

Assignment #5 – User-Defined Procedures and File I/O

Overview

In this assignment, you will discover how to create your own user-defined *Subs* and *Functions*, as well as working on how to get data from a text *file*.

This assignment is to be done individually, and it is worth 10% of the course grade.

Task Description

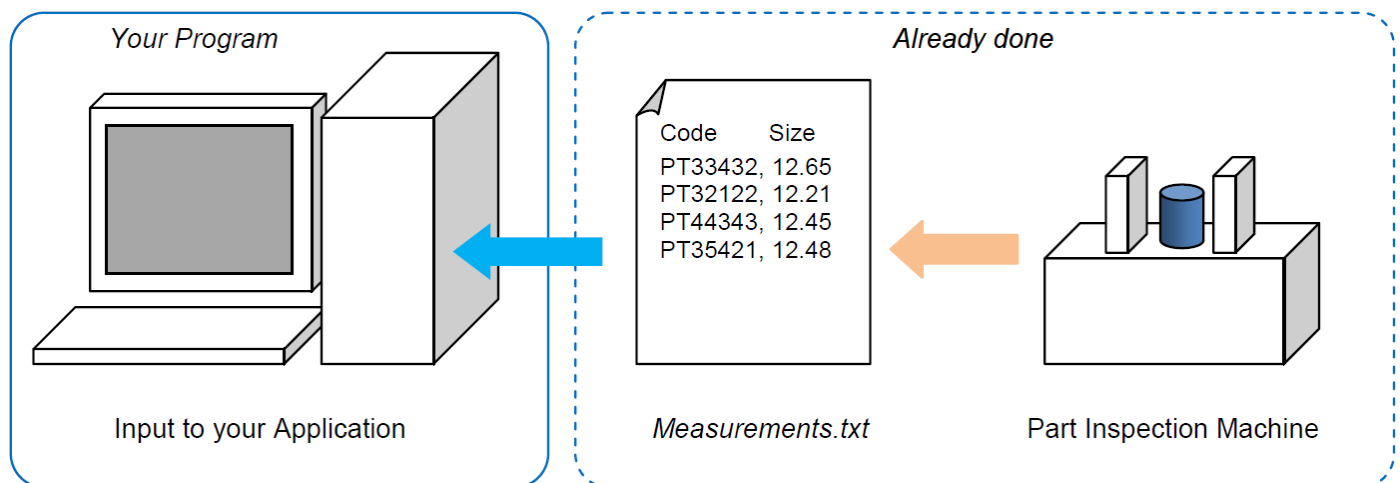
This assignment is a *statistical quality control* application that retrieves the output from a part inspection machine that has previously measured a bunch of machined parts. Using these measurements, our program will sort the individual parts into lists according to whether the part's machined size is...

- **under-sized** (in which case it is classified as scrap)
- **over-sized** (in which case it can be returned for further machining)
- **within spec** (in which case the part is acceptable)

Sometimes, data like this is read in real-time (as it is generated by the inspection machine). In other cases, the inspection machine saves its measured results to a text file which then serves as a permanent record of the measurements and which can be read and analyzed at a later date. That is what we'll do here.

The measurements from all the parts have previously been stored in a text file. The record of each part includes its *part code* and *measured size*. That data is stored in the order in which the parts are measured as they come off the machining line.

