

14/09/2021

PROYECTO-01- CARRILLO-ÁNGEL


ÁNGEL HUMBERTO CARRILLO CERVERA





Índice

Introducción	0
Definición de Código.....	1
Librerías utilizadas para el proyecto.....	1
Módulo de bienvenida al Sistema.....	22
Aplicación de bucle While para login.....	23
Módulo inicial del Menú al Sistema.....	23
Sección 1 – Resultados de productos más vendidos y rezagados	23
Sección 2 – Productos por reseña	26
Sección 3 – Total de ingresos y ventas (Estrategia)	27
Opción de cierre del menú de Sistema	28
Módulo de registro del Sistema.....	28
Módulo en caso de opción incorrecta del Sistema.....	29
Solución al Problema.....	29
Conclusión	29



Introducción

El presente proyecto se hace con el fin de cubrir los requerimientos para completar satisfactoriamente el curso de “Introducción a Python” impartido por EMTECH en colaboración con Santander Becas.

Objetivo

Poner en práctica las bases de programación en Python para análisis y clasificación de datos mediante la creación de programas de entrada de usuario y validaciones, uso y definición de variables y listas, operadores lógicos y condicionales para la clasificación de información.

Descripción del caso

LifeStore es una tienda virtual que maneja una amplia gama de artículos, recientemente, la Gerencia de ventas, se percató que la empresa tiene una importante acumulación de inventario. Asimismo, se ha identificado una reducción en las búsquedas de un grupo importante de productos, lo que ha redundado en una disminución sustancial de sus ventas del último trimestre.

Consigna

Derivado de la situación, la Gerencia de Ventas te solicita que realices un análisis de la rotación de productos identificando los siguientes elementos:

- 1) Productos más vendidos y productos rezagados a partir del análisis de las categorías con menores ventas y categorías con menores búsquedas.
- 2) Productos por reseña en el servicio a partir del análisis de categorías con mayores ventas y categorías con mayores búsquedas.
- 3) Sugerir una estrategia de productos a retirar del mercado así como sugerencia de cómo reducir la acumulación de inventario considerando los datos de ingresos y ventas mensuales.

Definición de Código

Librerías utilizadas para el proyecto

```
"""
This is the LifeStore_SalesList data:

lifestore_searches = [id_search, id product]
lifestore_sales = [id_sale, id_product, score (from 1 to 5), date, refund (1 for true or 0 to
false)]
lifestore_products = [id_product, name, price, category, stock]
"""

lifestore_products = [
    [1, 'Procesador AMD Ryzen 3 3300X S-AM4, 3.80GHz, Quad-Core, 16MB L2 Cache', 3019,
    'procesadores', 16],
    [2, 'Procesador AMD Ryzen 5 3600, S-AM4, 3.60GHz, 32MB L3 Cache, con Disipador Wraith Stealth',
    4209, 'procesadores', 182],
    [3, 'Procesador AMD Ryzen 5 2600, S-AM4, 3.40GHz, Six-Core, 16MB L3 Cache, con Disipador Wraith
    Stealth', 3089, 'procesadores', 987],
    [4, 'Procesador AMD Ryzen 3 3200G con Gráficos Radeon Vega 8, S-AM4, 3.60GHz, Quad-Core, 4MB L3,
    con Disipador Wraith Spire', 2209, 'procesadores', 295],
    [5, 'Procesador Intel Core i3-9100F, S-1151, 3.60GHz, Quad-Core, 6MB Cache (9na. Generación -
    Coffee Lake)', 1779, 'procesadores', 130],
    [6, 'Procesador Intel Core i9-9900K, S-1151, 3.60GHz, 8-Core, 16MB Smart Cache (9na. Generación
    Coffee Lake)', 11809, 'procesadores', 54],
    [7, 'Procesador Intel Core i7-9700K, S-1151, 3.60GHz, 8-Core, 12MB Smart Cache (9na. Generación
    Coffee Lake)', 8559, 'procesadores', 114],
    [8, 'Procesador Intel Core i5-9600K, S-1151, 3.70GHz, Six-Core, 9MB Smart Cache (9na.
    Generación - Coffee Lake)', 5399, 'procesadores', 8],
    [9, 'Procesador Intel Core i3-8100, S-1151, 3.60GHz, Quad-Core, 6MB Smart Cache (8va. Generación
    - Coffee Lake)', 2549, 'procesadores', 35],
    [10, 'MSI GeForce 210, 1GB GDDR3, DVI, VGA, HDCP, PCI Express 2.0', 889, 'tarjetas de video',
    13],
    [11, 'Tarjeta de Video ASUS AMD Radeon RX 570, 4GB 256-bit GDDR5, PCI Express 3.0', 7399,
    'tarjetas de video', 2],
    [12, 'Tarjeta de Video ASUS NVIDIA GeForce GTX 1660 SUPER EVO OC, 6GB 192-bit GDDR6, PCI Express
    x16 3.0', 6619, 'tarjetas de video', 0],
    [13, 'Tarjeta de Video Asus NVIDIA GeForce GTX 1050 Ti Phoenix, 4GB 128-bit GDDR5, PCI Express
    3.0', 3989, 'tarjetas de video', 1],
    [14, 'Tarjeta de Video EVGA NVIDIA GeForce GT 710, 2GB 64-bit GDDR3, PCI Express 2.0', 1439,
    'tarjetas de video', 36],
    [15, 'Tarjeta de Video EVGA NVIDIA GeForce GTX 1660 Ti SC Ultra Gaming, 6GB 192-bit GDDR6, PCI
    3.0', 8439, 'tarjetas de video', 15],
    [16, 'Tarjeta de Video EVGA NVIDIA GeForce RTX 2060 SC ULTRA Gaming, 6GB 192-bit GDDR6, PCI
    Express 3.0', 9799, 'tarjetas de video', 10],
    [17, 'Tarjeta de Video Gigabyte AMD Radeon R7 370 OC, 2GB 256-bit GDDR5, PCI Express 3.0', 4199,
    'tarjetas de video', 1],
    [18, 'Tarjeta de Video Gigabyte NVIDIA GeForce GT 1030, 2GB 64-bit GDDR5, PCI Express x16 3.0',
    2199, 'tarjetas de video', 5],
    [19, 'Tarjeta de Video Gigabyte NVIDIA GeForce GTX 1650 OC Low Profile, 4GB 128-bit GDDR5, PCI
    Express 3.0 x16', 4509, 'tarjetas de video', 8],
    [20, 'Tarjeta de Video Gigabyte NVIDIA GeForce RTX 2060 SUPER WINDFORCE OC, 8 GB 256 bit GDDR6,
    PCI Express x16 3.0', 11509, 'tarjetas de video', 10],
    [21, 'Tarjeta de Video MSI AMD Mech Radeon RX 5500 XT MECH Gaming OC, 8GB 128-bit GDDR6, PCI
    Express 4.0', 5159, 'tarjetas de video', 0],
    [22, 'Tarjeta de Video MSI NVIDIA GeForce GTX 1050 Ti OC, 4GB 128-bit GDDR5, PCI Express x16
    3.0', 3429, 'tarjetas de video', 0],
    [23, 'Tarjeta de Video MSI Radeon X1550, 128MB 64 bit GDDR2, PCI Express x16', 909, 'tarjetas de
    video', 10],
    [24, 'Tarjeta de Video PNY NVIDIA GeForce RTX 2080, 8GB 256-bit GDDR6, PCI Express 3.0', 30449,
    'tarjetas de video', 2],
    [25, 'Tarjeta de Video Sapphire AMD Pulse Radeon RX 5500 XT Gaming, 8GB 128-bit GDDR6, PCI
    Express 4.0', 5529, 'tarjetas de video', 10],
    [26, 'Tarjeta de Video VisionTek AMD Radeon HD 5450, 1GB DDR3, PCI Express x16 2.1', 1249,
    'tarjetas de video', 180],
    [27, 'Tarjeta de Video VisionTek AMD Radeon HD5450, 2GB GDDR3, PCI Express x16', 2109, 'tarjetas
    de video', 43],
```

