Problem Set 07 · UDFs, Selection Structures, and Flowcharts

Instructions:

1. Each problem in this problem set has a set of deliverables for you to submit. You are responsible for following the appropriate guidelines and instructions below. Create appropriately-named files as instructed.
2. Save all files to your Purdue career account in a folder specific to PS07.
3. Compress all deliverables into one zip folder named **PS07_yourlogin.zip**. Submit the zip file to the Blackboard drop box for PS07 before the due date. *REMEMBER*:
   - Only include deliverables. Do not include the problem document, blank templates, etc.
   - Only compress files into a .zip folder. No other compression format will be accepted.

Deliverables List

Item	Type	Deliverable
Problem 1: Sit-to-Stand Device	Paired	PS07_sitstand_exec_*yourlogin1*_*yourlogin2*.m PS07_sitstand_exec_*yourlogin1*_*yourlogin2*_report.pdf PS06_sitstand_com_your*loginW*_*yourloginZ*.m PS06_sitstand_springs_*yourloginX*_*yourloginY*.m
Problem 2: Global Distances	Individual	BACS_inputCoords.p BACS_dispDists.p PS06_nav_distances_*yourlogin*.m PS07_nav_exec_*yourlogin*.m PS07_nav_exec_*yourlogin*_report.pdf
Problem 3: Security Camera Placement	Paired	PS07_observatory_*yourlogin1*_*yourlogin2*.m PS07_observatory_*yourlogin1*_*yourlogin2*_report.pdf Test Cases (submitted in Answer Sheet)
Problem 4: Fractional Distillation	Paired	Flowchart (submitted in Answer Sheet) Test Cases (submitted in Answer Sheet)
	Individual	PS07_distillation_*yourlogin*.m PS07_distillation_*yourlogin*_report.pdf

Problem Set 07 · UDFs, Selection Structures, and Flowcharts

Answer Sheet

You must place your flowcharts and test cases in the Answer Sheet provided in the Assignment Files. The answer sheet is named PS07_answersheet_template.docx. You must resave it as **PS07_answersheet_*yourlogin*.docx** before submitting it.

- Complete the assignment information at the top of the answer sheet
- Your answer sheet will be used for all problems in this set
- Follow any additional instructions on the Answer Sheet
- Place items in the correct locations on the Answer Sheet

Problem 1: Sit-to-Stand Device

Paired programming

Learning Objectives

Variables	02.00 Assign and manage variables
Arrays	03.00 Manipulate arrays (vectors or matrices)
Text Display	05.00 Manage text output
User-Defined Functions	11.03 Create a user-defined function that adheres to programming standards
	11.04 Construct an appropriate function definition line
	11.05 Match the variables names used in the function definition line to those used in the function code
	11.06 Execute a user-defined function
	11.07 Create test cases to evaluate a user-defined function
	11.09 Track the passing of information to and from a user-defined function
	11.10 Break a problem into a series of sub-functions
	11.11 Coordinate the passing of information between functions

Problem Setup

In PS06 Problem 2, your team was divided into two pairs. Pair WZ wrote a user-defined function that calculates the scaled-lengths for a sit-to-stand device. Pair XY wrote a user-defined function that calculates spring stiffness values for the device.

In this problem, you are paired with someone from the other pair. You should have one program from PS06 and your partner should have the other. You will need a working version of both UDFs to complete this problem. Your job is to write an executive function that calls both the scaled-length UDF and the spring stiffnesses and prints the results to the Command Window. The figure below shows how the functions must interact with each other.

