a = {[0; 1], [0; 1], 'k'};`
  - `plot(a{:});`
- For `bannedFunctions`, think about which function MATLAB chooses if there's two .m files with the same name? Is it important whether or not a function is in your folder?
- Is there a MATLAB function that can tell you how many outputs a function has?

Project Option #2: Build your own website

Have you ever been close to the end of the semester and wondered “what do I need on the final to get a _____ in this class”? In this project, you’re going to build a custom website that can help you answer just that question. Instead of the usual MATLAB, we’re going to reach outside of our normal toolbox and explore other coding languages – specifically web languages like HTML, CSS, and JavaScript. Don’t panic, we’re not going to do anything too complicated with them.

For this project, you will build a website that helps you figure out what score you need on the final to achieve a certain score in a class, and you are going to upload that website to your Prism drive. The Prism drive is the space allocated by Georgia Tech just for you. That’s right – every GT student has their own space on the Georgia Tech servers that they can use to host websites (like resume websites, blogs, etc.). For our purposes, we’re going to pretend that every class is broken down into 5 sections:

1. Tests (before the final)
2. Homework
3. Quiz
4. Class Participation
5. Extra Credit
6. Final exam

But every class assigns different weights to these categories. For example, one class might assign 30% to tests while another class might assign 85%.

You will build a user interface that allows the user to enter the percentage assigned to each category, the grade they have for that category, and what grade they want coming out of the final. Then, your website will display what score they need on the final to achieve that score in the class.

Point breakdown:

- 20 points – Website inputs
  - +10 for having inputs for each of the 6 categories above for their associated weight
    - Note: Extra credit has no weight – the value entered into “score” is added onto the final grade
  - +10 for having inputs for each of the 6 categories above for the user’s score in that category
- 10 points – Desired grade and button
  - +5 for having an input that allows the user to enter what score they want in the class
  - +5 for having a button for calculating the score needed
- 50 points – Grade calculation
  - +50 for having the correct score show when the button is pressed
- 10 points – Input validation
  - +10 for displaying an error message if the user enters percentages that total to more than 100%

- 10 points – styling
    - +10 points if the page has at least 10 different CSS style definitions
- **Note – you will receive no credit if your site is not publicly accessible at your Prism URL**
    - **Your prism URL is http://www.prism.gatech.edu/~yourGTUsername/**

Resources to get you started:

- HTML and CSS
    - Here are a few slideshows for introductions to HTML and CSS.
    - If you want more information on HTML and CSS, there are lots of great resources online. Probably more than you could go through in a year. Google is your friend here.
        - HTML inputs
        - HTML buttons
- JavaScript
    - There are also lots of great tutorials online regarding JavaScript. But since our project is simple, you will probably find a lot of success in just searching your question. Examples:
        - "How to handle a button click in JavaScript"
        - "How to get text from input in JavaScript"
- Uploading your site to Prism
    - To upload your website to your Prism drive, you can follow the walkthrough put together by OIT. It has all the information you need regarding how to set up your account and start transferring files.
- Calculating the grade
    - If I had two categories in a class, test and homework, I could calculate the grade I need on the final with the following code:

```
test_raw = test_score * (test_weight/100)
hw_raw = hw_score * (hw_weight/100)
score_needed = (desired_score - extra_credit - test_raw - hw_raw) / (final_weight/100)
```

Live Examples:

- Here's an example of a bare-bones solution (don't copy this!)
    - http://www.cs1371.gatech.edu/website/

Notes:

- You may use external libraries if you think they will help you (example: jQuery). Warning: if you don't already know how to use them, it will probably just make things more complicated.

Extra Credit Opportunities:

- 10 points – Extra CSS styling
    - At least 25 different style definitions
- 20 points – the user can enter a comma separated lists of scores into any category, and the score for that category is calculated as the average of the comma separated list

- 70 points – Allow the user to search for a class and dynamically populate category weights
    - More info below

Getting Started on the Extra Credit

For the last part of the extra credit, instead of having the user input the weight of each category, you should allow the user to choose a class from a list, and then you will go get the category weights for that class. To do this, we have built a mock API which returns the weights of classes. To learn about APIs, you can skim over the API project, but here is a summary:
"In some cases, companies (or private users) will publish APIs to interact with their data. APIs have a few different definitions, but in the context that we're dealing with, an API is just an interface that allows you interact with data on a website. In the previous example, we used a URL to read HTML content. With an API, we are again going to use a URL, but this time we are going to get nicely formatted data that is easier to process, and often can be filtered to exactly what we want."