[28, 'Tarjeta de Video Zotac NVIDIA GeForce GTX 1660 Ti, 6GB 192-bit GDDR6, PCI Express x16 3.0', 9579, 'tarjetas de video', 3],

[29, 'Tarjeta Madre ASUS micro ATX TUF B450M-PLUS GAMING, S-AM4, AMD B450, HDMI, 64GB DDR4 para AMD', 2499, 'tarjetas madre', 10],

[30, 'Tarjeta Madre AORUS ATX Z390 ELITE, S-1151, Intel Z390, HDMI, 64GB DDR4 para Intel', 4029, 'tarjetas madre', 50],

[31, 'Tarjeta Madre AORUS micro ATX B450 AORUS M (rev. 1.0), S-AM4, AMD B450, HDMI, 64GB DDR4 para AMD', 2229, 'tarjetas madre', 120],

[32, 'Tarjeta Madre ASRock Z390 Phantom Gaming 4, S-1151, Intel Z390, HDMI, 64GB DDR4 para Intel\xa0', 4309, 'tarjetas madre', 10],

[33, 'Tarjeta Madre ASUS ATX PRIME Z390-A, S-1151, Intel Z390, HDMI, 64GB DDR4 para Intel\xa0', 4269, 'tarjetas madre', 43],

[34, 'Tarjeta Madre ASUS ATX ROG STRIX B550-F GAMING WI-FI, S-AM4, AMD B550, HDMI, max. 128GB DDR4 para AMD', 5289, 'tarjetas madre', 2],

[35, 'Tarjeta Madre Gigabyte micro ATX Z390 M GAMING, S-1151, Intel Z390, HDMI, 64GB DDR4 para Intel\xa0', 3419, 'tarjetas madre', 30],

[36, 'Tarjeta Madre Gigabyte micro ATX Z490M GAMING X (rev. 1.0), Intel Z490, HDMI, 128GB DDR4 para Intel', 4159, 'tarjetas madre', 10],

[37, 'Tarjeta Madre ASRock ATX Z490 STEEL LEGEND, S-1200, Intel Z490, HDMI, 128GB DDR4 para Intel', 4289, 'tarjetas madre', 60],

[38, 'Tarjeta Madre Gigabyte Micro ATX H310M DS2 2.0, S-1151, Intel H310, 32GB DDR4 para Intel\xa0', 1369, 'tarjetas madre', 15],

[39, 'ASUS T. Madre uATX M4A88T-M, S-AM3, DDR3 para Phenom II/Athlon II/Sempron 100', 2169, 'tarjetas madre', 98],

[40, 'Tarjeta Madre Gigabyte XL-ATX TRX40 Designare, S-STRX4, AMD TRX40, 256GB DDR4 para AMD', 17439, 'tarjetas madre', 1],

[41, 'Tarjeta Madre ASUS micro ATX Prime H370M-Plus/CSM, S-1151, Intel H370, HDMI, 64GB DDR4 para Intel', 3329, 'tarjetas madre', 286],

[42, 'Tarjeta Madre ASRock Micro ATX B450M Steel Legend, S-AM4, AMD B450, HDMI, 64GB DDR4 para AMD', 1779, 'tarjetas madre', 0],

[43, 'Tarjeta Madre ASUS ATX ROG STRIX Z390-E GAMING, S-1151, Intel Z390, HDMI, 64GB DDR4 para Intel', 6369, 'tarjetas madre', 5],

[44, 'Tarjeta Madre MSI ATX B450 TOMAHAWK MAX, S-AM4, AMD B450, 64GB DDR4 para AMD', 2759, 'tarjetas madre', 0],

[45, 'Tarjeta Madre ASRock ATX H110 Pro BTC+, S-1151, Intel H110, 32GB DDR4, para Intel', 2869, 'tarjetas madre', 25],

[46, 'Tarjeta Madre Gigabyte micro ATX GA-H110M-DS2, S-1151, Intel H110, 32GB DDR4 para Intel', 1539, 'tarjetas madre', 49],

[47, 'SSD XPG SX8200 Pro, 256GB, PCI Express, M.2', 1209, 'discos duros', 8],

[48, 'SSD Kingston A2000 NVMe, 1TB, PCI Express 3.0, M2', 2559, 'discos duros', 50],

[49, 'Kit SSD Kingston KC600, 1TB, SATA III, 2.5, 7mm', 3139, 'discos duros', 3],

[50, 'SSD Crucial MX500, 1TB, SATA III, M.2', 2949, 'discos duros', 4],

[51, 'SSD Kingston UV500, 480GB, SATA III, mSATA', 2399, 'discos duros', 0],

[52, 'SSD Western Digital WD Blue 3D NAND, 2TB, M.2', 5659, 'discos duros', 13],

[53, 'SSD Addlink Technology S70, 512GB, PCI Express 3.0, M.2', 2039, 'discos duros', 1],

[54, 'SSD Kingston A400, 120GB, SATA III, 2.5'', 7mm', 259, 'discos duros', 300],

[55, 'SSD para Servidor Supermicro SSD-DM128-SMCMVN1, 128GB, SATA III, mSATA, 6Gbit/s', 4399, 'discos duros', 10],

[56, 'SSD para Servidor Lenovo Thinksystem S4500, 480GB, SATA III, 3.5'', 7mm', 3269, 'discos duros', 3],

[57, 'SSD Adata Ultimate SU800, 256GB, SATA III, 2.5'', 7mm', 889, 'discos duros', 15],

[58, 'SSD para Servidor Lenovo Thinksystem S4510, 480GB, SATA III, 2.5'', 7mm', 3679, 'discos duros', 16],

[59, 'SSD Samsung 860 EVO, 1TB, SATA III, M.2', 5539, 'discos duros', 10],

[60, 'Kit Memoria RAM Corsair Dominator Platinum DDR4, 3200MHz, 16GB (2x 8GB), Non-ECC, CL16, XMP', 2519, 'memorias usb', 10],

[61, 'Kit Memoria RAM Corsair Vengeance LPX DDR4, 2400MHz, 32GB, Non-ECC, CL16', 5209, 'memorias usb', 5],

[62, 'Makena Smart TV LED 32S2 32'', HD, Widescreen, Gris', 2899, 'pantallas', 6],

[63, 'Seiki TV LED SC-39HS950N 38.5, HD, Widescreen, Negro', 3369, 'pantallas', 146],

[64, 'Samsung TV LED LH43QMREBGCXGO 43, 4K Ultra HD, Widescreen, Negro', 12029, 'pantallas', 71],

