ntic master
revolucionamos la comunicación

UNIVERSIDAD COMPLUTENSE DE MADRID

Python. Trabajo final

Causas de muerte en el mundo

Autor: Cristóbal Pareja Flores

Actualizado: 27 de septiembre de 2020

NOTA PRELIMINAR

Antes de desarrollar el proyecto que se propongo a continuación, te recerdo la importancia de leer las instrucciones de entrega.

1. CAUSAS DE MUERTE EN EL MUNDO

La página https://ourworldindata.org/ ofrece un sinfín de datos abiertos sobre distintos aspectos problemas mundiales, como son la pobreza, las enfernedades, el hambre, el cambio climático, la guerra o la desigualdad. La simple inspección de estos datos nos lleva a una reflexión que siempre es útil y necesaria. Pero es posible ir más allá, examinando estos datos con rigor científico. De los muchos problemas a los que se enfrenta la humanidad, ahora nos centramos en las causas de muerte, y entrando en la página mencionada, .../causes-of-death, encontramos el gráfico siguiente:

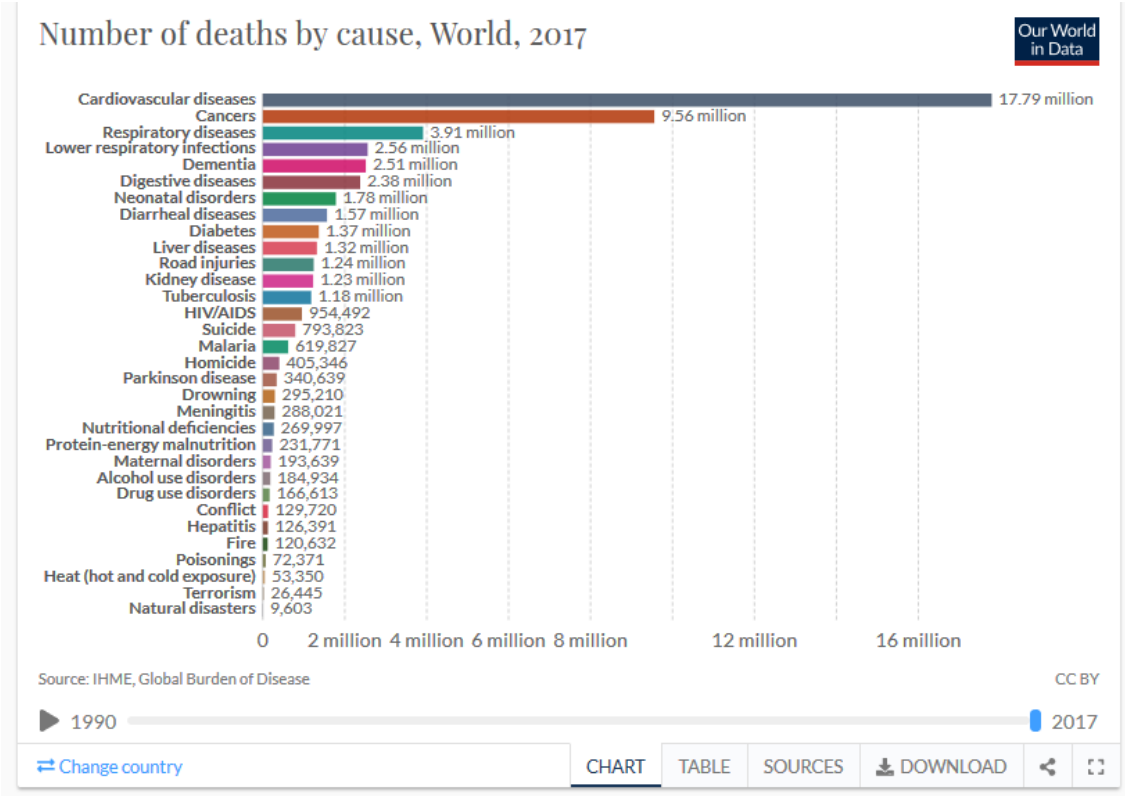
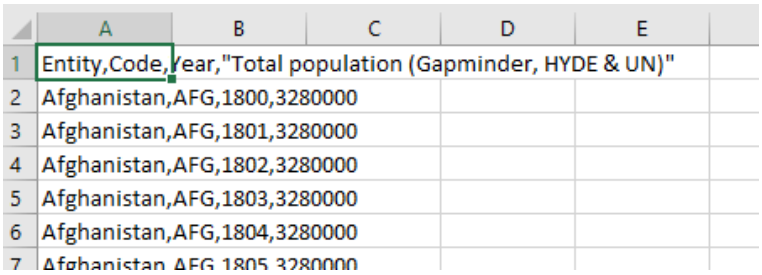