Problem Set 07 · UDFs, Selection Structures, and Flowcharts

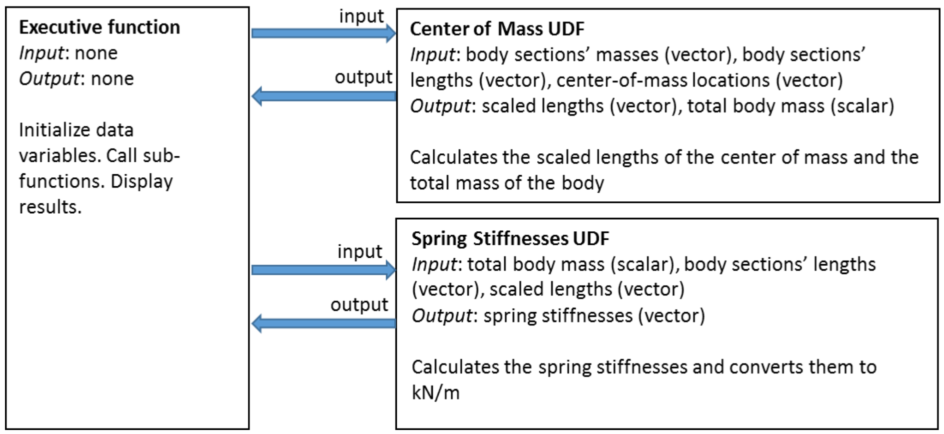

Figure 1. Function Interaction Details

Table 1 shows the sample test patient's measurements that should be used.

Table 1. Sample body measurements

Section	**Length of section (m)**	**Mass of section (kg)**	**Center-of-mass location (m)**	**Adjustable length, d (m)**
Shank, *s*	$l_s = 0.421$	3.1	$l_{cs} = 0.55l_s$	0.75
Thigh, *t*	$l_t = 0.432$	7.39	$l_{ct} = 0.59l_t$	
HAT, *H*	$l_H = 0.8$	24.13	$l_{cH} = 0.41l_H$	

Problem Steps

Now that you have a UDF to calculate the center-of-mass scaled lengths and a UDF to calculate the spring stiffnesses, you can write an executive function (i.e. a no-input, no-output function) to allow you to call the sub-UDFs and display the calculation results.

1. Open your PS06_sitstand_com and PS06_sitstand_springs functions. Examine your solution and make any necessary corrections to the code.
2. Using *PS07_sitstand_exec_template.m*, create an executive function and name it **PS07_sitstand_exec_*yourlogin1_yourlogin2*.m**. Using the proper code sections within the template, the executive function must
    a. Initialize the sample patient measurements.
    b. Call the center-of-mass UDF and the spring stiffness UDF,
    c. Prints the results of the parameters to find the center of mass of the system and choices of spring stiffness for each of the spring positions
3. Once your code is debugged and working, run your function. In the `COMMAND WINDOW OUTPUT` section of your **executive function**:
    a. Paste as comments the function call and the displayed results for center of mass and spring stiffnesses.

Publish your **executive function** as a PDF file using the file name indicated in the Deliverables list.

Problem 2: Global Distances

Individual programming

Learning Objectives

Variables	02.00 Assign and manage variables
Arrays	03.00 Manipulate arrays (vectors or matrices)
Text Display	05.00 Manage text output
User-Defined Functions	11.03 Create a user-defined function that adheres to programming standards
	11.04 Construct an appropriate function definition line
	11.05 Match the variables names used in the function definition line to those used in the function code
	11.06 Execute a user-defined function
	11.07 Create test cases to evaluate a user-defined function
	11.09 Track the passing of information to and from a user-defined function
	11.10 Break a problem into a series of sub-functions
	11.11 Coordinate the passing of information between functions

Problem Setup

Boiler Aeronautics Consulting Systems (BACS) wants you to continue working on the navigation software that you started in PS06 Problem 3. They want to build on the work you did on PS06_nav_distances but want a user input display and don't want the results displayed in the MATLAB Command Window. Instead, they want you to create an executive function that:

- opens a dialog window where a user inputs their desired airport codes and latitude and longitude values,
- calculates the great circle and rhumb line distances, and
- displays the results in a dialog box.

BACS is on a rushed timeline. Your manager decides that you must use BACS legacy functions for the input dialog function and the display function. Each function is in BACS function library as p-code, which is a protected file type that can be called within MATLAB but whose code you cannot read. P-codes do not have help lines due to the protection process, so below is the help documentation for each function.