[65, 'Samsung Smart TV LED UN70RU7100FXZX 70, 4K Ultra HD, Widescreen, Negro', 21079, 'pantallas', 7],

[66, 'TCL Smart TV LED 55S425 54.6, 4K Ultra HD, Widescreen, Negro', 8049, 'pantallas', 188],

[67, 'TV Monitor LED 24TL520S-PU 24, HD, Widescreen, HDMI, Negro', 3229, 'pantallas', 411],

[68, 'Makena Smart TV LED 40S2 40'', Full HD, Widescreen, Negro', 4229, 'pantallas', 239],

[69, 'Hisense Smart TV LED 40H5500F 39.5, Full HD, Widescreen, Negro', 5359, 'pantallas', 94],

[70, 'Samsung Smart TV LED 43, Full HD, Widescreen, Negro', 7679, 'pantallas', 10],

```

[71, 'Samsung Smart TV LED UN32J4290AF 32, HD, Widescreen, Negro', 4829, 'pantallas', 3],
[72, 'Hisense Smart TV LED 50H8F 49.5, 4K Ultra HD, Widescreen, Negro', 9759, 'pantallas', 11],
[73, 'Samsung Smart TV LED UN55TU7000FXZX 55, 4K Ultra HD, Widescreen, Negro/Gris', 10559,
'pantallas', 4],
[74, 'Logitech Bocinas para Computadora con Subwoofer G560, Bluetooth, Inalámbrico, 2.1, 120W
RMS, USB, negro', 4239, 'bocinas', 1],
[75, 'Lenovo Barra de Sonido, Alámbrico, 2.5W, USB, Negro', 441, 'bocinas', 11],
[76, 'Acteck Bocina con Subwoofer AXF-290, Bluetooth, Inalámbrico, 2.1, 18W RMS, 180W PMPO, USB,
Negro', 589, 'bocinas', 18],
[77, 'Verbatim Bocina Portátil Mini, Bluetooth, Inalámbrico, 3W RMS, USB, Blanco', 178,
'bocinas', 1],
[78, 'Ghia Bocina Portátil BX300, Bluetooth, Inalámbrico, 40W RMS, USB, Rojo - Resistente al
Agua', 769, 'bocinas', 2],
[79, 'Naceb Bocina Portátil NA-0301, Bluetooth, Inalámbrico, USB 2.0, Rojo', 709, 'bocinas',
31],
[80, 'Ghia Bocina Portátil BX800, Bluetooth, Inalámbrico, 2.1 Canales, 31W, USB, Negro', 1359,
'bocinas', 15],
[81, 'Ghia Bocina Portátil BX900, Bluetooth, Inalámbrico, 2.1 Canales, 34W, USB, Negro -
Resistente al Agua', 1169, 'bocinas', 20],
[82, 'Ghia Bocina Portátil BX400, Bluetooth, Inalámbrico, 8W RMS, USB, Negro', 549, 'bocinas',
31],
[83, 'Ghia Bocina Portátil BX500, Bluetooth, Inalámbrico, 10W RMS, USB, Gris', 499, 'bocinas',
16],
[84, 'Logitech Audífonos Gamer G332, Alámbrico, 2 Metros, 3.5mm, Negro/Rojo', 1089, 'audifonos',
83],
[85, 'Logitech Audífonos Gamer G635 7.1, Alámbrico, 1.5 Metros, 3.5mm, Negro/Azul', 2159,
'audifonos', 39],
[86, 'ASUS Audífonos Gamer ROG Theta 7.1, Alámbrico, USB C, Negro', 8359, 'audifonos', 20],
[87, 'Acer Audífonos Gamer Galea 300, Alámbrico, 3.5mm, Negro', 1719, 'audifonos', 8],
[88, 'Audífonos Gamer Balam Rush Orphix RGB 7.1, Alámbrico, USB, Negro', 909, 'audifonos', 15],
[89, 'Cougar Audífonos Gamer Phontum Essential, Alámbrico, 1.9 Metros, 3.5mm, Negro.', 859,
'audifonos', 4],
[90, 'Energy Sistem Audífonos con Micrófono Headphones 1, Bluetooth, Inalámbrico, Negro/Grafito',
539, 'audifonos', 1],
[91, 'Genius GHP-400S Audífonos, Alámbrico, 1.5 Metros, Rosa', 137, 'audifonos', 16],
[92, 'Getttech Audífonos con Micrófono Sonority, Alámbrico, 1.2 Metros, 3.5mm, Negro/Rosa', 149,
'audifonos', 232],
[93, 'Ginga Audífonos con Micrófono GI18ADJ01BT-RO, Bluetooth, Alámbrico/Inalámbrico, 3.5mm,
Rojo', 160, 'audifonos', 139],
[94, 'HyperX Audífonos Gamer Cloud Flight para PC/PS4/PS4 Pro, Inalámbrico, USB, 3.5mm, Negro',
2869, 'audifonos', 12],
[95, 'Iogear Audífonos Gamer GHG601, Alámbrico, 1.2 Metros, 3.5mm, Negro', 999, 'audifonos', 2],
[96, 'Klip Xtreme Audífonos Blast, Bluetooth, Inalámbrico, Negro/Verde', 769, 'audifonos', 2]
]

```

```

lifestore_sales = [
[1, 1, 5, '24/07/2020', 0],
[2, 1, 5, '27/07/2020', 0],
[3, 2, 5, '24/02/2020', 0],
[4, 2, 5, '22/05/2020', 0],
[5, 2, 5, '01/01/2020', 0],
[6, 2, 5, '24/04/2020', 0],
[7, 2, 4, '31/01/2020', 0],
[8, 2, 4, '07/02/2020', 0],
[9, 2, 4, '02/03/2020', 0],
[10, 2, 4, '07/03/2020', 0],
[11, 2, 4, '24/03/2020', 0],
[12, 2, 4, '24/04/2020', 0],
[13, 2, 4, '02/05/2020', 0],
[14, 2, 4, '03/06/2020', 0],
[15, 2, 3, '10/11/2019', 1],
[16, 3, 5, '21/07/2020', 0],
[17, 3, 4, '21/07/2020', 0],
[18, 3, 5, '11/06/2020', 0],
[19, 3, 5, '11/06/2020', 0],
[20, 3, 5, '20/05/2020', 0],
[21, 3, 5, '15/05/2020', 0],
[22, 3, 5, '02/05/2020', 0],
[23, 3, 5, '30/04/2020', 0],