Our API returns the data associated with the weights for a given class. You can access this API by using a specific URL like this:

http://cs1371.gatech.edu/getClassInfo/?class=className

Where "className" is the class you want the data for. Here are some example URLS, you can try them out in your browser or just follow the hyperlink:

- http://cs1371.gatech.edu/getClassInfo/?class=CS1371
- http://cs1371.gatech.edu/getClassInfo/?class=MATH1501
- http://cs1371.gatech.edu/getClassInfo/?class=ECON2100

If you try out those URLs, you'll notice that the result is not a nicely formatted HTML page like you're used to. Instead, you see a bunch of data formatted as a JSON string. This is much easier for programmers (such as yourself) to work with. Don't let JSON scare you, it is just one of the many ways to format data. Luckily for us, JSON is very easy to use in JavaScript.
The data returned from our API will have each category and the corresponding weight for that category. It will also have a "success" field that has a value of true if the class was found and false otherwise.

To earn this extra credit, you should present a list of classes to the user that they can choose from. When they select one (you can have another button for selection if you want), then your website should go get the data from the API shown above, and fill in the weights in the appropriate boxes. Here's a list of classes that our API will display fake data for: CS1371, AE1350, AE1601, AE1770, CHEM1211, CHEM1212, CHEM1315, CHEM2311, ECON1101, ECON2100, ECON2101, ENGL1101, ENGL1102, MATH1111, MATH1113, MATH1501, MATH1502, PHYS2211, PHYS2212, PHYS2213, PHIL2010, PHIL2025, PSYC2005, PSYC2020, PSYC2103, NRE2110, NRE2698, CS1331, CS1332, ECE1010, ECE2020, ECE2026.

Note that this type of request is called a "GET" request because we are passing the necessary data inside of the URL. Feel free to search around for how to make a GET request in JavaScript.

Project Option #3: Become an Excel master

In this project, you will step outside of MATLAB land and learn more about Microsoft Excel and how you can use Excel built-in functions while also learning about how MATLAB can help supplement our newly acquired Excel skills.

For the first part of this project, you will read and follow all of the steps listed in the "Excel_Project.pdf" file. This file will walk you through the basics of excel, as well as dive into some more advanced techniques that will help introduce you to the important topics.

For the second portion of this project, you will need to seek out data that includes at least both double and char types. Some resources and websites to get you started are below. You are encouraged to be as creative as possible here, pursue some of your personal interest, or relate it to your future career. Past the initial data, you must complete some math operations and general functions. The more unique the function, and relevant the math, the higher the point value (this means only using the sum function and adding two columns for no reason will not receive high marks). Remember to keep your ultimate goals in mind when manipulating your data (what do you want to show?). Absolute addressing and conditional statements should both be used at least once.

One section of your final spreadsheet needs to incorporate vlookup and conditional formatting heavily. How you choose to use those techniques is up to you, but your goal should be to impress us and use those tools in a way that shows how helpful they are. You will have a chance to explain why and how you used those tools in your report.

Based on your data, create at least two visuals (graphs) representing your results. At least one must be for data received of type char, and at least one must be for data received of type double. For at least one of your graphs, make the same type of plot/graph using MATLAB techniques taught in this course. Write a brief statement about the utility and merits of each graphing method (2-3 sentences).

Find one insight in your data that crosses multiple data sets. An example of this would be if he surveyed sales of the donut shop by time of day, type of donut, etc., and found that, of the morning buyers, 70% preferred chocolate, as compared to the afternoon buyers, of whom only 30% preferred chocolate. What might this mean? What might drive the data to be this way. Support your claim with visual graphics or spreadsheet sections (4-5 sentences).

Finally, reflect on your use of Excel and knowledge of MATLAB. Talk through some of the steps you might have taken if you had to import the data and manipulate it in MATLAB. What might be easier? What might be more difficult? This written portion should not include code, but special attention will be given to those projects that mirror some of the excel functions in MATLAB for better comparison (1 paragraph).