BACS_inputCoords.p

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Program Description
%   This function displays a dialog box that allows a user to input
%   the airport code, latitude, and longitude for the starting
%   airport and the destination airport. The function returns those
%   values for use in other code.
%
% Function Call
%   [startLoc,startLat,startLon,endLoc,endLat,endLon] = BACS_inputCoords
%
```

Problem Set 07 · UDFs, Selection Structures, and Flowcharts

```
% Input Arguments
%   1. None
%
% Output Arguments
%   1. startLoc = starting airport code (string)
%   2. startLat = starting location latitude (decimal degrees)
%   3. startLon = starting location longitude (decimal degrees)
%   4. endLoc = ending location (string)
%   5. endLat = ending location latitude (decimal degrees)
%   6. endLon = ending location longitude (decimal degrees)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

BACS_dispDists.p

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Program Description
%   Displays the great circle and rhumb line distances between two
%   airports in a message box. Will cause the Command Window to 'wait'
%   until the user clicks OK in the message box.
%
% Function Call
%   BACS_dispDists(loc1,loc2,gc_dist,rl_dist)
%
% Input Arguments
%   1. loc1 = starting airport code (string)
%   2. loc2 = destination airport code (string)
%   3. gc_dist = great circle distance between locations (km)
%   4. rl_dist = rhumb line distance between locations (km)
%
% Output Arguments
%   1. None
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

As an aid, you have been provided with the PS06 table of airport codes and latitudes and longitudes. Use this information to test and debug your code.

Airport	Code	Latitude (decimal degrees)	Longitude (decimal degrees)
Chicago – O'Hare International	ORD	41.978603	-87.904842
Frankfurt International	FRA	50.026403	8.543131
Tokyo International	HND	35.5523	139.78
Los Angeles International	LAX	33.942536	-118.408075
Dubai International	DXB	25.2528	55.3644

Problem Steps

1. Open your PS06_nav_distances code. Examine your solution and make any necessary corrections to the calculations. Remove the print commands. Remove the airport codes from the required inputs for the function because they are not needed without the print commands.
2. Using *PS07_nav_exec_template*.m, create an executive function that calls `BACS_inputCoords`, your revised PS06_nav_distances function, and `BAC_dispDists` in the correct order to obtain the necessary location information, perform the calculations, and display the results.

Problem Set 07 · UDFs, Selection Structures, and Flowcharts

3. Test and debug your function.

4. Publish the **executive function** to a PDF file using the file name indicated in the Deliverables list. Note that you when you publish this function, you will need to add the inputs in the input dialog box and will need to click OK on the results dialog box to complete the publishing.

References

http://www.edwilliams.org/avform.htm
http://www.opennav.com/
https://www.fcc.gov/media/radio/dms-decimal

Problem 3: Security Camera Placement

Paired Programming

Learning Objectives

Variables	02.00 Assign and manage variables
Text Display	05.00 Manage text output
Relational & Logical Operators	14.00 Perform and evaluate relational and logical operations
User-Defined Functions	11.03 Create a user-defined function that adheres to programming standards
	11.04 Construct an appropriate function definition line
	11.05 Match the variables names used in the function definition line to those used in the function code
	11.06 Execute a user-defined function
Flowcharts	15.02 Track a flowchart with a selection structure
	15.09 Create test cases to evaluate a flowchart
	15.10 Construct a flowchart using standard symbols and pseudocode
Selection Structures	16.01 Convert between these selection structure representations: English, a flowchart, and code
	16.02 Code a selection structure

Problem Setup

An astronomical society restored their historic observatory and converted it to a museum just in time for the recent solar eclipse. They hired your engineering firm to design a security system for the updated building. The building's layout, with pertinent measurements but not drawn to scale, is shown in Fig. 2.

The society wants the ability to place security cameras wherever they choose in each of the rooms. At night, all the doors will be closed and the cameras will be turned on. There are four doors: one between the office and lobby, one between the lobby and the exhibit hall, one between the exhibit hall and the observatory's vestibule, and an exterior door in the lobby.