```

[24, 3, 5, '27/04/2020', 0],
[25, 3, 4, '22/04/2020', 0],
[26, 3, 5, '19/04/2020', 0],
[27, 3, 5, '16/04/2020', 0],
[28, 3, 3, '14/04/2020', 0],
[29, 3, 5, '14/04/2020', 0],
[30, 3, 5, '14/04/2020', 0],
[31, 3, 5, '13/04/2020', 0],
[32, 3, 5, '13/04/2020', 0],
[33, 3, 5, '06/04/2020', 0],
[34, 3, 5, '02/04/2020', 0],
[35, 3, 5, '01/04/2020', 0],
[36, 3, 5, '16/03/2020', 0],
[37, 3, 5, '11/03/2020', 0],
[38, 3, 4, '10/03/2020', 0],
[39, 3, 5, '02/03/2020', 0],
[40, 3, 5, '27/02/2020', 0],
[41, 3, 4, '27/02/2020', 0],
[42, 3, 5, '03/02/2020', 0],
[43, 3, 5, '31/01/2020', 0],
[44, 3, 5, '30/01/2020', 0],
[45, 3, 5, '28/01/2020', 0],
[46, 3, 5, '25/01/2020', 0],
[47, 3, 5, '19/01/2020', 0],
[48, 3, 5, '13/01/2020', 0],
[49, 3, 5, '11/01/2020', 0],
[50, 3, 4, '09/01/2020', 0],
[51, 3, 5, '08/01/2020', 0],
[52, 3, 4, '06/01/2020', 0],
[53, 3, 5, '04/01/2020', 0],
[54, 3, 5, '04/01/2020', 0],
[55, 3, 5, '03/01/2020', 0],
[56, 3, 5, '02/01/2020', 0],
[57, 3, 5, '01/01/2020', 0],
[58, 4, 4, '19/06/2020', 0],
[59, 4, 4, '04/06/2020', 0],
[60, 4, 5, '16/04/2020', 0],
[61, 4, 4, '07/04/2020', 0],
[62, 4, 5, '06/04/2020', 0],
[63, 4, 5, '06/04/2020', 0],
[64, 4, 5, '30/03/2020', 0],
[65, 4, 4, '08/03/2020', 0],
[66, 4, 5, '25/02/2020', 0],
[67, 4, 3, '29/01/2020', 0],
[68, 4, 5, '23/01/2020', 0],
[69, 4, 4, '11/01/2020', 0],
[70, 4, 5, '09/01/2020', 0],
[71, 5, 4, '03/07/2020', 0],
[72, 5, 4, '14/05/2020', 0],
[73, 5, 4, '05/05/2020', 0],
[74, 5, 5, '04/05/2020', 0],
[75, 5, 4, '04/05/2020', 0],
[76, 5, 5, '03/05/2020', 0],
[77, 5, 5, '26/04/2020', 0],
[78, 5, 5, '23/04/2020', 0],
[79, 5, 5, '17/04/2020', 0],
[80, 5, 5, '13/04/2020', 0],
[81, 5, 5, '06/04/2020', 0],
[82, 5, 5, '26/04/2020', 0],
[83, 5, 5, '24/03/2020', 0],
[84, 5, 5, '22/03/2020', 0],
[85, 5, 4, '10/03/2020', 0],
[86, 5, 5, '25/02/2020', 0],
[87, 5, 4, '24/02/2020', 0],
[88, 5, 5, '15/02/2020', 0],
[89, 5, 5, '30/01/2020', 0],
[90, 5, 5, '17/01/2020', 0],
[91, 6, 5, '05/05/2020', 0],
[92, 6, 5, '22/03/2020', 0],

[93, 6, 5, '04/02/2020', 0],
[94, 7, 5, '25/07/2020', 0],
[95, 7, 5, '17/06/2020', 0],
[96, 7, 5, '15/04/2020', 0],
[97, 7, 5, '03/04/2020', 0],
[98, 7, 5, '31/03/2020', 0],
[99, 7, 5, '28/03/2020', 0],
[100, 7, 5, '22/02/2020', 0],
[101, 8, 5, '20/04/2020', 0],
[102, 8, 5, '16/02/2020', 0],
[103, 8, 5, '27/01/2020', 0],
[104, 8, 5, '20/01/2020', 0],
[105, 10, 4, '14/05/2020', 0],
[106, 11, 5, '30/06/2020', 0],
[107, 11, 5, '02/04/2020', 0],
[108, 11, 5, '05/03/2020', 0],
[109, 12, 5, '05/05/2020', 0],
[110, 12, 4, '09/04/2020', 0],
[111, 12, 5, '09/04/2020', 0],
[112, 12, 5, '02/04/2020', 0],
[113, 12, 5, '25/03/2020', 0],
[114, 12, 5, '24/03/2020', 0],
[115, 12, 5, '06/03/2020', 0],
[116, 12, 5, '04/03/2020', 0],
[117, 12, 4, '27/02/2020', 0],
[118, 13, 4, '17/04/2020', 0],
[119, 17, 1, '05/09/2020', 1],
[120, 18, 5, '30/06/2020', 0],
[121, 18, 4, '14/03/2020', 0],
[122, 18, 5, '27/02/2020', 0],
[123, 18, 4, '02/02/2020', 0],
[124, 18, 4, '01/02/2020', 0],
[125, 21, 5, '14/04/2020', 0],
[126, 21, 5, '12/02/2020', 0],
[127, 22, 5, '20/04/2020', 0],
[128, 25, 5, '28/03/2020', 0],
[129, 25, 5, '20/03/2020', 0],
[130, 28, 5, '30/03/2020', 0],
[131, 29, 4, '04/05/2020', 0],
[132, 29, 5, '24/04/2020', 0],
[133, 29, 4, '24/04/2020', 0],
[134, 29, 4, '17/04/2020', 0],
[135, 29, 5, '04/04/2020', 0],
[136, 29, 5, '09/03/2020', 0],
[137, 29, 5, '07/03/2020', 0],
[138, 29, 5, '26/02/2020', 0],
[139, 29, 5, '09/02/2020', 0],
[140, 29, 5, '06/02/2020', 0],
[141, 29, 5, '26/01/2020', 0],
[142, 29, 4, '25/01/2020', 0],
[143, 29, 1, '13/01/2020', 1],
[144, 29, 1, '10/01/2020', 0],
[145, 31, 1, '02/05/2020', 1],
[146, 31, 1, '02/05/2020', 1],
[147, 31, 1, '01/04/2020', 1],
[148, 31, 4, '20/03/2020', 0],
[149, 31, 3, '14/03/2020', 0],
[150, 31, 1, '11/01/2020', 0],
[151, 33, 5, '20/03/2020', 0],
[152, 33, 4, '27/02/2020', 0],
[153, 40, 5, '24/05/2020', 0],
[154, 42, 5, '27/07/2020', 0],
[155, 42, 5, '04/05/2020', 0],
[156, 42, 4, '04/05/2020', 0],
[157, 42, 4, '04/05/2020', 0],
[158, 42, 5, '04/05/2020', 0],
[159, 42, 5, '27/04/2020', 0],
[160, 42, 5, '26/04/2020', 0],
[161, 42, 4, '19/04/2020', 0],