- www.census.gov
- http://factbook.gatech.edu/quick-facts/admissions-enrollment/
- http://m.bbref.com/m?p=XXteamsXXCHCXX2016.shtml

For the final part of this project, we are going to see the limitations of Excel and see if we can get MATLAB to help us out. Open up "random_people.xlsx" and take a look at the data. What if we wanted to find the first and last name of everyone who has an email? What about if we wanted to find the ip address of everyone with a particular area code? All of these operations are rather difficult to perform in Excel. They are not impossible, but the formulas are not pretty (they are much prettier in Google Sheets however). Instead, we're going to use MATLAB to solve this problem. Using MATLAB, perform the following operations:

1. Find a cell array of first names for those people who have a valid email (the email column is not blank)
2. Find a cell array of the ip addresses of those who have a homework that starts with the letter “S”
3. Find a cell array of emails for those who have a favorite website that is NOT a .com, and who are female.

However, there is one catch here. You may not use any iteration for these problems. That’s right, no for loops or while loops. Hint: the `cellfun` function will be extremely useful here. Combine that with logical indexing and you just saved yourself the trouble of writing loops over and over again.

Extra Credit:

- **(35 points)** Do one more `cellfun` operation.
    - Take a look at the Excel file 'chipotleOrders.xls'. This file contains real Chipotle orders that you will be analyzing.
    - You want to determine what the most popular thing at Chipotle is. You could just determine the most popular item (Barbacoa Burrito, Steak Bowl, etc), but you want to go a step further and determine not only the most popular item, but also the most common combination of toppings for that item. However, the catch is that, like with the previous cellfun portion of the project, you MAY NOT use iteration at all to do this. Feel free to use cellfun and search around for other functions that may help you in your endeavour.
    - The final output of the function that you write should be a string that says the most popular item (the item that occurs the most) and the list of the most common set of toppings for that item. Note that you are not looking for simply the most common toppings, but rather you are looking for the most popular COMBINATION of toppings for the most popular item.
- **(30 points)** Using Excel Macros, create a “table of contents” sheet with buttons to navigate between the different sheets of your document for part 2. There should be a button for each sheet, and when clicked, it should navigate to that sheet automatically.
- **(35 points)** Redo all of the calculations you did with MATLAB (using `cellfun`) in Excel using Excel formulas. Make a new column for each calculation.

Project Option #4: Import data from the internet and learn about APIs

Throughout the semester, we have used MATLAB to process data, but that data has always been given to you directly. Many times you'll want to use MATLAB to process data, but you have to get the data off a website first! This project will be two parts: reading a given website to extract data from it, and an open-ended portion of accessing some API to get whatever data you would like.

In the first part of this project, you should pull the top twenty nonfiction bestsellers from the New York Times. The list of best sellers is available at: http://www.nytimes.com/books/best-sellers/combined-print-and-e-book-nonfiction/?action=click&contentCollection=Books&referrer=https%3A%2F%2Fwww.google.com%2F&region=Footer&module=WeeklyListsIndex&version=Nonfiction&pgtype=Reference.

All webpages are made up of HTML content, which is what you will want to pull down into MATLAB, and process accordingly. You don't need to necessarily understand how the HTML works, just treat it as an ugly string that you have to process to get the data that you want.

After pulling the data, create a structure of the top 15 books with their rank, name, authors, and description. The exact format of the structure is not important so think about what the best design would be!

- A helper function, `nyTimesParser`, has been provided that will cut down some of the parsing. This function takes in the HTML for the entire webpage and returns only the relevant chunk of HTML that contains the data you need.

Now one question you may have at this point is "do I always have to go through ugly HTML parsing when I want to pull data from the internet?". Luckily, the answer to that question is no. In some cases, companies (or private users) will publish APIs to interact with their data. APIs have a few different definitions, but in the context that we're dealing with, an API is just an interface that allows you interact with data on a website. In the previous example, we used a URL to read HTML content. With an API, we are again going to use a URL, but this time we are going to get nicely formatted data that is easier to process, and often can be filtered to exactly what we want.

A nice example of this is the OMDb API, which provides data about movies. Try going to this link: http://www.omdbapi.com/?t=Interstellar. What you'll notice is that instead of seeing a nicely formatted HTML page, you see a bunch of data formatted as a JSON string. This is much easier for programmers (such as yourself) to work with. Don't let JSON scare you, it is just one of the many ways to format data, and luckily many people have already written awesome libraries that do the JSON processing for us. Yes, you are allowed (and encouraged) to use this JSON parsing library (or any JSON parsing library) on your project. How did the website know which movie we were looking for? If you look closely at the URL, you'll notice the last part "t=Interstellar". This is one of the many ways to pass data to an API – through the URL This is called a GET request, but that's all we'll get into for this project. By changing the name after the equal sign, you change what movie you are looking for.