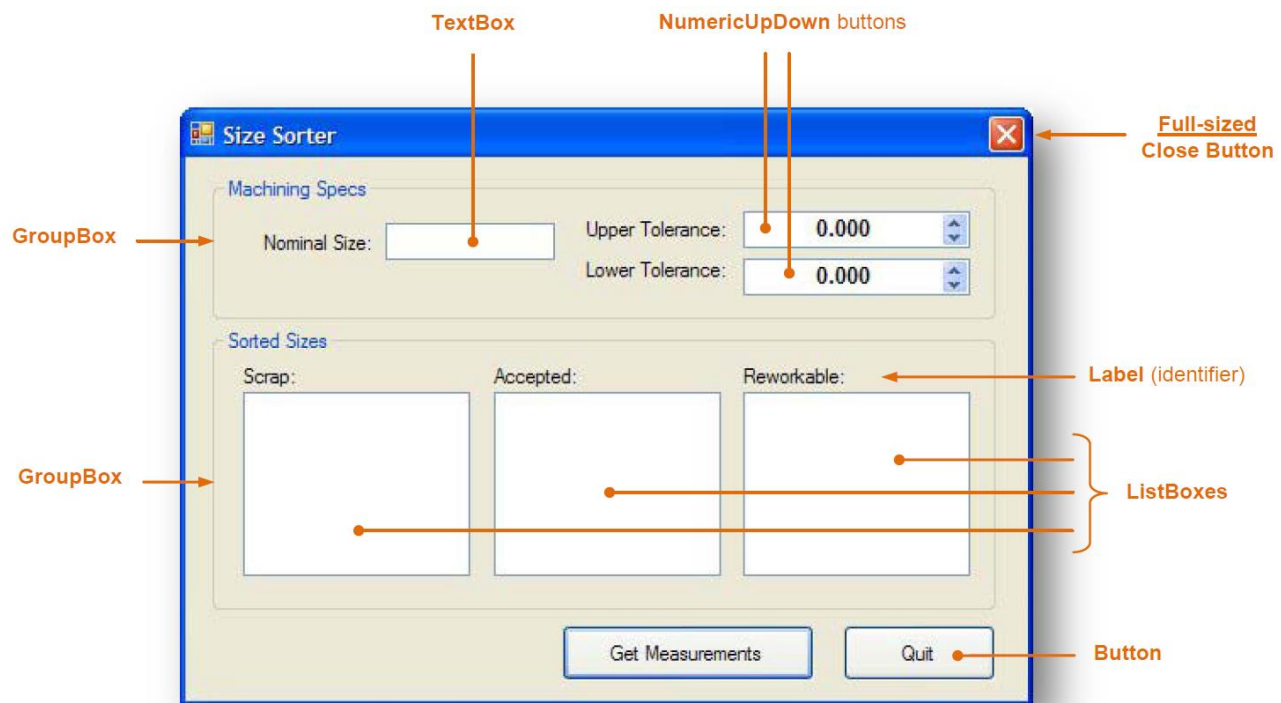
This assignment reads the data from the text file and, based on the specified acceptable size range, sorts the parts into three categories depending on whether they are *under-sized*, *over-sized*, or *acceptable*.



User Interface

1. Create a NEW Visual Basic Windows Forms App (.NET Framework), and you must name the project as **A5_SPC_PartsAnalysis_YourCollegeUsername** to be located in your College OneDrive.
2. The user interface includes two areas, as shown below. The top portion of the user interface is enclosed in a *GroupBox* that contains a *TextBox* to enter Nominal Size and two *NumericUpDown* controls to specify the tolerances; and the bottom portion is also enclosed in a *GroupBox* that contains three *ListBoxes* to display size-sorted parts.

The measured part sizes are to be read from a text file when the *Get Measurements* button is clicked. The data for each part (see below) includes its *part code* and *measured size* which are separated by a comma.



Typical contents of the *Measurements.txt*[†] File:

PT22333, 12.25

PT33433, 12.65

PT54454, 12.55

PT45323, 12.21

PT44544, 12.34

Part code and measured size for one part

[†]note that the *Measurement.txt* file is available for downloading on eConestoga.

