**2021 NJC CT**

**Task 1**

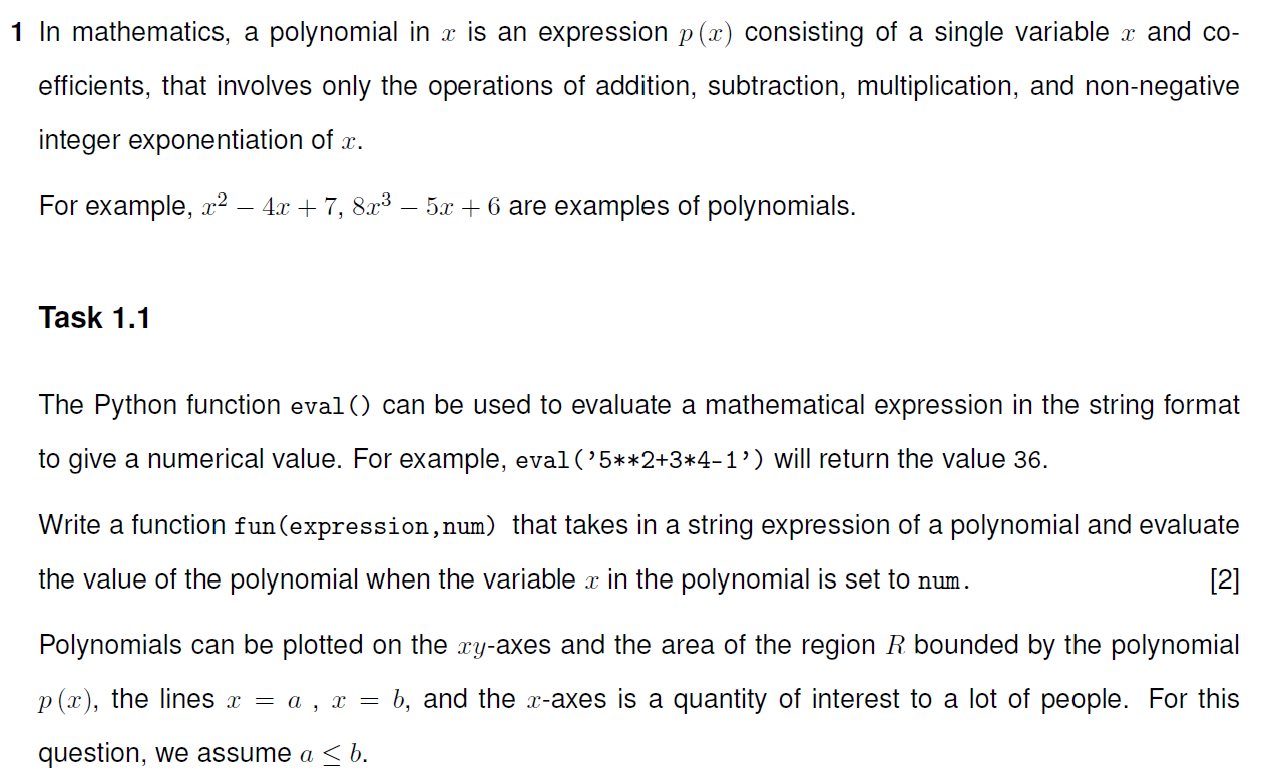
| A hierarchical file storage management is to be implented which allows files on a computer to be physically stored in 3 tier of storage:   * Tier-1 is an online hard disk drive attached to the computer. * Tier-2 is a online cloud storage provided by a cloud storage provider. * Tier-3 is an offline DVD storage.   A file listings of all the files stored on the computer is given in the file, FILE\_LIST.CSV.  Each line contains the **filename**, **time-stamp** and the **size** of the file in bytes.  The time-stamp is in the following format : "dd/mm/yy HH:MM" ,  where dd is the 2 digit day, mm is the 2 digit month, yy is the 2 digit year and HH:MM is the hour and minute in 24 hour format.  For each sub-task, add a comment statement, at the beginning of the code using the hash symbol "#", to indicate the sub-task the program code belongs to, for example:   | In [1]: | | *#Task 1.1*  Program code | | --- |   Output: | | --- | --- | --- |   The files provided are FILE\_LIST.CSV, TIER\_3\_UPDATE.CSV | |
| --- | --- | --- | --- | --- |
| **Task 1.1**  Write Python code to read the contents of the file FILE\_LIST.CSV and group the file listings into 3 **Python Lists**, tier-1, tier-2 and tier-3 for processing and then output into 3 files according to the following requirements:   | Group | Description | Output | | --- | --- | --- | | Tier 1 files | The last-modified time-stamps on these files must be less than 100 days from the current time-stamp. | The file listings in this group is to be sorted in descending order of the last-modifed time-stamp (latest modified files will appear first in the list) and saved in a file named TIER-1.CSV | | Tier 2 files | The last-modified time-stamps on these files must be greater or equal to 100 days but less than 365 days. | The file listings in this group is to be sorted in descending order of the last-modifed time-stamp (latest modified files will appear first in the list) and saved in a file named TIER-2.CSV | | Tier 3 files | The last-modified time-stamps on these files must be equal or greater then 365 days from the current time-stamp. | The file listings in this group is to be sorted in descending order of the last-modifed time-stamp (latest modified files will appear first in the list) and saved in a file named TIER-3.CSV |   The sorting of the file-listings must be done using a **quick-sort** algorithm. You **cannot** use any build-in sorting functions. The datetime string format should be in the form "dd/mm/yy HH:MM" , where dd is the 2 digit day, mm is the 2 digit month, yy is the 2 digit year and HH:MM is the hour and minute in 24 hour format.  EFor example the contents of file TIER-3.CSV should look like this:   | WMSysPr9.prx,07/12/19 17:53,316640  DtcInstall.log,07/12/19 17:17,776  Professional.xml,07/12/19 17:10,30831  :  :  pyshellext.amd64.dll,05/12/17 20:42,55456  system.ini,29/09/17 21:44,219  CoreSingleLanguage.xml,29/09/17 21:42,35138 | | --- | | [15] |