For the second part of this project, write a function that takes in a movie name and outputs some interesting data about that movie using the OMDb API. It doesn't matter how you output the data. Think about what would be considered good design if you had to turn this in to your boss! What should happen if the movie doesn't exist? Make sure you account for that!

Homework 15– Project Homework

For the last part of this project, pick any other API and write some cool function to interact with that API. Notice that most APIs require API keys, which can usually be obtained by signing up for a free account. If you have to get an API key that is okay, but clearly state how you went about getting that key so we can reproduce it and give you a grade. The API doesn't matter (Twitter, Spotify, OpenWeatherMap, etc), but rather the cool way in which you interact with it! For example, many developers use a Twitter API to access trending topics or tweets mentioning a specific person.

Extra credit opportunities:

- +20 – update your OMDb API function to show the Rotten Tomatoes rating
- +30 – update your last function to pull data from two different APIs and correlate them in some meaningful way
- +30 – write `nyTimesParser`
- +20 – update your first function to additionally pull the isbn for each book (also available in the HTML)

Some things that may help:

- `webread`

- Regular expressions may be helpful (but not mandatory)
- HTML is very structured, which should help with your string parsing
- Many JSON libraries are already available for MATLAB, don't reinvent the wheel!

Project Option #5: Learn about 3D animation and build your own plane simulator

In this project, you will use your MATLAB skills to build an animated representation of an airplane flying. This homework will be different from the homework assignments you have completed up until now. Instead of having many problems to solve with different functions, you will be graded only on one function, `Bleriot()`.

The function `Bleriot()` should be able to make the body of a propellor plane and animate the movement of the plane as it flies through the air. There will be more instructions on how to do this later on. You are not required to have any other functions besides `Bleriot()` when you hand in your files, but you may submit helper functions to aid your main function.

Before you begin writing your plane simulator function, you should and must first finish the ABCs for surface plotting and animation. There are 3 ABC files total, one on surface plotting and two on animation. Unlike other ABCs you have had this semester, there is no autograder to check your answers for these. The only way to see if you completed the ABCs correctly is to run the solution code (`.p` files) and visually compare your answers. Ensure that you load the `ABCs_surfacePlotting.mat` file and run the surface plotting ABCs with inputs when testing.

Once you have completed the ABCs, refer to the Bleriot function description further down. You have also been provided with a grading scheme and example extra credit opportunities. You can earn up to 100 points extra credit to get a max score of 200 on this assignment.

Notes:

You have a lot of freedom in this project to change things up and be creative. At the very minimum, you must show the plane existing on screen, have it move, perform at least one "roll", one loop, and move as a cohesive unit. The animation must be plotted 3 dimensionally. Failure to do this, and handing in a 2D animation will result in a max grade of 50.

Grading Scheme:

Ten percent (10%) of your grade for this project will come from your completion of the ABCs (hand-graded). Another Sixty percent (60%) will come from a hand-graded score of your `Bleriot` function.

The remaining thirty percent (30%) of your grade will be based on a write-up that answers specific questions and explains how your plane simulator function works. Make sure you look through the rubric provided to ensure you have fulfilled all the requirements for this project homework.

Extra Credit:

The animation aspect of this homework will give you an opportunity to be creative. If you go above and beyond what the problem statement specifies, there is potential to receive extra credit. Extra credit will be awarded by your TAs based on how many creative additions you make to your plane simulation. You should make note of any extra credit you have incorporated at the bottom of your file (below all of the code) as comments, in addition to mentioning it in your write-up. ***The TAs are NOT responsible for helping you implement your extra credit!***

Function Description:

Function Name: `Bleriot`

Inputs:

None

You may have more inputs to your function.

Outputs:

None

Plot Outputs:

1. Animated 3D plot of a plane flying

Function Description:

After completing the ABCs for surface plotting and animation, you should now feel comfortable making the body of a plane on a 3D plot and animate it flying.