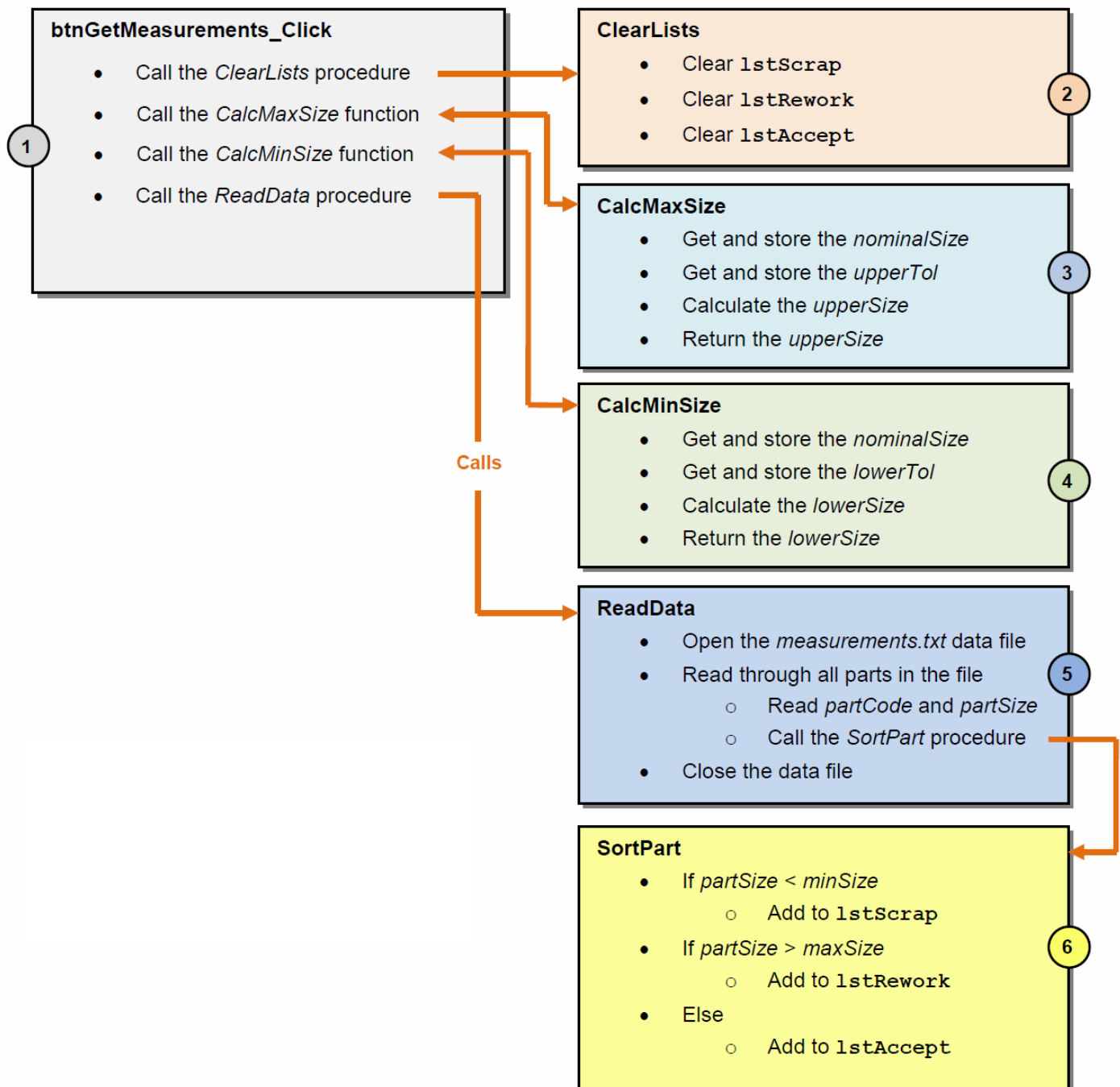
As the data for each part is read, its size gets compared to the acceptable upper and lower size limits (a combination of the *nominal size* and *tolerances*) and the data gets added to one of three lists depending on whether its measured size falls below the acceptable range, within it, or above it. Use a separate *ListBox* for each sorted list. To confirm that the part has been sorted correctly, your program will combine each part's code and measured size into a single data string (more details on page 5).

