MIS 322 **LAP 5** Page 1

VB.NET of 6

Name: Score: / 10pts

**Project Description**

* A Monte Carlo simulation is comprised of a large number of trials or repetitions performed that involve some level of randomness. This application will perform a simulation looking at the output of a process which includes creating a product and then inspecting the product for defects. The test to determine if a product is defective is never 100% and can either miss a defective product or result in a false positive where a product that is not defective is inaccurately deemed defective.
* Work on your own. The TAs and instructor can offer help and hints at the cost of partial points.

**Monte Carlo Simulation Application**

* The application will have two forms and one module
* Remember to include all StyleSheet elements

**MainForm Design**

* Set up the first form to resemble the one shown below in figure 1 (form color shown is BlueViolet)
* Name this form **MainForm**
* Add a menu strip to the top of the form (note the underlined letters)
  + Simulation and Exit will be the two top level menu items
  + Simulation will have two menu items below it
    - Run Sim
    - Reset Sim
  + Exit will not have any menu items under it

A picture containing text, screenshot, computer

Description automatically generated

Figure 1: MainForm Design View

**OutputForm Design**

* Set up the second form to resemble the one shown below in figure 2 (form color shown is BlueViolet)
* Name this form **OutputForm**
* The five textboxes are read-only
* Add a menu strip to the top of the form (note the underlined letters)
  + Simulation and Close will be the two top level menu items
  + Simulation will have one menu item below it
    - Rerun Sim
  + Close will not have any menu items under it

A screenshot of a computer

Description automatically generated

Figure 2: OutputForm in Design View

**Monte Carlo Simulation Functionality**

**MainForm**

* Run Sim menu click event will do the following:
  + Verify the values input for the three textboxes
    - Valid entries for the number of trials must be a positive, non-zero integer
    - Valid entries for defect rate and test accuracy must be positive numbers in the range from 0 to 100
  + Set three global variables to hold the values in the textboxes
  + Show OutPutForm in a modal mode
* Reset Sim menu click event will do the following:
  + Clear out the values in all textboxes
  + Reset any relevant class and global variables to their initial values
* Exit menu click event will do the following:
  + Close the application

**OutputForm**

* The activation event of OutputForm will do the following:
  + Run MonteCarloSim independent sub procedure
* Code the MonteCarloSim independent sub procedure to do the following:
  + Loop through all the products made in the trials
    - One product is made during each trial
  + The loop will call a function (see module section for product defect determination function requirements) to determine randomly if each product is defective by generating a random number and then passing it as an argument to the function
    - The random number generated will be between 1 and 1000
    - Divide the random number by 10 to generate a number that has 1 decimal place
      * Example: 57.4
  + Next the loop will call a function (see module section for testing products) to test for defective products by passing another random number (showing one decimal place just like above) and whether the product is defective or not as determined by the previous function
  + The last part of the loop will keep a running total of the four possible outcomes:
    - Part is not defective, Test shows part not defective
    - Part is not defective, Test shows part is defective (false positive)
    - Part is defective, Test shows part not defective (missed identifying defective product)
    - Part is defective, Test shows part defective
  + After the loop, set the textboxes on OutputForm as follows:
    - Products Made = Number of trials
    - Good Products = Products that are not defective (regardless of outcome of inspection test)
    - Defective Products = Defective products (regardless of outcome of inspection test)
    - Identified Defects = Defective products that are correctly identified by the inspection test
    - False Positive = Good products that are incorrectly identified as defective by inspection test
* Rerun Sim menu click event will do the following:
  + Run MonteCarloSim independent sub procedure
* Close menu click event will do the following:
  + Close OutputForm

**Module**

* Product Defect function:
  + Declare as a Boolean datatype
  + Set a variable as a parameter to accept a random number that is generated in the MonteCarloSim independent sub procedure
  + In the function, compare the random number with the defect rate variable
    - If the random number is less than or equal to the defect rate, the product is defective and the function will return True
    - Otherwise, the product is not defective and the function will return False
* Product Inspection function:
  + Declare as a string datatype
  + Set two variables as parameters to accept a random number generated in the MonteCarloSim independent sub procedure and a Boolean variable that determined if the product is defective
  + If the product is defective:
    - Compare the random number with the test accuracy
      * If the test accuracy is greater than or equal to the random number, the defect in the product has been found and the function will return “found defect”
      * Otherwise, the defect in the product is not found and the function will return “missed defect”
  + If the product is not defective:
    - Compare the random number with the test accuracy
      * If text accuracy is less than the random number, the function will return “false positive”
      * Otherwise, the product is not defective and the test has correctly identified the product as not defective and will return “no defect”

Graphical user interface, application

Description automatically generated

Figure 3: MainForm During Run Time

Graphical user interface, application

Description automatically generated

Figure 4: OutputForm After Simulation Has Run

* Zip and upload your project to Canvas when it is complete