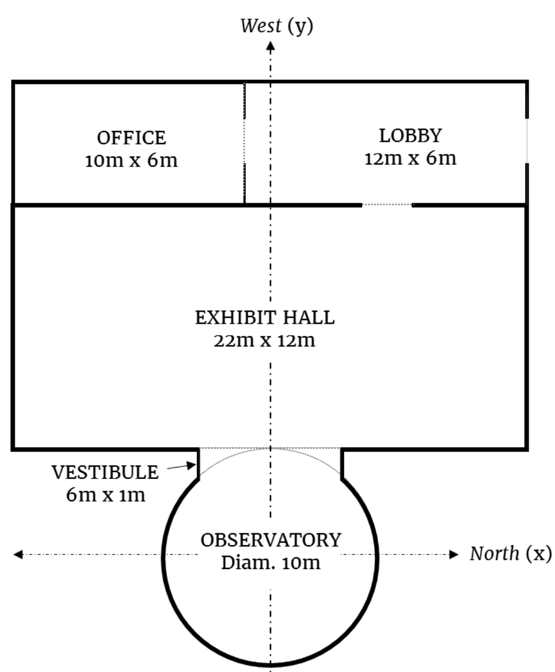

Figure 2. Museum floor plan

Each camera will communicate its coordinates to a central receiver. Code must be written to confirm the location of the

Problem Set 07 · UDFs, Selection Structures, and Flowcharts

camera based on its reported coordinates. The building's coordinate system has its origin in the center of the observatory. Note that vestibule is part of the observatory room for camera placement. Cameras must be on the ceiling inside a room; they cannot be on any wall or door or outside the building.

You must ensure that each camera has valid coordinates, regardless of its location within a room. The firm's security system planning team developed a flowchart to check coordinates, and your supervisor has approved the flowchart for the next step in development. Your task is to test the flowchart and then convert the flowchart into code.