You have been provided with a Bleriot.m file that contains template code for setting up your animation. There are several helper functions listed at the bottom of Bleriot.m. These functions are there to help you create your animation. You do not have to use the given functions, as long as you create your animation in the way that is specified in the next section. Here is a complete list of the given functions and a quick description:

- `drawProp`: draws a propeller that is already defined variable
- `drawPilot`: draws a pilot that is an already defined variable
- `drawWheel`: draws the wheels with an already defined variable
- `drawTail`: draws the tail with an already defined variable
- `drawWing`: draws the wings with an already defined variable
- `drawBody`: draws the body of plane with an already defined variable
- `plane`: a helper function to help draw the body of the plane
- `rsurf`: a function that creates the 3D arrays for the drawing
- `rplot3`: a function that plots(which includes the rotation
- `makePilot`: outputs a variable containing the pilot array
- `makeWheel`: outputs a variable containing the wheel array
- `makeProp`: outputs a variable containing the propeller array
- `makeWing`: outputs a variable containing the wing array
- `make_clouds`: outputs a variable containing the cloud array
- `draw_clouds`: draws the cloud with an already defined variable
-

You are encouraged to look through the code to see how they may help you.

The file `Bleriot_sample.p` has also been provided for you to see an example of what your animation may look like.

Homework 15– Project Homework

Requirements:

- The plane exists on the screen and has all the necessary components (body, wings, tail, propeller)
- The plane moves
- The plane performs at least one “roll”
- The plane performs at least one “loop”
- The plane moves as a cohesive unit
- The plane stays visible on screen throughout the animation (viewing purposes)
- There is at least one cloud in the animation
- The cloud moves (as to make the plane look like it is flying)

Extra Credit – MAXIMUM of 100 Points (up to 10 points each)

- Enhance the background (add a sun, create a sunset feel)
- Add a skydiver
- Create a “dog-fight” with multiple planes
- Add multiple planes doing different things
- Change the model and design of the airplane (like to a Lockheed SR-71 Blackbird)
- etc.

Rubric:

ABCs:

	Points Available
ABCs_surfacePlotting	5
ABCs_animation1 & ABCs_animation2	5

Function:

	Points Available
Plane exists on the screen	15
Plane moves	15
Plane performs at least one “roll” and one “loop”	10
Plane moves as a cohesive unit	10
Plane stays visible on screen throughout the animation	5
There is at least one cloud in the animation	5
Extra credit	100

Write-Up:

	Points Available
Explains how you made the plane and placed it on a 3D plot	10
Explains how you managed to move your plane appropriately	10
Explains the most challenging aspect of making the function	5
Explains anything else that was relevant to your clock, including extra credit.	5

You may format your write-up in any way, and use as many images as you want, so long as you address everything. Please limit your write-up to 2 pages.

Project Option #6: Write your own “Choose your own adventure” (text-based input) game

Do you miss the days when games didn't involve complicated graphics and a fancy console to play? Simpler times, when we still used flip phones and the having a Tamagotchi made you the coolest kid in school? Well then this project is for you! Hearken back to times of old, and create a game that lets you follow a story down multiple different paths, like a choose your own adventure story book. If you have never played one of these late 90's to early 2000's games, here are some classic examples for you to check out.

Emily is Away
Zork

Here are the basic requirements for you to receive full credit for this project:

- There must be at least 15 unique endgame options.
- There must be at least 10 user input steps.
- You must have at least one step that includes a user selecting a location on an image.
- You must have at least three user inputs that allow the user to type in any string.
- You must have at least two user inputs that allow the user to select a single option.
- You must have at least one user input that allow the user to select a value within a range using a slider bar.

Extra credit opportunities:

- +50 – every time the user is asked for input, account for any problems that may arise. For example, if you expect a double but get a string, keep asking for input. No user input should cause your function to error
- +10 – incorporate Professor Smith and Rogers into your game in some fun (and respectful) way
- +30 – at least 25 different endgame options
- +10 – incorporate a few random states (inputting the same answer doesn’t always result in the same state)

Some functions to get you started:

- `inputdlg`
- `uicontrol`

- `ginput`
- `rand() > 0.75` will return true 75% of the time

Project Option #7: Choose your own project!

Have a really cool idea for a project? Want to do a slight variation of one of the projects listed above? Great! We highly encourage you to use those creative juices to come up with your own amazing project ideas and we'll even give you a homework grade for it. However, in order to do your own project**, you must get it approved beforehand**. To get project approval, you must email Ryan Williams (rwilliams306@gatech.edu) and copy your TAs no later than Monday, April 18, 2017 at 11:59pm. Any submissions after this time will not be accepted. You must include the following information in your description:

- Awesome project title
- Project description
- Deliverables (what are you going to turn in)

You will receive a follow-up email if your project is approved or denied. Note that your project must be on the same workload level as those listed above to be approved (e.g. Pythagorean's Theorem function from Homework 1 will not be accepted!). If you want to receive extra credit, your project must go above and beyond! Make sure to list how many points you hope to receive in your request.