3. The table below lists all the controls to be included in the user interface with their property settings.

CONTROL	PROPERTY	SETTING
Form	Text	Size Sorter
	FormBorderStyle	FixedSingle
	MaximizeBox	False
	MinimizeBox	False
	StartPosition	CenterScreen
Group Box	Text	Machining Specs
Label	Text	Nominal Size:
Label	Text	Upper Tolerance:
Label	Text	Lower Tolerance:
TextBox	(Name)	txtNominalSize
	TextAlign	Right
NumericUpDown	(Name)	nudUpperTol
	DecimalPlaces	3
	Increment	0.001
	Maximum	0.05
	Minimum	0
	TextAlign	Center
NumericUpDown	(Name)	nudLowerTol
	DecimalPlaces	3
	Increment	0.001
	Maximum	0
	Minimum	-0.05
	TextAlign	Center
Group Box	Text	Sorted Sizes
Label	Text	Scrap:
Label	Text	Accepted:
Label	Text	Reworkable:
ListBox	(Name)	LstScap
ListBox	(Name)	LstAccpeted
ListBox	(Name)	LstRework
Button	(Name)	btnMeasurement
	Text	Get Measurements
Button	(Name)	btnQuit
	Text	Quit

Program Design and Coding

4. Now let's design the program and organize code so that each distinct operation is done *in its own specialized procedure*. As we have seen in the user interface, all activities will be initiated by the *btnGetMeasurements* button, which triggers the VB *button click* event procedure that subsequently calls 3 user-defined subroutine procedures and 2 user-defined functions. The following is the program layout.



(1) The *btnGetMeasurements* event Procedure:

This procedure is initiated by clicking the *Get Measurements* button, and only calls subroutines and functions that are listed in the followings.

(2) The **ClearLists** Subroutine Procedure:

This procedure will specialize in clearing the data from the lists. That's all. No input, no processing, and no output.

(3) The **CalMaxSize** Function Procedure:

This procedure will calculate the *upper size limit* of the part (its maximum allowable size) which is the sum of its *nominal size* and *upper tolerance*. It doesn't have any input parameter, but has a return value.

(4) The **CalMinSize** Function Procedure:

This procedure will calculate the *lower size limit* of the part (its *nominal size* less the *lower tolerance*).

(5) The **ReadData** Subroutine Procedure:

Reading data from a *file* is just another form of *input*. It is a 3-step process to read data from a file, which are Open the file, Read the data from the file, and Close the file.

For this assignment, a text file that is named as *Measurements.txt* is given and available on eConestoga for you to use. This is to simulate that such a data file was generated from another data recording process when each part was inspected. The file needs to be downloaded and stored to your College OneDrive (just root directory...do not put it inside a folder), such as


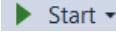
OneDrive - Conestoga College \Measurements.txt

Your program needs to store the name of this file in a **String** variable (named *dataFile*), and will read a *part code* and *measured size* for each part listed in the file.