[162, 42, 5, '14/04/2020', 0],
[163, 42, 5, '09/04/2020', 0],
[164, 42, 4, '05/04/2020', 0],
[165, 42, 4, '21/03/2020', 0],
[166, 42, 5, '09/03/2020', 0],
[167, 42, 5, '09/03/2020', 0],
[168, 42, 5, '03/03/2020', 0],
[169, 42, 4, '23/02/2020', 0],
[170, 42, 4, '03/02/2020', 0],
[171, 42, 4, '09/01/2020', 0],
[172, 44, 5, '16/04/2020', 0],
[173, 44, 5, '11/04/2020', 0],
[174, 44, 5, '21/03/2020', 0],
[175, 44, 4, '02/03/2020', 0],
[176, 44, 4, '01/03/2020', 0],
[177, 44, 5, '05/01/2020', 0],
[178, 45, 1, '11/02/2020', 1],
[179, 46, 2, '07/03/2020', 1],
[180, 47, 4, '02/07/2020', 0],
[181, 47, 5, '10/06/2020', 0],
[182, 47, 5, '18/04/2020', 0],
[183, 47, 4, '16/04/2020', 0],
[184, 47, 5, '08/04/2020', 0],
[185, 47, 4, '07/04/2020', 0],
[186, 47, 5, '23/03/2020', 0],
[187, 47, 5, '10/03/2020', 0],
[188, 47, 3, '11/02/2020', 0],
[189, 47, 5, '18/01/2020', 0],
[190, 47, 5, '17/01/2020', 0],
[191, 48, 4, '02/08/2020', 0],
[192, 48, 3, '27/04/2020', 0],
[193, 48, 5, '25/04/2020', 0],
[194, 48, 5, '23/04/2020', 0],
[195, 48, 5, '22/02/2020', 0],
[196, 48, 5, '10/02/2020', 0],
[197, 48, 5, '14/01/2020', 0],
[198, 48, 5, '09/01/2020', 0],
[199, 48, 5, '09/01/2020', 0],
[200, 49, 5, '06/04/2020', 0],
[201, 49, 5, '19/04/2020', 0],
[202, 49, 5, '22/04/2020', 0],
[203, 50, 5, '04/05/2020', 0],
[204, 51, 5, '23/03/2020', 0],
[205, 51, 4, '04/02/2020', 0],
[206, 51, 5, '03/01/2020', 0],
[207, 52, 5, '19/03/2020', 0],
[208, 52, 5, '02/01/2020', 0],
[209, 54, 4, '03/08/2020', 0],
[210, 54, 5, '02/08/2020', 0],
[211, 54, 5, '04/07/2020', 0],
[212, 54, 5, '01/07/2020', 0],
[213, 54, 5, '03/06/2020', 0],
[214, 54, 5, '23/05/2020', 0],
[215, 54, 4, '15/05/2020', 0],
[216, 54, 5, '11/05/2020', 0],
[217, 54, 5, '08/05/2020', 0],
[218, 54, 5, '04/05/2020', 0],
[219, 54, 4, '04/05/2002', 0],
[220, 54, 5, '04/05/2020', 0],
[221, 54, 5, '04/05/2020', 0],
[222, 54, 4, '30/04/2020', 0],
[223, 54, 4, '24/04/2020', 0],
[224, 54, 5, '23/04/2020', 0],
[225, 54, 4, '17/04/2020', 0],
[226, 54, 5, '15/04/2020', 0],
[227, 54, 5, '14/04/2020', 0],
[228, 54, 4, '14/04/2020', 0],
[229, 54, 5, '13/04/2020', 0],
[230, 54, 5, '13/04/2020', 0],

```


[231, 54, 5, '13/04/2020', 0],
[232, 54, 5, '09/04/2020', 0],
[233, 54, 5, '03/04/2020', 0],
[234, 54, 5, '03/04/2020', 0],
[235, 54, 5, '30/03/2020', 0],
[236, 54, 5, '26/03/2020', 0],
[237, 54, 5, '20/03/2020', 0],
[238, 54, 2, '19/03/2020', 1],
[239, 54, 5, '17/03/2020', 0],
[240, 54, 5, '14/03/2020', 0],
[241, 54, 5, '13/03/2020', 0],
[242, 54, 4, '02/03/2020', 0],
[243, 54, 5, '01/03/2020', 0],
[244, 54, 5, '25/02/2020', 0],
[245, 54, 5, '20/02/2020', 0],
[246, 54, 4, '17/02/2020', 0],
[247, 54, 5, '14/02/2020', 0],
[248, 54, 5, '12/02/2020', 0],
[249, 54, 4, '10/02/2020', 0],
[250, 54, 5, '07/02/2020', 0],
[251, 54, 5, '31/01/2020', 0],
[252, 54, 5, '30/01/2020', 0],
[253, 54, 5, '29/01/2020', 0],
[254, 54, 5, '27/01/2020', 0],
[255, 54, 5, '25/01/2020', 0],
[256, 54, 5, '23/01/2020', 0],
[257, 54, 5, '23/01/2020', 0],
[258, 54, 4, '22/01/2020', 0],
[259, 57, 5, '05/07/2020', 0],
[260, 57, 5, '23/05/2020', 0],
[261, 57, 5, '23/05/2020', 0],
[262, 57, 5, '01/05/2020', 0],
[263, 57, 5, '06/04/2020', 0],
[264, 57, 5, '09/03/2020', 0],
[265, 57, 5, '25/02/2020', 0],
[266, 57, 5, '10/02/2020', 0],
[267, 57, 4, '04/02/2020', 0],
[268, 57, 5, '04/02/2020', 0],
[269, 57, 5, '28/01/2020', 0],
[270, 57, 5, '27/01/2020', 0],
[271, 57, 4, '22/01/2020', 0],
[272, 57, 5, '08/01/2020', 0],
[273, 57, 5, '07/01/2020', 0],
[274, 60, 5, '17/06/2020', 0],
[275, 66, 5, '06/05/2020', 0],
[276, 67, 5, '24/04/2020', 0],
[277, 74, 4, '12/02/2020', 0],
[278, 74, 5, '18/02/2020', 0],
[279, 84, 5, '05/05/2020', 0],
[280, 85, 5, '05/05/2020', 0],
[281, 85, 5, '28/04/2020', 0],
[282, 89, 3, '06/01/2020', 0],
[283, 94, 4, '10/04/2020', 0]
]

```


```


lifestore_searches = [
    [1, 1],
    [2, 1],
    [3, 1],
    [4, 1],
    [5, 1],
    [6, 1],
    [7, 1],
    [8, 1],
    [9, 1],
    [10, 1],
    [11, 2],
    [12, 2],
    [13, 2],

```





[14, 2],
[15, 2],
[16, 2],
[17, 2],
[18, 2],
[19, 2],
[20, 2],
[21, 2],
[22, 2],
[23, 2],
[24, 2],
[25, 2],
[26, 2],
[27, 2],
[28, 2],
[29, 2],
[30, 2],
[31, 2],
[32, 2],
[33, 2],
[34, 2],
[35, 3],
[36, 3],
[37, 3],
[38, 3],
[39, 3],
[40, 3],
[41, 3],
[42, 3],
[43, 3],
[44, 3],
[45, 3],
[46, 3],
[47, 3],
[48, 3],
[49, 3],
[50, 3],
[51, 3],
[52, 3],
[53, 3],
[54, 3],
[55, 3],
[56, 3],
[57, 3],
[58, 3],
[59, 3],
[60, 3],
[61, 3],
[62, 3],
[63, 3],
[64, 3],
[65, 3],
[66, 3],
[67, 3],
[68, 3],
[69, 3],
[70, 3],
[71, 3],
[72, 3],
[73, 3],
[74, 3],
[75, 3],
[76, 3],
[77, 3],
[78, 3],
[79, 3],
[80, 3],
[81, 3],
[82, 3],