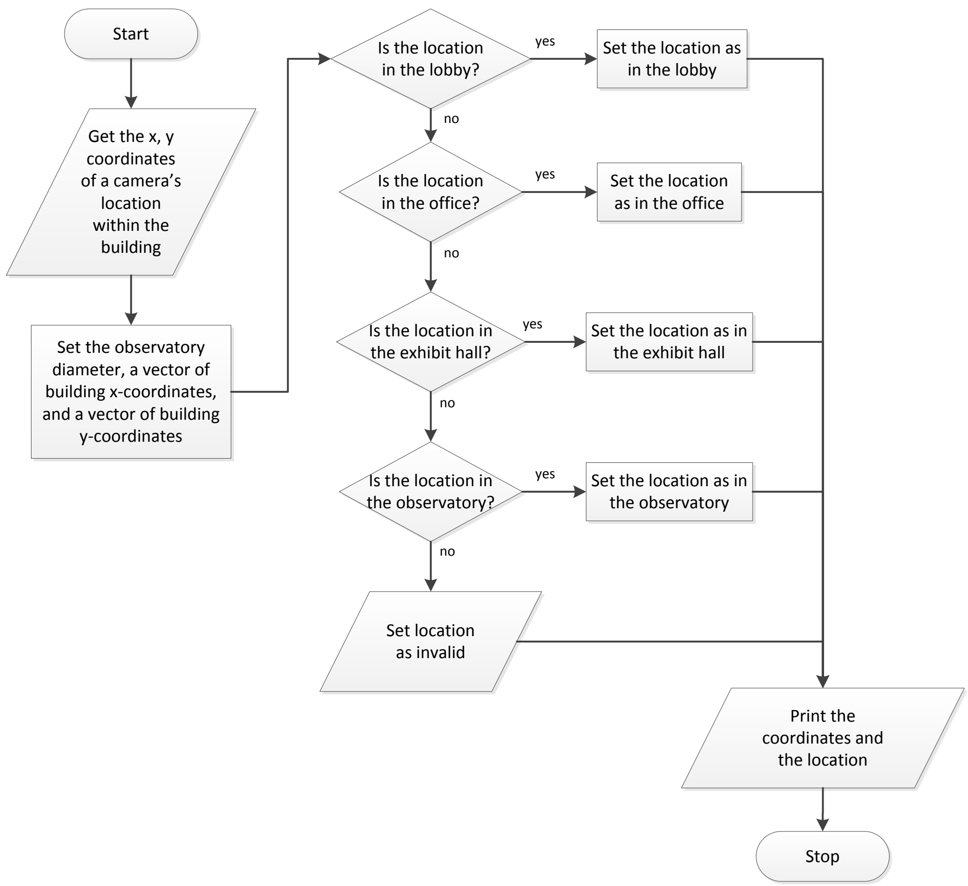

Figure 3. Security Camera Placement Flowchart

Problem Steps

1. **Before you start to code**: Review the flowchart to understand the process for determining a camera's location. Note that information is printed to the MATLAB Command Window.
2. In your Answer Sheet:
    a. Add a series of test cases to thoroughly examine all the possible paths in the flowchart.

b. Record the corresponding flowchart outputs. These outputs must be determined independently of MATLAB (using the flowchart) so that they provide a means to check and debug your MATLAB code as you write the code.

3. Translate the flowchart to a MATLAB user-defined function. Name your UDF file **PS07_observatory_*yourlogin1_yourlogin2*.m**. Your function must:

   a. Accept the x- and y-coordinates of a camera location.

   b. Print the coordinates and the location, or that the location is invalid, to the Command Window.

4. Test your function by calling it with the test cases you created in Step 2.a.

5. For each test case, paste the function call and results displayed in the Command Window as comments under the `COMMAND WINDOW OUTPUTS` section of your function file.

6. Publish your function to a PDF using any valid set of inputs and name the published file as required in the deliverables list.

Problem 4: Fractional Distillation

Paired Flowchart Creation

Individual Programming

Learning Objectives

Variables	02.00 Assign and manage variables
Text Display	05.00 Manage text output
Relational & Logical Operators	14.00 Perform and evaluate relational and logical operations
User-Defined Functions	11.03 Create a user-defined function that adheres to programming standards
	11.04 Construct an appropriate function definition line
	11.05 Match the variables names used in the function definition line to those used in the function code
	11.06 Execute a user-defined function
Flowcharts	15.01 Construct a flowchart for a selection structure using standard symbols and pseudocode
	15.02 Track a flowchart with a selection structure
	15.09 Create test cases to evaluate a flowchart
	15.10 Construct a flowchart using standard symbols and pseudocode
Selection Structures	16.01 Convert between these selection structure representations: English, a flowchart, and code

	16.02 Code a selection structure

Problem Setup

Land, air and maritime vehicles operate primarily on gasoline, kerosene and diesel. These products are derivatives from petroleum and are produced in refineries in a process commonly known as fractional distillation.

Fractional distillation is the process in which a mixture is separated into its component parts, or fractions. The process uses boiling point to separates chemical compounds by heating them to a temperature at which one or more fractions of the compound will vaporize.

On an industrial level, chemical engineers design distillation towers (shown in Figure 4) to separate petroleum into its derivatives at high pressure and temperature. This process is generally slow, complex, and expensive due to the vast amount of components that are present in crude oil.

Figure 4. Distillation Columns

A simplified distillation process is shown below:

