DEPARTMENT OF ECONOMICS
AND BUSINESS ECONOMICS
AARHUS UNIVERSITY

EXAM ASSIGNMENT

Study Programme and level	**MSc Operations and Supply Chain Analytics + elective**						
Term	Winter 23-24o						
Course name and exam code(s)	Tools for Analytics						460202E016
Exam form and duration	Written exam, WOAI						3 hours
Date and time	16 January 2024						15.00 – 18.00
Supplementary material/aids	All	X	Specified		No		
Hand-in of extra material (appendix) in WISEflow allowed	Yes	X					
Hand-in of hand-written material allowed	Yes		No	X			
Anonymous exam	Yes	X	No		Please do **not** write your name or student ID number anywhere. Use your flow-id number (find this on the cover sheet in WISEflow).		
Other relevant information	**Avoid being suspected of exam cheating** Remember to state references and use quotation marks, if you copy text from other sources or re-use parts of a previously submitted exam paper (plagiarism and self-plagiarism). Students must answer the exam assignment **individually**. All submitted exam papers are checked for plagiarism, so cheating and collaboration between students will be detected. Please note that use of AI in exams at AU is not allowed.						
Number of pages (incl. front page)	6 pages						

Practical information

- This exam is with internet. You may use any on-line or book-based resource you would like, but you must include citations for any code that you use (directly or indirectly). You may not consult with anyone else during this exam. That is, you cannot ask direct questions on the internet, use answers given on the internet after the start of this exam, use AI based tools (such as ChatGPT), or consult with each other, not even for hypothetical questions.
- This assignment has an appendix available for download from WISEflow.
- Please note that the weights on each assignment are only guideline weights, and that they only provide information regarding the relative weight of the assignments. The final evaluation will be given based on the total material handed in.
- If you find that some information is missing in the assignments, you may make the necessary assumptions and clearly specify these.
- Handing in: You must hand in a single Excel macro enabled file (`.xlsm`) as “Appendix material”. Due to the system, you must also upload an empty PDF document named `yourFlowId.pdf`.
- An Excel template file is given in the appendix that you should use as a starting point. Rename it to `<your flow id>.xlsm`.

VBA - Assignment 1 (60%)

Consider the Excel template file provided. The file contains different datasets in worksheets starting with *Data*. You may assume that all datasets have a header in the first row (starting in cell A1). Each column contains either doubles or strings. If the header title of a column starts with *str* then the column contains strings; otherwise the column contains doubles.

A dataset can be stored in two arrays where one store the column headers and one the data. For instance, the following code in VBA read the data in worksheet *Data1* into two arrays:

```
Sub ReadData1()
   Dim aryData As Variant
   Dim aryHeaders() As String
   Dim rng As Range

   ThisWorkbook.Worksheets("Data1").Activate
   Set rng = Range("A1:" & RngGetCurRegionLastCol(Range("A1"), asLetter:=True) & "1")
   Call AryRead(aryHeaders, rng)
   Set rng = RngGetCurRegionRange(Range("A1"), row:=2, col:=1)
   Call AryRead(aryData, rng)
End Sub
```

The array storing the header titles contains strings. However, since we do not know the datatype of a column in the dataset, the array `aryData` is of type `Variant`. That is, we don't write `Dim aryData() As Variant` (which we normally do when defining a dynamic array), but `Dim aryData As Variant`, since a variant in itself can be an array.

Write a set of VBA procedures that answer/complete the following questions/tasks. All procedures should be coded in module *EM_Transform* and MUST be documented using a skeleton similar to the examples given in the course notes (Section D1).

Question 1

Given a header title, create a procedure `IsColumnString` with the following features:

- The procedure takes a header title (string) as argument.
- Returns true if the string starts with *str* and false otherwise.

That is, if call the procedure with string `str_desc` then the procedure would return true, while calling the procedure with string `item_id` would return false. Hint: The `LEFT` string function in VBA may be useful.

Question 2

Given the array with header titles, create a procedure `GetColumnIndex` that return the column index of a header title with the following features:

- The procedure takes an array as argument containing the header titles.
- The procedure takes a string as argument with the header title to search for.
- Returns the index of the header title (if found) else -1.

For instance, if the header titles are array `("Ship", "Profit", "Cost")` with index starting from 1. Then a call to the procedure searching for `Cost` would return 3 while searching for `Boat` would return -1.

Question 3

Create a procedure `SelectColumns` that selects a set of columns in a dataset stored using two arrays with the following features:

- The procedure takes the two arrays representing the dataset (header and data) as arguments.
- The procedure takes an array of strings as argument containing the header titles to select. If this array contains header titles not present in the dataset, then they are ignored.
- The modified dataset is output/pasted to the worksheet named *Select* (starting in cell A1), i.e. no new arrays are needed to store the new dataset. This worksheet is cleared if it contains old data.
- The procedure should work for different datasets stored in the arrays (e.g. the dataset in worksheet *Data2*).
- The procedure does not have any intermediate steps which use ranges to select columns.

Hint: The course procedure `AryPasteColumn` stored in module `ModAry` may be useful.