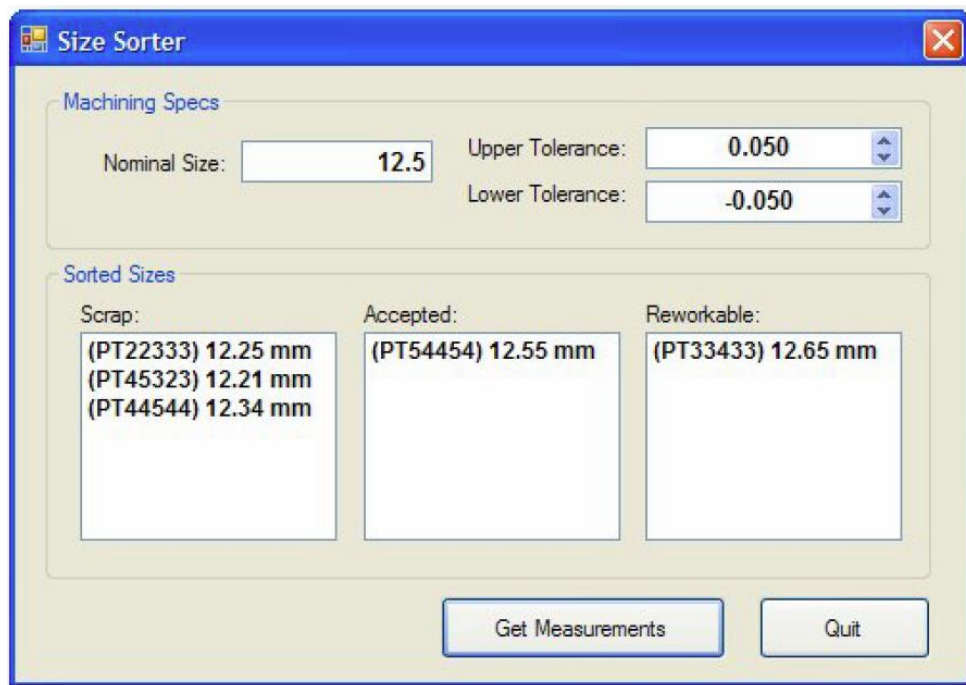
(6) The **SortPart** Subroutine Procedure:

After reading a *part code* and *measured size* for one part, the program needs to sort that data into one of the three lists based on whether the part size is too large, too small, or just right. The *part code* and *measured size* for one part need to be combined and re-formatted into one string, so that the text displayed in each list box will convey more information. Here is the full text string and how it will look as output...

The diagram illustrates the process of formatting data for display. At the top, the code snippet `"(" & partCode & ")" " & partSize & " mm"` is shown. Dashed arrows point from each part of this code to a final formatted string below: `(PT22333) 12.25 mm`. The text `(PT22333)` is in red, `12.25` is in black, and `mm` is in red. To the right of the code, the text "The combined variables and text data" is written. To the right of the formatted string, the text "The way it will look in the list" is written.

5. More events to be added in coding:
 - (1) The **Quit** button event when it is clicked. This time, you need to add a pop-up message to ask the user to confirm if s/he really wants to quit the program. If yes, close the program; if not, go back to the program and keep it running.
 - (2) Add Data Checking code to check to see if the *nominal size* value is *zero* or *less*.
6. Click **Save All** button  to save the project, and then click **Start** button  to run the program. A sample result is shown below. Enter the nominal size and tolerances as shown below to get the outputs.

Please note that the following sample result is only based on a 5-record measurement text file. When your assignment is being evaluated by the instructor, a different measurement text file that has the same format, but contains a much larger number of records, might be used. Thus, you need to make sure that your program be able to work on any text file that has the same format.



Machining Specs	
Nominal Size:	12.5
Upper Tolerance:	0.050
Lower Tolerance:	-0.050

Sorted Sizes		
Scrap:	Accepted:	Reworkable:
(PT22333) 12.25 mm	(PT54454) 12.55 mm	(PT33433) 12.65 mm
(PT45323) 12.21 mm		
(PT44544) 12.34 mm		

Get Measurements Quit

Assignment Submissions

You need to submit a PDF file of your source code with a title page, as well as uploading the zipped file of your VB project to Assignment #5 dropbox.

- a. For the PDF file, you must include your **full name**, course number, course name, etc. on the title page and on the FIRST line of your source code, as a comment.
- b. For the zipped file, refer to Tutorial #04 (available in Week 04 syllabus) on how to compress a Visual Basic 2019 project for the detailed instructions.
 - i. The due date of Assignment #5 is shown on eConestoga. Both files must be submitted by the due date – **late submission will NOT be accepted.**