Figura 1. Número de muertes en el mundo en 2017, según su causa

Aunque deseamos ya inspeccionar los datos, sacar conclusiones, comparar... nos contenemos, vamos paso a paso, y empezamos por recoger los datos de los habitantes del planeta por países.

a) En las páginas de World in Dats puedes encontrar una tabla csv ('population.csv') sobre el crecimiento de la población mundial por países (Population growth by country). Algo así:



	A	B	C	D	E
1	Entity,Code,Year,"Total population (Gapminder, HYDE & UN)"				
2	Afghanistan,AFG,1800,3280000				
3	Afghanistan,AFG,1801,3280000				
4	Afghanistan,AFG,1802,3280000				
5	Afghanistan,AFG,1803,3280000				
6	Afghanistan,AFG,1804,3280000				
7	Afghanistan,AFG,1805,3280000				

Diseña una función que cargue dicha tabla y genere un diccionario, que para cada país y cada año, nos dé su población.

Ten cuidado, además de los países, esta tabla contiene información de regiones (Europa por ejemplo), con o sin código, y del mundo entero.

b) Como simples demostraciones, diseña expresiones o funciones que calculen lo siguiente:

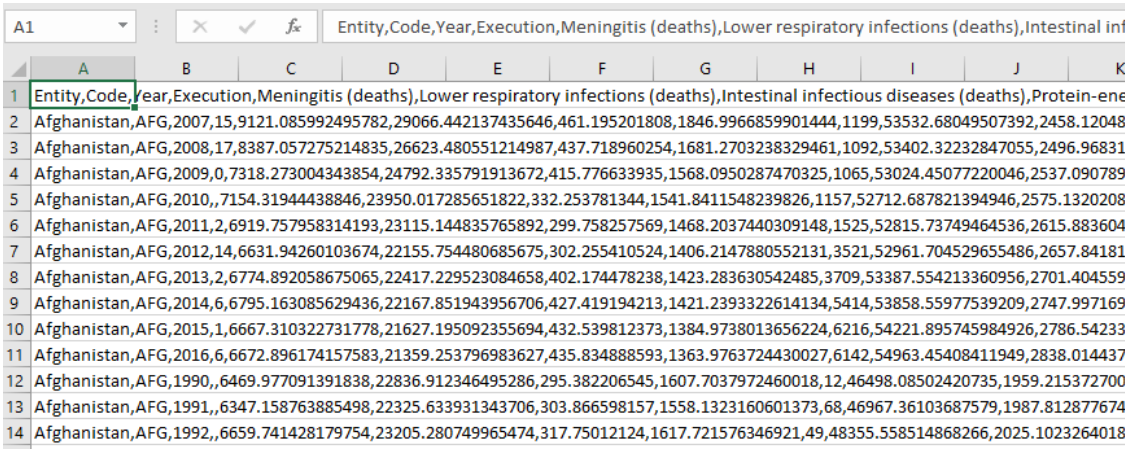
- La población española en 1800.
- La población mundial (El mundo tiene sus propios registros en la tabla csv bajo el nombre "World" y el código "OWID_WRL") en los años 10.000, 9.000, 8.000, …, 1.000 antes de la era común.
- El incremento de la población en Europa entre dos años dados. Esto se ha de resolver con una función cuyos requisitos indiquen los años posibles de uso.

c) Diseña ahora una función que genere una tabla de datos con las poblaciones de los distintos países entre dos años dados como parámetros (por ejemplo, entre 2007 y 2016). Esta tabla puede ser una simple lista de listas, o un array (de numpy) o un dataframe (de pandas) o incluso un diccionario.

d) Sería ideal mostrar la evolución de la población mundial entre dos fechas dadas, de la era común, y compararla con la de un país (otro parámetro). Para que los gráficos sean apreciables, convendría cambiar la escala del país, y lógicamente esa escala dependerá de la población del país, de manera que la función correspondiente tendrá que calcular esa escala con algún criterio razonable.

Vamos ahora con la mortalidad. La información mostrada en la Figura 1 se pueden ver también en forma de tabla seleccionando el año (desde 2007

hasta 2016), y lo que es más útil, estos datos también están disponibles al completo en una tabla csv ('annual-number-of-deaths-by-cause.csv') que, para cada país y año, contiene el número de muertes por distintas causas:



	A	B	C	D	E	F	G	H	I	J	K
1	Entity,Code,Year,Execution,Meningitis (deaths),Lower respiratory infections (deaths),Intestinal infectious diseases (deaths),Protein-ene										
2	Afghanistan,AFG,2007,15,9121.085992495782,29066.442137435646,461.195201808,1846.9966859901444,1199,53532.68049507392,2458.12048										
3	Afghanistan,AFG,2008,17,8387.057275214835,26623.480551214987,437.718960254,1681.2703238329461,1092,53402.32232847055,2496.96831										
4	Afghanistan,AFG,2009,0,7318.273004343854,24792.335791913672,415.776633935,1568.0950287470325,1065,53024.45077220046,2537.090789										
5	Afghanistan,AFG,2010,,7154.31944438846,23950.017285651822,332.253781344,1541.8411548239826,1157,52712.687821394946,2575.1320208										
6	Afghanistan,AFG,2011,2,6919.757958314193,23115.144835765892,299.758257569,1468.2037440309148,1525,52815.73749464536,2615.883604										
7	Afghanistan,AFG,2012,14,6631.94260103674,22155.754480685675,302.255410524,1406.2147880552131,3521,52961.704529655486,2657.84181										
8	Afghanistan,AFG,2013,2,6774.892058675065,22417.229523084658,402.174478238,1423.283630542485,3709,53387.554213360956,2701.404559										
9	Afghanistan,AFG,2014,6,6795.163085629436,22167.851943956706,427.419194213,1421.2393322614134,5414,53858.55977539209,2747.997169										
10	Afghanistan,AFG,2015,1,6667.310322731778,21627.195092355694,432.539812373,1384.9738013656224,6216,54221.895745984926,2786.54233										
11	Afghanistan,AFG,2016,6,6672.896174157583,21359.253796983627,435.834888593,1363.9763724430027,6142,54963.45408411949,2838.014437										
12	Afghanistan,AFG,1990,,6469.977091391838,22836.912346495286,295.382206545,1607.7037972460018,12,46498.08502420735,1959.215372700										
13	Afghanistan,AFG,1991,,6347.158763885498,22325.633931343706,303.866598157,1558.1323160601373,68,46967.36103687579,1987.812877674										
14	Afghanistan,AFG,1992,,6659.741428179754,23205.280749965474,317.75012124,1617.721576346921,49,48355.558514868266,2025.1023264018										

e) Prepara una función que cargue los datos en una tabla, en el formato que te parezca más adecuado, teniendo en cuenta su utilidad en los apartados siguientes. Por unos pocos ejemplos de su uso similares a los planteados en el apartado b).

f) Diseña una función que, para una causa de muerte ("Road injuries" por ejemplo), calcule las tasas de muerte en un país ("Spain" por ejemplo) y año (2015 por ejemplo), donde una tasa está calculada como el número de muertes por esa causa en ese país y año por cada 10000 habitantes.

g) Algunas de las funciones anteriores o pequeñas variantes suyas nos resultarán útiles para generar un gráfico adecuado que nos permita comparar la mortalidad por una causa en un año, en una colección de países.

h) Desearíamos saber el total de muertes en un año dado por cada causa. Esto se puede calcular de varias maneras. Te pido ahora que resuelvas este problema con un programa iterativo, que recorre países y va acumulando en un diccionario las cantidades de cada causa.

i) Te pido ahora que resuelvas el ejercicio anterior con la técnica map-reduce.

j) Diseña una colección de funciones que trabajan sucesivamente, cargando los datos, mostrando una parte pequeña de los mismos y, finalmente, generando una tabla como la de la Figura 1.

Ejercicio complementario, de carácter optativo

En un gráfico como el anterior, los datos de mayor orden no permiten ver bien los datos de órdenes más pequeños. Por eso es mejor usar una tabla logarítmica (o más precisamente logarítmica-lineal). En ella, la escala en el eje horizontal no crece linealmente (2 millones, 4 millones, 6, 8, 10, etc.), sino exponencialmente ($10^3, 10^4, 10^5, 10^6$, etc.), lo que equivale visualmente a suavizar mucho las barras de órdenes mayores logarítmicamente.

k) Te pido hacer esto, obteniendo un gráfico similar al presentado, pero:

k.1) Versión básica: con los datos mundiales, para el año elegido (un parámetro)

k.2) Vesión intermedia: donde el país elegido sea también un parámetro

k.3) Versión más ambiciosa: comparando la tasa de los datos mundiales con la de un país (cantidades por millón de habitantes mundial y del país, en el año elegido).

Elige la versión más ambiciosa que puedas realizar.

APÉNDICE A. DESCRIPCIÓN DE LA ENTREGA

La entrega consistirá en una carpeta comprimida, identificada con los apellidos y nombre del estudiante (por ejemplo, “ParejaFloresCristobal”) y sin tildes ni eñes. En ella una carpeta por cada problema planteado, que contendrá los siguientes archivos:

- Un archivo ipynb con la solución de los apartados resueltos, explicados adecuadamente, con excepción del apartado10 (map-reduce). Cada apartado tendrá su rótulo, en modo markdown, con el enunciado, una explicación de lo logrado y de lo no logrado, y unos pocos ejemplares de prueba bien elegidos que muestren el funcionamiento de dicho apartado. Algún apartado se puede resolver en varias funciones, y será necesario ilustrar el funcionamiento de cada una de ellas con tests apropiados.