Test `SelectColumns` by writing a procedure `SelectTest` that:

- Loads the dataset in worksheet *Data1* in two arrays (a dataset of orders to a set of ships).
- Select columns *ship*, *price* and *item_id* (in that sequence).
- Create message box with the value of cell B2 in worksheet *Select* along with the column name.
- Given the original dataset, select columns *ship*, *boat* and *item_id* (in that sequence).
- Create message box with the value of cell B2 in worksheet *Select* along with the column name.

Add a button running the procedure to worksheet *Select*.

Question 4

Create a procedure `Filter` that filter some rows based on a criteria in a dataset stored using two arrays with the following features:

- The procedure takes the two arrays representing the dataset (header and data) as arguments.
- The procedure takes a string as argument containing the header title of the column to filter on.
- The procedure takes a string as argument containing the compare operator used. The valid compare operators are
  - `<=` : less than or equal (valid for a double column).
  - `>=` : greater than or equal (valid for a double column).
  - `=` : equal (valid for all column types).
  - `in` : contained in (valid for a string column).
- The procedure takes a variant as argument containing the value to compare with.
- The modified dataset is output to the worksheet named *Filter* (starting in cell A1), i.e. no new arrays are needed to store the new dataset. This worksheet is cleared if it contains old data.
- If the header title or the compare operator is not valid then nothing should be output to the *Filter* worksheet.
- The procedure should work for different datasets stored in the arrays.
- The procedure does not have any intermediate steps which use ranges to filter rows.

For instance, calling the procedure with header title `price`, compare operator `<=` and value `4`, will output the rows of the dataset where the values in the *price* column is less than or equal 4. Similar calling the procedure with header title `str_item_desc`, compare operator `in` and value `Gasket`, will return the rows of the dataset where the values in the *str_item_desc* column contain the string `Gasket`.

Hint: The course procedure `AryPasteRow` stored in module `ModAry` may be useful.

Test `Filter` by finishing the code of procedure `FilterTest` that:

- Loads the dataset in worksheet *Data1* in two arrays.
- Filter so get rows where price is greater than or equal 500000.
- Create message box with the value of cell G2 in worksheet *Filter* along with the column name.
- Given the original dataset, filter so get rows where the item description contains *Gasket*.
- Create message box with the value of cell B2 in worksheet *Filter* along with the column name.

Add a button running the procedure to worksheet *Filter*.

Question 5

The `Filter` procedure can only be used with a single compare operator. Consider the dataset in worksheet *Data1* and discuss how a sequence of calls to `Filter` can be used to find rows where

1) price is less than or equal to 400 AND supplier id equals 11,
2) price is less than or equal to 400 OR price is greater than or equal to 3000,
3) (price is less than or equal to 400 AND supplier id equals 11) OR item description contains *Gasket*.

Add your discussion as a VBA comment in module *EM_transform* (at most 10 sentences). Note you should not code this, just comment.

VBA - Assignment 2 (40%)

A company sells a product and the demand D of the product depends on the price p. The demand has been estimated to be $D = D_1 + D_2$, where D_i denote the demand from Customer Segment i. The demand from Customer Segment 1 is stochastic and follows a custom discrete distribution:

- $D_1 = \frac{625}{p} - 5$ with probability 0.25
- $D_1 = \frac{625}{p}$ with probability 0.5
- $D_1 = \frac{625}{p} + 5$ with probability 0.25

The demand from Customer Segment 2 is stochastic and follows a Possion distribution with parameter/mean equal to $8000/p^2$. Hint: The course procedure `RandInvPoisson` stored in module `ModRand` may be useful.

Write a set of VBA procedures that answer/complete the following questions/tasks. All procedures should be coded in module *EM_Demand* and MUST be documented using a skeleton similar to the examples given in the course notes (Section D1).

Question 1

Create two procedures `Demand1` and `Demand2` that return the stochastic demand D_1 and D_2 given a price. Next, create a procedure `Demand` that use the procedures `Demand1` and `Demand2` to calculate the stochastic demand.

Question 2

The company has a production capacity of 75 units. If the demand is above the production capacity then sales is 75 and the remaining demand is lost at a cost of l DKK per unit. The profit is given by the sales revenue minus the lost sales cost.

Create a procedure `Profit` that calculates the profit given price p and lost sales cost l with the following features:

- Stochastic demand are used inside the procedure and returned (`ByRef`)
- The number of sold units is returned (`ByRef`).
- The number of lost sales units is returned (`ByRef`).
- The profit is returned (`ByRef`).
- The default value of l is 10 DKK.

You may test `Profit` using procedure `ProfitTest`.

Question 3

The company consider setting the price p of the product in the range $5,\ldots,40$. Make a procedure `ProfitSim` that does a simulation with the following features:

- An input box is used to give the lost sales cost l.
- Given a price calculate measures: demand, number of sold units, number of lost sales units and the profit.
- For each price in the range do 500 runs and calculate the averages for each measure.
- Output the results in worksheet *Simulation* so one row is given for each price with the averages.

Add a button running the procedure to worksheet *Simulation*. Which price gives the best profit given $l = 15$ (add this as a comment inside the procedure)?