| **Task 1.2**  The file names in the Windows operating system have two parts; the file's name, then a period followed by the extension (suffix). The extension is an abbreviation that signifies the file type. Extensions are important because they tell your computer what icon to use for the file, and what application can open the file.  The follwing examples show the filenames and their extensions:   | File name | File extension | | --- | --- | | bootstat.dat | dat | | pyshellext.amd64.dll | dll | | Prog.is.fun.pdf | pdf | | ..txt | txt | | Modules | '' (empty string) | | . | '' (empty string) | | .. | '' (empty string) |   Implement the following function using Python code to extract the file extension from the given file name:  FUNCTION get\_extension(s: STRING)RETURNS STRING  The function must support the test cases in the example file names shown above. | [4] |
| **Task 1.3**  A MongoDB database named file\_stats is to be used to store statistics about the file system. A MongoDB collection named storage is to be created to store documents based on the storage tier level. An example of a document in the storage collection looks like this:  {"storage\_tier":1,  "file\_count":14,  "file\_names":["WindowsUpdate.log","bootstat.dat", "setupact.log","PFRO.log","explorer.exe","notepad.exe",  "HelpPane.exe","regedit.exe","bfsvc.exe","splwow64.exe", "diagerr.xml","diagwrn.xml","comsetup.log","setuperr.log"]  "file\_types":{"log": 5, "dat": 1, "exe": 6, "xml": 2}  }  The description of the attributes used in the document is as follow:   | **Attribute name** | **Value description** | | --- | --- | | storage\_tier | Storage tier number | | file\_count | Number of files in the storage tier | | file\_names | List of filenames in the storage tier | | file\_types | Document object containing the file extensions and the number of files with the file extension in the storage tier. Example in the document shown above, storage\_tier 1 has 5 file\_s with extension log | |  |
|  |  |
| Using the 3 files, TIER-1.CSV,TIER-2.CSV and TIER-3.CSV created in Task 1.1 and the function get\_extension implemented in Task 1.2 write Python code to :   * create the MongoDB database and collection. * create the documents based on the required statistics given above. | [5] |
| **Task 1.4**  Assuming that the file TIER\_3\_UPDATE.CSV is generated using the code implementation in Task 1.1 at a future date from the current date and time. It contains the file listings for Tier-3 storage.  Write Python code to update the MongoDB database that you have created in **Task 1.3**. You may re-use your code in Task 1.1 to Task 1.3. | [2] |
| Name and save your Jupyter Notebook as  Task1\_<your name>\_<NRIC number>.ipynb. | |