[83, 3],
[84, 3],
[85, 3],
[86, 3],
[87, 3],
[88, 3],
[89, 3],
[90, 4],
[91, 4],
[92, 4],
[93, 4],
[94, 4],
[95, 4],
[96, 4],
[97, 4],
[98, 4],
[99, 4],
[100, 4],
[101, 4],
[102, 4],
[103, 4],
[104, 4],
[105, 4],
[106, 4],
[107, 4],
[108, 4],
[109, 4],
[110, 4],
[111, 4],
[112, 4],
[113, 4],
[114, 4],
[115, 4],
[116, 4],
[117, 4],
[118, 4],
[119, 4],
[120, 4],
[121, 4],
[122, 4],
[123, 4],
[124, 4],
[125, 4],
[126, 4],
[127, 4],
[128, 4],
[129, 4],
[130, 4],
[131, 5],
[132, 5],
[133, 5],
[134, 5],
[135, 5],
[136, 5],
[137, 5],
[138, 5],
[139, 5],
[140, 5],
[141, 5],
[142, 5],
[143, 5],
[144, 5],
[145, 5],
[146, 5],
[147, 5],
[148, 5],
[149, 5],
[150, 5],
[151, 5],







[152, 5],
[153, 5],
[154, 5],
[155, 5],
[156, 5],
[157, 5],
[158, 5],
[159, 5],
[160, 5],
[161, 6],
[162, 6],
[163, 6],
[164, 6],
[165, 6],
[166, 6],
[167, 6],
[168, 6],
[169, 6],
[170, 6],
[171, 7],
[172, 7],
[173, 7],
[174, 7],
[175, 7],
[176, 7],
[177, 7],
[178, 7],
[179, 7],
[180, 7],
[181, 7],
[182, 7],
[183, 7],
[184, 7],
[185, 7],
[186, 7],
[187, 7],
[188, 7],
[189, 7],
[190, 7],
[191, 7],
[192, 7],
[193, 7],
[194, 7],
[195, 7],
[196, 7],
[197, 7],
[198, 7],
[199, 7],
[200, 7],
[201, 7],
[202, 8],
[203, 8],
[204, 8],
[205, 8],
[206, 8],
[207, 8],
[208, 8],
[209, 8],
[210, 8],
[211, 8],
[212, 8],
[213, 8],
[214, 8],
[215, 8],
[216, 8],
[217, 8],
[218, 8],
[219, 8],
[220, 8],







[221, 8],
[222, 9],
[223, 10],
[224, 11],
[225, 11],
[226, 11],
[227, 11],
[228, 11],
[229, 12],
[230, 12],
[231, 12],
[232, 12],
[233, 12],
[234, 12],
[235, 12],
[236, 12],
[237, 12],
[238, 12],
[239, 12],
[240, 12],
[241, 12],
[242, 12],
[243, 12],
[244, 13],
[245, 13],
[246, 15],
[247, 15],
[248, 15],
[249, 15],
[250, 17],
[251, 17],
[252, 17],
[253, 18],
[254, 18],
[255, 18],
[256, 18],
[257, 18],
[258, 18],
[259, 18],
[260, 18],
[261, 18],
[262, 18],
[263, 18],
[264, 21],
[265, 21],
[266, 21],
[267, 21],
[268, 21],
[269, 21],
[270, 21],
[271, 21],
[272, 21],
[273, 21],
[274, 21],
[275, 21],
[276, 21],
[277, 21],
[278, 21],
[279, 22],
[280, 22],
[281, 22],
[282, 22],
[283, 22],
[284, 25],
[285, 25],
[286, 25],
[287, 25],
[288, 25],
[289, 25],







[290, 25],
[291, 25],
[292, 25],
[293, 25],
[294, 26],
[295, 26],
[296, 26],
[297, 26],
[298, 26],
[299, 27],
[300, 28],
[301, 28],
[302, 28],
[303, 28],
[304, 28],
[305, 29],
[306, 29],
[307, 29],
[308, 29],
[309, 29],
[310, 29],
[311, 29],
[312, 29],
[313, 29],
[314, 29],
[315, 29],
[316, 29],
[317, 29],
[318, 29],
[319, 29],
[320, 29],
[321, 29],
[322, 29],
[323, 29],
[324, 29],
[325, 29],
[326, 29],
[327, 29],
[328, 29],
[329, 29],
[330, 29],
[331, 29],
[332, 29],
[333, 29],
[334, 29],
[335, 29],
[336, 29],
[337, 29],
[338, 29],
[339, 29],
[340, 29],
[341, 29],
[342, 29],
[343, 29],
[344, 29],
[345, 29],
[346, 29],
[347, 29],
[348, 29],
[349, 29],
[350, 29],
[351, 29],
[352, 29],
[353, 29],
[354, 29],
[355, 29],
[356, 29],
[357, 29],
[358, 29],







[359, 29],
[360, 29],
[361, 29],
[362, 29],
[363, 29],
[364, 29],
[365, 31],
[366, 31],
[367, 31],
[368, 31],
[369, 31],
[370, 31],
[371, 31],
[372, 31],
[373, 31],
[374, 31],
[375, 35],
[376, 39],
[377, 39],
[378, 39],
[379, 40],
[380, 40],
[381, 40],
[382, 40],
[383, 40],
[384, 40],
[385, 40],
[386, 40],
[387, 40],
[388, 40],
[389, 42],
[390, 42],
[391, 42],
[392, 42],
[393, 42],
[394, 42],
[395, 42],
[396, 42],
[397, 42],
[398, 42],
[399, 42],
[400, 42],
[401, 42],
[402, 42],
[403, 42],
[404, 42],
[405, 42],
[406, 42],
[407, 42],
[408, 42],
[409, 42],
[410, 42],
[411, 42],
[412, 44],
[413, 44],
[414, 44],
[415, 44],
[416, 44],
[417, 44],
[418, 44],
[419, 44],
[420, 44],
[421, 44],
[422, 44],
[423, 44],
[424, 44],
[425, 44],
[426, 44],
[427, 44],







[428, 44],
[429, 44],
[430, 44],
[431, 44],
[432, 44],
[433, 44],
[434, 44],
[435, 44],
[436, 44],
[437, 45],
[438, 46],
[439, 46],
[440, 46],
[441, 46],
[442, 47],
[443, 47],
[444, 47],
[445, 47],
[446, 47],
[447, 47],
[448, 47],
[449, 47],
[450, 47],
[451, 47],
[452, 47],
[453, 47],
[454, 47],
[455, 47],
[456, 47],
[457, 47],
[458, 47],
[459, 47],
[460, 47],
[461, 47],
[462, 47],
[463, 47],
[464, 47],
[465, 47],
[466, 47],
[467, 47],
[468, 47],
[469, 47],
[470, 47],
[471, 47],
[472, 48],
[473, 48],
[474, 48],
[475, 48],
[476, 48],
[477, 48],
[478, 48],
[479, 48],
[480, 48],
[481, 48],
[482, 48],
[483, 48],
[484, 48],
[485, 48],
[486, 48],
[487, 48],
[488, 48],
[489, 48],
[490, 48],
[491, 48],
[492, 48],
[493, 48],
[494, 48],
[495, 48],
[496, 48],