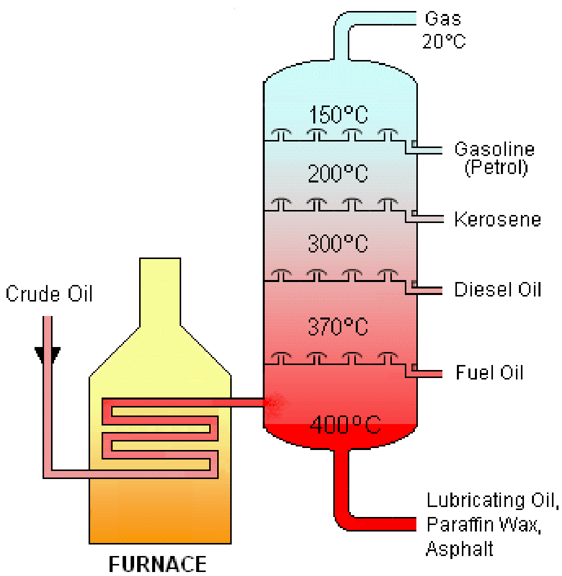

Figure 5. Petroleum Distillation Process

The compounds in crude oil have different boiling points. The mixture is heated to a high temperature to vaporize the components. Each compound vaporizes at its boiling point. The vapor enters the column and cools. As it cools, it condenses into liquid and is captured in trays.

Distillate Compound	True Boiling Point
Liquefied Petroleum Gas (LPG)	20°C
Gasoline	150°C
Kerosene	200°C
Diesel Oil	300°C
Fuel Oil	370°C
Residuals	400°C

An important factor in the distillation process is the percent distillate volume, PDV (with units % v/v, which is read as "percent volume by volume"), which is based on the crude oil's True Boiling Point (TBP). The PDV indicates the fraction of a particular volume of crude oil that has been distilled and the TBP is the temperature at which that PDV is achieved.

Problem Set 07 · UDFs, Selection Structures, and Flowcharts

For a particular petroleum batch, the following simplified correlation is used:

$$PDV = \begin{cases} 0.144(TBP) - 1.206 & ;\ 8.4\,℃ \leq TBP \leq 500\,℃ \\ 0 & ;\ 0\,℃ \leq TBP < 8.4\,℃ \end{cases}$$

You need to create a MATLAB user-defined function that will identify the distillate compound based on a true boiling temperature and calculate the PDV at that temperature (As a simplification, you can assume that the input temperature = TBP).

The UDF must:

- Accept any TBP in the interval [0, 500] °C as the input argument
- Display appropriate, useful error messages to the Command Window for invalid inputs
- Calculate the PDV using the provided correlation
- Return no output arguments
- Print the distillate name and PDV value to the Command Window for valid inputs

You can assume this in an ideal system and that a compound starts to vaporize at its exact TBP (e.g., at 300°C, the distillate is diesel oil). Any TBP less than 20°C is classified as crude oil.

Problem Steps

1. **Before you start to code**: Create a flowchart to outline how information should move through the code.
    - Work with your paired partner to complete the flowchart.
    - Draw a flowchart to:
        - Check for valid temperatures,
        - Correctly identify the distillate at the input temperature, and
        - Calculate the PDV.
    - You can draw the flowchart using any means that result in a clear image for the answer sheet. Make sure your flowchart is legible. Options include:
        - Drawing it by hand and taking a <u>clear</u> photo
        - Drawing it directly in the Word answer sheet using Word's drawing tools
        - Drawing it in Microsoft's Powerpoint, Publisher, or Visio
        - Using another flowchart tool, such as Lucidchart
2. In your Answer Sheet,
    - Paste a clear image of your flowchart
    - Select a series of test cases to thoroughly test all possible paths on your flowchart
    - Record the distillate and PDV or the error for each test case
3. Working individually, translate your flowchart into a user-defined function. Use programming standards to place code in the appropriate sections within the template.

Problem Set 07 · UDFs, Selection Structures, and Flowcharts

4. Test your function using the test cases that you specified in Step 2.
5. For each test case, paste the function call and results displayed in the Command Window as comments under the `COMMAND WINDOW OUTPUTS` section of your function file.
6. Publish your function to a PDF using any valid set of inputs and name the published file as required in the deliverables list.

References

Fahim, M. A., Al-Sahhaf, T. A., & Elkilani, A. (2009). Fundamentals of Petroleum Refining. Burlington, NL: Elsevier Science.

Jones, D. (1995). Elements of petroleum processing. Chichester; New York: John Wiley & Sons.

Green, D. W., & Perry, R. H. (2007). Perry's Chemical Engineers' Handbook (8th Edition) (8). Blacklick, USA, US: McGraw-Hill.