**ENGR 102 Sect 508 Lab 8a**

**100 points**

**Reading assignment:**

|  |  |
| --- | --- |
| **Lecture Slides** | **L08** |
| **zyBook chapter 8** | **Complete all participation and challenge activities** |

*Attention!!*

*Team submission. one submission per team.*

*Submit* *your Py-files together with your word/pdf file with screenshots of your tests outputs. Include any derivations, comments and supplemental notes in your word/pdf files.*

*No pictures by the phone – it is impossible to read. You will be allowed to resubmit and reupload HW as many times as you want to within the due date/time, only last submission will be graded. No late submissions. For submission you may use this file as template: rename file including your name. Do not forget to put your name inside of this file as well. If it is a team work use Team Header, include the team number and all team members.*

**Activity #1[30 points]: An Engineering Design Problem – to be done as a team**

This activity is meant to help you practice top-down design, outside the context of programming. We’ll look at an engineering challenge, and your team is to use a top-down design approach to describe how you would go about approaching the challenge. This is meant to be very broad and open-ended, and you are not expected to know the details of how individual pieces of this might work. The idea here is to get you to think of the **problems** that would need to be solved in order to meet this larger challenge. The goal is for you to be able to take a large, broad problem, and decompose it into smaller problems, repeatedly, until you have reached problems that seem small enough that they have a more straightforward solution.

Your team should pick **one** of these 3 options (each of these exists but the assumption is that you are not familiar with them already):

* An autonomous personal movement device – essentially an autonomous wheelchair. The device should operate with the same capabilities as a standard motorized wheelchair, but be able to operate autonomously, rather than being controlled directly by the person in the chair.
* A system for automatically inspecting rigid parts coming off of an assembly line, to examine whether they match some shape design. The process should not require human intervention, and you can assume pieces are coming off on a conveyor belt or some similar device.
* A system for retrieving devices that are at the bottom of the ocean. You can assume that the devices are emitting some sort of signal of your choice.

As a team, come up with a hierarchy stating the problems to be solved in creating such a system. Your base in this case should be something like “Autonomous Wheelchair”, with the next level of the hierarchy being the major problems that you would need to address in creating the device. Each of those problems should be further decomposed into smaller problems and so forth.

Your hierarchy should have a minimum of 5 levels, and at most 7 levels (not every branch of the hierarchy needs to go to the maximum depth, but many should). Because this is a very broad problem, in theory you could go into far more detail; that is not needed here. However, the “leaves” of your tree should be reasonably straightforward and less open-ended problems to be solved (e.g. things that have been done before, might have an off-the-shelf solution, or might still be new things, but at a level that it’s reasonable to expect a small team of people to be able to address them).

This should be done as a team, and the hierarchy should be written in a document that you submit in PDF form. You may use any office software (e.g. Word, Powerpoint) that you find most useful in creating your hierarchy.

**Activity #2: [70 points] Top-Down Design of a Program**

Following the process described in the lecture, you should, as a team, perform a top-down design for a program. Then, construct code for the program.

You want to create a program that first lets a user enter arbitrary data points (as x, y values), and then will provide a linear interpolation/extrapolation from those points to determine a value at any point a user gives. That is, the user should be able to enter **any** x value and get a y value back, such that the y value is the best estimate (using a linear estimation) from the nearest x value(s).

You are to create a document that includes parts a-c and g.

1. First, as a team, develop a top-down design for the program. Develop a hierarchy for the individual pieces, breaking each one down into as small of a piece so that the code for that portion of the program is “obvious”. You should put this hierarchy into a document (that you will submit as a PDF file later).
2. Next, as a team, determine what variables you will use for the main sections of your code. You should decide on the main data that you will need to keep that will be used in more than one “node” of the design. Write a list of these, along with a very brief (one sentence or less) description of what that variable will store. Note that you do not need to decide on variables that will be used only within one “node” of a program (e.g. you don’t need to describe a loop iterator if it is not going to be used outside the loop). Again, this should go into your initial document.

x: the value at which the user wants to find y

coordinates: a 2D matrix for the first and final coordinate (user input)

x\_initial = the x value for the intial point

y\_initial = y value for initial point

x\_final = x value for final point

y\_final = y value for final point

interpolation = the calculated value for the interpolation between the 2 points (that the user previously input into that 2D matrix)

y = the final value of y calculated by the program

1. As a team, discuss briefly the test cases you believe you will need to allow for. Write down a list of exceptional/edge cases that you need to be able to handle.