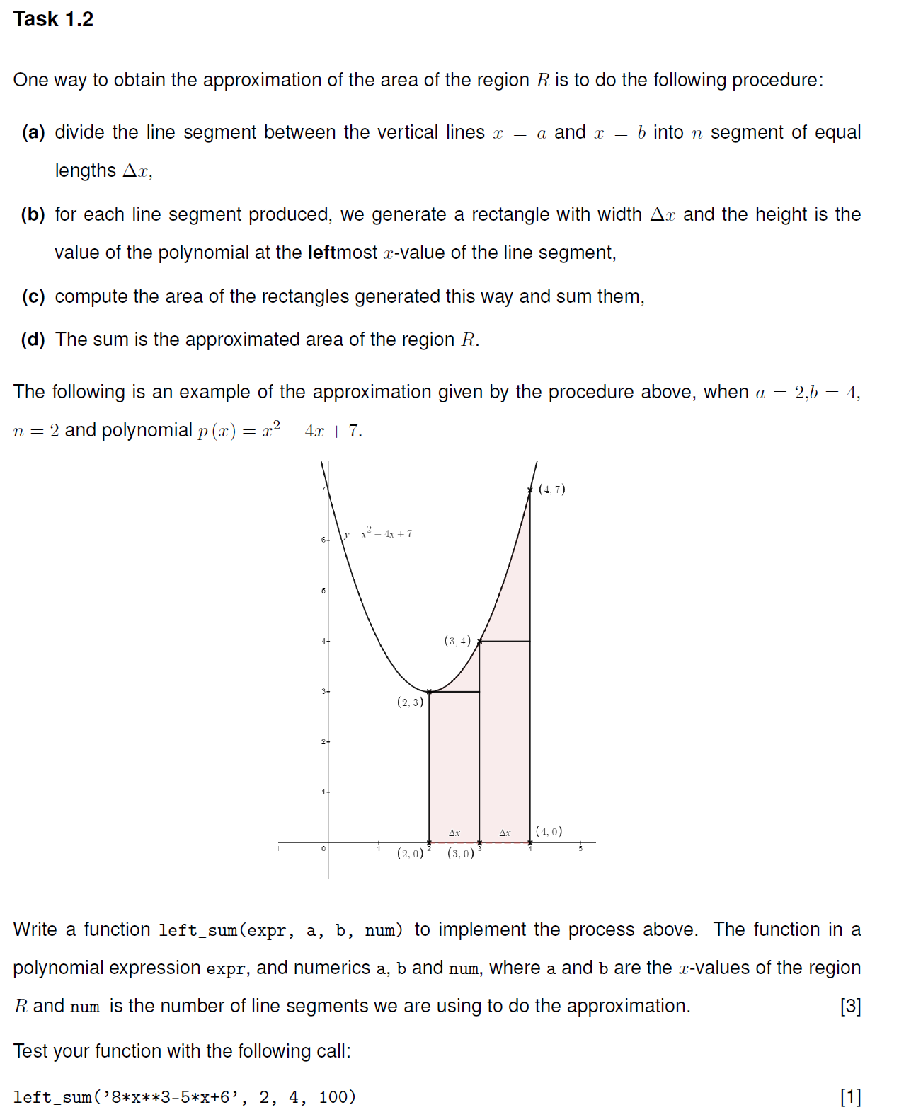
**2021 NJC Prelim**

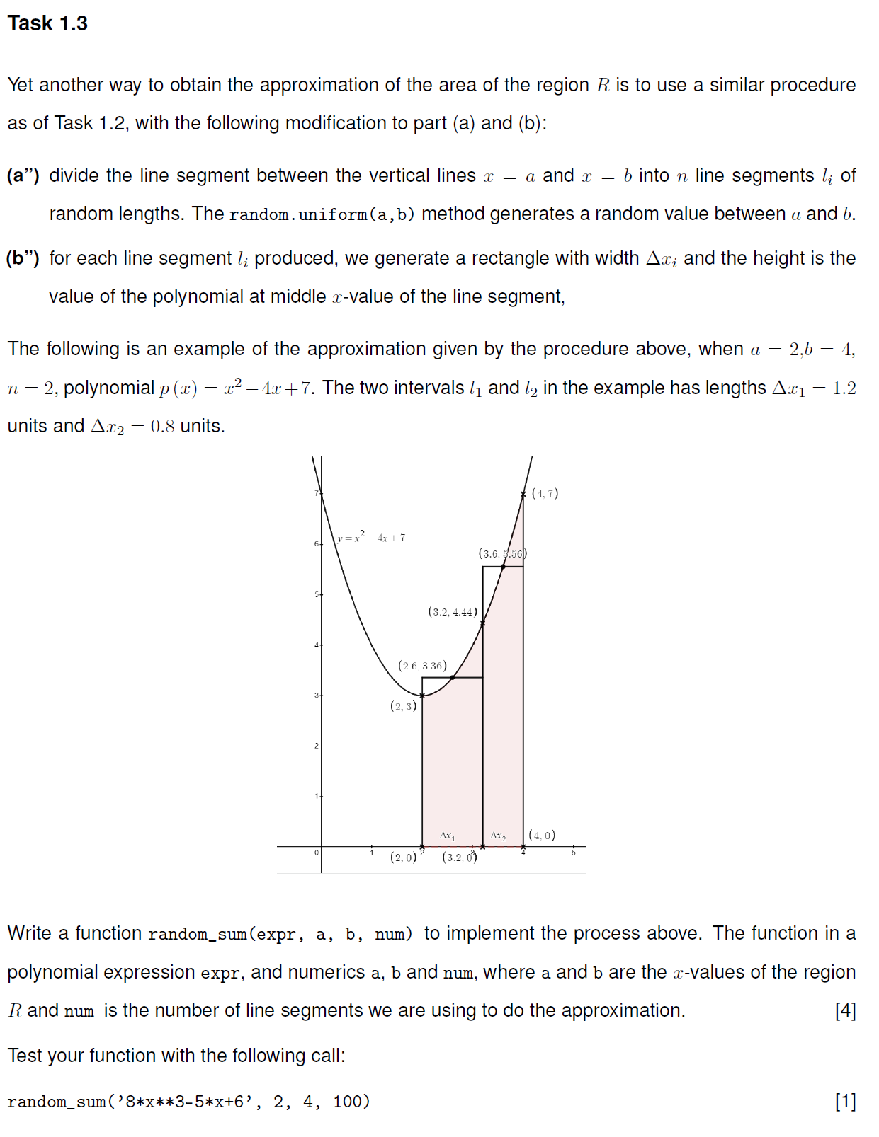
**Task 1**

| The file COVID19.CSV contains data of confirmed cases of COVID19 infections over a period of time.  Each row, except the header row, in the file records the cases for a particular province/state of a country **or** for the entire country.  The rows are not sorted in any order.  For each sub-task, add a comment statement, at the beginning of the code using the hash symbol "#", to indicate the sub-task the program code belongs to, for example:   | In [1]: | | *#Task 1.1*  Program code | | --- |   Output: | | --- | --- | --- |   The file provided is COVID19.CSV1 | |
| --- | --- | --- | --- | --- |
| **Task 1.1**  Write Python code to determine the following statistics from the file given:   * number of different countries in the data collection. * number of days that data were collected. * start date and end date of the data collection .   Output the 3 statistics above. | [3] |
| **Task 1.2**  Write Python code to aggregate the data for each Country/Region, for each of the day and write the new data in a file named countries.csv in the following format:  Country, <start\_date>,..., <end\_date>  where start\_date and end\_date are the start and end dates of the data collected. The file is sorted in **ascending** order of the Country/Region name.  The file contents should look like this:   | Country/Region,1/1/20,1/2/20,1/3/20,1/4/20,...  Australia,0,0,0,0,...  France,0,0,2,3,...  Mainland China,547,639,916,1399,...  : | | --- | | [10] |
| **Task 1.3**  Write Python code to determine the top five country/region with the  **largest** single day increase in confirmed cases and the date this occurred.  Example of output:  Mainland China 15133 2/13/20  South Korea 851 3/3/20  Iran 835 3/3/20  Italy 587 3/4/20  Others 99 2/17/20 | [5] |
| The contents of the file, countries.csv, created in **Task 1.2** is to be stored as documents in a collection named **cases**, on a MongoDB database named **covid**.  Each document must be store in the following structure:  {  "date": "1/22/20",  "countries":  [  {-  "country": "Afghanistan",  "cases": 0  },  {  "country": "Algeria",  "cases": 0  },  {  "country": "Andorra",  "cases": 0  },  {  "country": "Argentina",  "cases": 0  }  ]  } |  |
| **Task 1.4**  Write Python code to create the MongoDB database and collection and insert all the documents into the collection. | [5] |
| **Task 1.5**  Write Python code to connect to the MongoDB database created in **Task 1.4** and print the number of cases reported on 1/28/20 in Mainland China. | [2] |
|  |  |
| Name and save your Jupyter Notebook as  Task1\_<your name>\_<NRIC number>.ipynb. | |

**2022 NJC Prelim**







2022 A Level