[497, 48],
[498, 48],
[499, 49],
[500, 49],
[501, 49],
[502, 49],
[503, 49],
[504, 49],
[505, 49],
[506, 49],
[507, 49],
[508, 49],
[509, 50],
[510, 50],
[511, 50],
[512, 50],
[513, 50],
[514, 50],
[515, 50],
[516, 51],
[517, 51],
[518, 51],
[519, 51],
[520, 51],
[521, 51],
[522, 51],
[523, 51],
[524, 51],
[525, 51],
[526, 51],
[527, 52],
[528, 52],
[529, 52],
[530, 52],
[531, 52],
[532, 54],
[533, 54],
[534, 54],
[535, 54],
[536, 54],
[537, 54],
[538, 54],
[539, 54],
[540, 54],
[541, 54],
[542, 54],
[543, 54],
[544, 54],
[545, 54],
[546, 54],
[547, 54],
[548, 54],
[549, 54],
[550, 54],
[551, 54],
[552, 54],
[553, 54],
[554, 54],
[555, 54],
[556, 54],
[557, 54],
[558, 54],
[559, 54],
[560, 54],
[561, 54],
[562, 54],
[563, 54],
[564, 54],
[565, 54],







[566, 54],
[567, 54],
[568, 54],
[569, 54],
[570, 54],
[571, 54],
[572, 54],
[573, 54],
[574, 54],
[575, 54],
[576, 54],
[577, 54],
[578, 54],
[579, 54],
[580, 54],
[581, 54],
[582, 54],
[583, 54],
[584, 54],
[585, 54],
[586, 54],
[587, 54],
[588, 54],
[589, 54],
[590, 54],
[591, 54],
[592, 54],
[593, 54],
[594, 54],
[595, 54],
[596, 54],
[597, 54],
[598, 54],
[599, 54],
[600, 54],
[601, 54],
[602, 54],
[603, 54],
[604, 54],
[605, 54],
[606, 54],
[607, 54],
[608, 54],
[609, 54],
[610, 54],
[611, 54],
[612, 54],
[613, 54],
[614, 54],
[615, 54],
[616, 54],
[617, 54],
[618, 54],
[619, 54],
[620, 54],
[621, 54],
[622, 54],
[623, 54],
[624, 54],
[625, 54],
[626, 54],
[627, 54],
[628, 54],
[629, 54],
[630, 54],
[631, 54],
[632, 54],
[633, 54],
[634, 54],







[635, 54],
[636, 54],
[637, 54],
[638, 54],
[639, 54],
[640, 54],
[641, 54],
[642, 54],
[643, 54],
[644, 54],
[645, 54],
[646, 54],
[647, 54],
[648, 54],
[649, 54],
[650, 54],
[651, 54],
[652, 54],
[653, 54],
[654, 54],
[655, 54],
[656, 54],
[657, 54],
[658, 54],
[659, 54],
[660, 54],
[661, 54],
[662, 54],
[663, 54],
[664, 54],
[665, 54],
[666, 54],
[667, 54],
[668, 54],
[669, 54],
[670, 54],
[671, 54],
[672, 54],
[673, 54],
[674, 54],
[675, 54],
[676, 54],
[677, 54],
[678, 54],
[679, 54],
[680, 54],
[681, 54],
[682, 54],
[683, 54],
[684, 54],
[685, 54],
[686, 54],
[687, 54],
[688, 54],
[689, 54],
[690, 54],
[691, 54],
[692, 54],
[693, 54],
[694, 54],
[695, 54],
[696, 54],
[697, 54],
[698, 54],
[699, 54],
[700, 54],
[701, 54],
[702, 54],
[703, 54],







[704, 54],
[705, 54],
[706, 54],
[707, 54],
[708, 54],
[709, 54],
[710, 54],
[711, 54],
[712, 54],
[713, 54],
[714, 54],
[715, 54],
[716, 54],
[717, 54],
[718, 54],
[719, 54],
[720, 54],
[721, 54],
[722, 54],
[723, 54],
[724, 54],
[725, 54],
[726, 54],
[727, 54],
[728, 54],
[729, 54],
[730, 54],
[731, 54],
[732, 54],
[733, 54],
[734, 54],
[735, 54],
[736, 54],
[737, 54],
[738, 54],
[739, 54],
[740, 54],
[741, 54],
[742, 54],
[743, 54],
[744, 54],
[745, 54],
[746, 54],
[747, 54],
[748, 54],
[749, 54],
[750, 54],
[751, 54],
[752, 54],
[753, 54],
[754, 54],
[755, 54],
[756, 54],
[757, 54],
[758, 54],
[759, 54],
[760, 54],
[761, 54],
[762, 54],
[763, 54],
[764, 54],
[765, 54],
[766, 54],
[767, 54],
[768, 54],
[769, 54],
[770, 54],
[771, 54],
[772, 54],







[773, 54],
[774, 54],
[775, 54],
[776, 54],
[777, 54],
[778, 54],
[779, 54],
[780, 54],
[781, 54],
[782, 54],
[783, 54],
[784, 54],
[785, 54],
[786, 54],
[787, 54],
[788, 54],
[789, 54],
[790, 54],
[791, 54],
[792, 54],
[793, 54],
[794, 54],
[795, 56],
[796, 56],
[797, 57],
[798, 57],
[799, 57],
[800, 57],
[801, 57],
[802, 57],
[803, 57],
[804, 57],
[805, 57],
[806, 57],
[807, 57],
[808, 57],
[809, 57],
[810, 57],
[811, 57],
[812, 57],
[813, 57],
[814, 57],
[815, 57],
[816, 57],
[817, 57],
[818, 57],
[819, 57],
[820, 57],
[821, 57],
[822, 57],
[823, 57],
[824, 57],
[825, 57],
[826, 57],
[827, 57],
[828, 57],
[829, 57],
[830, 57],
[831, 57],
[832, 57],
[833, 57],
[834, 57],
[835, 57],
[836, 57],
[837, 57],
[838, 57],
[839, 57],
[840, 57],
[841, 57],






[842, 57],
[843, 57],
[844, 57],
[845, 57],
[846, 57],
[847, 57],
[848, 57],
[849, 57],
[850, 57],
[851, 57],
[852, 57],
[853, 57],
[854, 57],
[855, 57],
[856, 57],
[857, 57],
[858, 57],
[859, 57],
[860, 57],
[861, 57],
[862, 57],
[863, 57],
[864, 57],
[865, 57],
[866, 57],
[867, 57],
[868, 57],
[869, 57],
[870, 57],
[871, 57],
[872, 57],
[873, 57],
[874, 57],
[875, 57],
[876, 57],
[877, 57],
[878, 57],
[879, 57],
[880, 57],
[881, 57],
[882, 57],
[883, 57],
[884, 57],
[885, 57],
[886, 57],
[887, 57],
[888, 57],
[889, 57],
[890, 57],
[891, 57],
[892, 57],
[893, 57],
[894, 57],
[895, 57],
[896, 57],
[897, 57],
[898, 57],
[899, 57],
[900, 57],
[901, 57],
[902, 57],
[903, 57],
[904, 59],
[905, 63],
[906, 63],
[907, 63],
[908, 63],
[909, 66],
[910, 66],