**Exceptional:** when the first and last points are the same, and therefore the line is continuous and has a slope of 0. (2,5) and (2,5)

**Exceptional:** When points are different but the y coordinates stays the same and therefore the graph is continuous. (2,5) and (7,5)

**Exceptional:** when y changes and x stays the same (vertical asymptote) this is not a function and therefore we can’t find the value of y at a given x. (2,5) and (2,7)

**Typical:** initial (2,5) final (10,20)

1. As a team, create one file with the comments/outline of the code. Share this file among all four team members.
2. Next, divide the coding tasks among all four team members, so that each person has a different section of the program.
   1. Divide the “leaf” nodes among the team members so that each person has approximately the same number of items to implement.
   2. If you have done a good job with the top-down design, and in specifying the variables that will carry over from one section to the next, then people should be able to write code for just their own section without seeing the other sections of the code!
3. Once each person has written their own separate code, bring the files together as a team, and have one person combine all the code into one program.
   1. You will likely need to thoroughly test and debug your code together as a team at this point.
4. As a team, write a short summary (a few sentences – about 1 paragraph for each)
   1. Describing the difficulty with which your team was able to combine the code at the end. Did this provide your team any insight into how the design itself might have been specified more clearly? We were able to use everyone’s part of the code without any errors when we combined it. But maybe if it was a more complex program we would have had more difficulties and we decided we should be more specific next time.
   2. Describing any benefits and drawbacks you saw into dividing the coding like this. Can you see reasons why this might be a good idea? Can you see reasons why this might be a bad idea? The benefits we saw was that since everyone did only a small part of the overall program the workload was a lot less on everyone. The disadvantages we see is that if we wouldn’t have been together when we planned the structure of it our description might have not been detailed enough for all the members to fully understand what they were doing.

Submit:

1. Your document for parts a, b, c, and g as a single word/PDF file,
2. The single debugged file from part f.

Code:

*# By submitting this assignment, all team members agree to the following:  
# “Aggies do not lie, cheat, or steal, or tolerate those who do”  
# “I have not given or received any unauthorized aid on this assignment”  
#  
# Names: Alexia Perez  
# Bethany Gawalis  
# Sam Lyzzaik  
# Tyler Scataglia  
# Section: 508  
# Assignment: Lab 8a  
# Date: 18-10-2018***from** math **import** \*  
**import** numpy  
  
*# Define the variables we are going to use - Code by Bethany Gawalis*first\_coordinate= str(input(**"What is the first coordinate: "**))  
first\_coordinate = first\_coordinate.split(**","**)  
second\_coordinate= str(input(**"What is the second coordinate: "**))  
second\_coordinate= second\_coordinate.split(**","**)  
  
*# Get a specific number for x and y from each coordinate - Code by Tyler Scataglia*x\_initial = int(first\_coordinate[0])  
y\_initial = int(first\_coordinate[1])  
x\_final = int(second\_coordinate[0])  
y\_final = int(second\_coordinate[1])  
  
*# Calculate the interpolation between the 2 points - Code by Alexia Perez***if** (y\_final-y\_initial)==0:  
 print(**"We assume the slope is 0 and therefore y will be constant: "**,y\_initial)  
**elif** (y\_final-y\_initial)!=0 **and** (x\_final-x\_initial)==0:  
 print(**"Your graph is not linear, therefore we can't calculate the value of y."**)  
**elif** (y\_final-y\_initial)!=0 **and** (x\_final-x\_initial)!=0:  
 interpolation = (y\_final-y\_initial)/(x\_final-x\_initial)  
*# Get user input for the x value at which y should be evaluated* x = int(input(**"At what value of x do you want to evaluate y: "**))  
  
*# Get the equation to find y - Code by Sam Lyzzaik  
# Equation to find y => y-y\_initial = interpolation(x-x\_initial)  
# if we solve for y we get:* y = (interpolation \* (x - x\_initial)) - y\_initial  
*# Print the result and tell the user the program is finished:* print(**"The value of y at x="**, x, **"is:"**, y)  
  
print(**"Finished!"**)

Output (All Test Cases):