  Este archivo se entregará también en formato pdf. En el archivo ipynb (y en el pdf correspondiente) se han de poder ver las comprobaciones parciales de cada uno de los apartados.

- Para el problema de map-reduce, el programa o los programas necesarios en Python, ejecutables desde la línea de comandos, que implementan las tareas realizadas.

- Archivos de datos completos, de manera que permitan ilustrar todas las características de las soluciones aportadas.

- Un archivo llamado “instrucciones_de_uso.{txt, docx, pdf}” que describa la invocación que se debe hacer, desde la línea de comandos, de cada uno de los programas aportados fuera del archivo ipynb, indicando las librerías que será necesario tener instaladas para su correcta ejecución. En todo caso, el uso incorrecto de un programa deberá disparar el correspondiente aviso con las instrucciones necesarias para su correcto funcionamiento.

- Un pequeño archivo llamado “documentaciónProyecto.{docx, pdf, txt}” que contenga la información siguiente:

  - Si procede, las decisiones de diseño generales o comentarios adicionales sobre el trabajo entregado.

  - La ayuda recibida o fragmentos de código tomados de otro estudiante o de cualquier otra fuente.

  - Una pequeña ficha de autoevaluación, es decir, el nombre y apellidos del autor y la relación de partes de la práctica logradas, junto con la nota estimada por el estudiante en cada una de dichas partes o apartados.

  Esta parte se puede incluir en un apartado al final del archivo de Jupyter con las soluciones propuestas.

APÉNDICE B. COMENTARIOS ADICIONALES

- No se deberá entregar ningún programa que no esté debidamente comprobado y sea correcto.

- Cada programa estará debidamente organizado y documentado. Cada función incluida en los programas entregados deberá ser clara, estar bien organizada y estar debidamente documentada, siguiendo normas estándar. Este asunto es de gran importancia y se tendrá en cuenta fuertemente en la evaluación.

- Se sobreentiende que cada entrega está hecha exclusivamente por el autor, considerándose inaceptable entregar cualquier trabajo realizado total o parcialmente por otra persona distinta del autor firmante, o un trabajo en que se ha copiado una parte del código propuesto por otros compañeros o de cualquier otra fuente. En todo caso, se ha de consignar con total claridad la ayuda recibida o tomada prestada, por cortesía y por honradez.

APÉNDICE C. ALGUNOS ERRORES FRECUENTES

- Organizar un programa sin usar una estructura adecuada en funciones.

- Diseñar funciones sin parámetros; diseñar funciones sin documentar; emplear identificadores inadecuados; emplear variables globales; definir funciones con constantes literales que registran valores propios del programa y se repiten en distintos lugares del mismo, tales como nombres de archivos, fechas, tamaños de muestras, etc.

- Las funciones necesarias para que nuestra aplicación trabaje no deben contener lecturas de datos (que deben proporcionarse normalmente a través de los parámetros) ni salidas (esto irá normalmente en pequeños subprogramas de demostración). Se ha de separar el funcionamiento de nuestra aplicación de sus demostraciones de funcionamiento.

- Recorrer listas a través de su índice cuando no es necesario, sin explotar la potencia de Python.

- Usar rutas absolutas: f = open('/Users/blacky/Desktop/Proyectos.txt', 'w')

  Esto impide ejecutar un programa fuera del ordenador en que se ha diseñado.

  Lo correcto es usar rutas relativas: f = open('./Proyectos.txt', 'r')

  Para comprobar que lo has hecho bien, cambia la carpeta de tu programa de ubicación dentro de tu ordenador: debería funcionar bien en este nuevo lugar.

- Usar archivos csv (o xlsx) para almacenar información textual plana. Cuando se genera un archivo csv, comprueba que cada fila está organizada en celdas distintas, y no todas las componentes en la primera celda, sin separador alguno.

- La presentación de un proyecto en un archivo ipynb permite ir diseñando pequeñas piezas de código, ir explicando sus porqués e ir mostrando el funcionamiento de cada una. No presentes definiciones todas de golpe, y no olvides ir mostrando el funcionamiento de cada función o cada pequeña pieza de código nueva.

- Un error frecuente que cometen, sobre todo, programadores de otros lenguajes, es programar en Python como si fuera Java (o C++, etc.). Familiarízate con los conceptos nuevos y úsalos adecuadamente: funciones que devuelven varios valores a la vez en una tupla, funciones de orden superior, listas intensionales, conjuntos, diccionarios, etc. Son recursos que puedes y debes usar adecuadamente.