[911, 66],
[912, 66],
[913, 66],
[914, 66],
[915, 66],
[916, 66],
[917, 66],
[918, 66],
[919, 66],
[920, 66],
[921, 66],
[922, 66],
[923, 66],
[924, 67],
[925, 67],
[926, 67],
[927, 67],
[928, 67],
[929, 67],
[930, 67],
[931, 67],
[932, 67],
[933, 67],
[934, 67],
[935, 67],
[936, 67],
[937, 67],
[938, 67],
[939, 67],
[940, 67],
[941, 67],
[942, 67],
[943, 67],
[944, 67],
[945, 67],
[946, 67],
[947, 67],
[948, 67],
[949, 67],
[950, 67],
[951, 67],
[952, 67],
[953, 67],
[954, 67],
[955, 67],
[956, 70],
[957, 73],
[958, 73],
[959, 73],
[960, 73],
[961, 74],
[962, 74],
[963, 74],
[964, 74],
[965, 74],
[966, 74],
[967, 76],
[968, 76],
[969, 80],
[970, 84],
[971, 84],
[972, 84],
[973, 84],
[974, 84],
[975, 84],
[976, 84],
[977, 84],
[978, 84],
[979, 84],




```
[980, 85],
[981, 85],
[982, 85],
[983, 85],
[984, 85],
[985, 85],
[986, 85],
[987, 85],
[988, 85],
[989, 85],
[990, 85],
[991, 85],
[992, 85],
[993, 85],
[994, 85],
[995, 85],
[996, 85],
[997, 85],
[998, 85],
[999, 85],
[1000, 85],
[1001, 85],
[1002, 85],
[1003, 85],
[1004, 85],
[1005, 85],
[1006, 85],
[1007, 85],
[1008, 85],
[1009, 85],
[1010, 85],
[1011, 85],
[1012, 85],
[1013, 85],
[1014, 85],
[1015, 89],
[1016, 89],
[1017, 89],
[1018, 89],
[1019, 89],
[1020, 89],
[1021, 89],
[1022, 91],
[1023, 91],
[1024, 93],
[1025, 94],
[1026, 94],
[1027, 94],
[1028, 94],
[1029, 94],
[1030, 94],
[1031, 95],
[1032, 95],
[1033, 95]
]
```

Módulo de bienvenida al Sistema

```
"""
Comienza el Módulo del Sistema
"""
#Uso de librería básica para contraseñas
import getpass

#Librería para tabular resultados
from tabulate import tabulate
```

```
#Almacenamiento de usuarios y contraseñas; poco seguro pero aplicable al caso
usrlist = open('usrlist.txt').read().splitlines()
```

```
#Bienvenida y primera solicitud al usuario
print("Bienvenido a la tienda virtual LifeStore v1.0")
eleccion = input("Escribe una opción 'login' o 'register': ")
```

Aplicación de bucle While para login

```
#Aplicación de bucle while para reiniciar solicitud de login sin salir
while eleccion != "login" or "register":
    #login
    if eleccion == 'login':
        print("Ingrese un nombre de usuario y contraseña")
        usr = input("Usuario: ")
        psw = getpass.getpass("Contraseña: ")
        combo = usr + ':' + psw
        if combo in usrlist:

            print(f"\nInició sesión con el nombre de usuario: {usr} \n\nDerivado de la situación,
la Gerencia de Ventas de LifeStore solicitó mostrar un análisis de la rotación de productos.\n")
            while True:
```

Módulo inicial del Menú al Sistema

```
        print(f"{usr} selecciona la opción del reporte que deseas conocer:\n\n 1) Reporte de
productos más vendidos y productos rezagados.\n 2) Reporte de productos por reseña en el servicio.\n
3) Reporte del total de ingresos y ventas promedio mensuales, total anual y meses con más ventas al
año.\n Presiona x para salir.")
        opcion=input()
        if opcion=='1':
            print(f"\nReporte de productos más vendidos y productos rezagados\n")
```

Sección 1 – Resultados de productos más vendidos y rezagados

```
        """ SECCIÓN 1
        1) Productos más vendidos y productos rezagados a partir del análisis de las
        categorías con menores ventas y categorías con menores búsquedas.
        """

        """
        Productos más vendidos y productos rezagados:
        1.1 Generar un listado de los 15 productos con mayores ventas.
        """

        #Lista total de productos agrupados con número total de ventas
        prod_mas = []
        #Variable para el conteo de venta
        cont = 0

        #Función para enlistar productos y contabilizar el número de ventas
        for prod in lifestore_products:
            for vent in lifestore_sales:
                if prod[0] == (vent[1]) and vent[4]==0:
                    cont +=1
            prod_mas.append([prod[0],cont])
            cont=0

        #Función burbuja para reordenar el listado anterior de mayor a menor tomando el
        valor listado en la columna 2 (cont)
        for recorrido in range(1,len(prod_mas)):
            for posicion in range(len(prod_mas)-recorrido):
                if prod_mas[posicion][1] < prod_mas[posicion+1][1]:
                    temp = prod_mas[posicion]
                    prod_mas[posicion] = prod_mas[posicion + 1]
```

```

        prod_mas[posicion+1]=temp

#Imprimir titulo de la tabla señalando
print(f'Lista de Venta de Productos - Top 15 (descartando devoluciones)')

#Imprimir tabla de los 50 productos más vendidos
print(tabulate(prod_mas[:15],headers=['#','Id_Prod','Ventas'],tablefmt='grid',
showindex='always'))

"""
Productos más vendidos y productos rezagados:
1.2 Generar un listado de los 100 productos con mayor búsquedas.
"""

#Lista total de productos agrupados con número total de búsquedas
prod_bus = []
#Variable para el conteo de búsquedas
cont = 0

#Función para enlistar productos y contabilizar el número de búsquedas
for prod in lifestore_products:
    for busq in lifestore_searches:
        if prod[0] == (busq[1]):
            cont +=1
        prod_bus.append([prod[0],cont])
    cont=0

#Función burbuja para reordenar el listado anterior de mayor a menor tomando el
valor listado en la columna 2 (cont)
for recorrido in range(1,len(prod_bus)):
    for posicion in range(len(prod_bus)-recorrido):
        if prod_bus[posicion][1] < prod_bus[posicion+1][1]:
            temp = prod_bus[posicion]
            prod_bus[posicion] = prod_bus[posicion + 1]
            prod_bus[posicion+1]=temp

#Imprimir titulo de la tabla señalando
print(f'Lista de Productos con mayor búsqueda - Top 20')

#Imprimir tabla de los 20 productos más buscados
print(tabulate(prod_bus[:20],headers=['#','Id_Prod','#
Búsqueda'],tablefmt='grid', showindex='always'))

"""
Productos más vendidos y productos rezagados:
1.3 Por categoría generar un listado con los 50 productos con menores ventas.
"""

#Lista total de productos agrupados con número total de ventas
prod_men = []
#Lista de categorías
categor=[]
#Lista de ventas por categoría
categ_vent=[]
#Variable para el conteo de venta
cont = 0

#Listar todas las categorías presentes en lifestore_products (sin duplicar)
for cat in lifestore_products:
    if cat[3] not in categor:
        categor.append(cat[3])
#print(categor)

#Listar productos y contabilizar el número de ventas
for prod in lifestore_products:
    for vent in lifestore_sales:
        if prod[0] == (vent[1]) and vent[4]==0:
            cont +=1
        prod_men.append([prod[3],cont])

```

```

        cont=0
        #print(prod_men)

        #Cruzar lista de categorías con lista de productos con ventas (Merge)
        for i in categor:
            for p in prod_men:
                if p[0] == i:
                    cont += p[1]
            categ_vent.append([i,cont])

        #Función burbuja para reordenar el listado anterior de menor a mayor tomando el
valor listado en la columna 2 (cont)
        for recorrido in range(1,len(categ_vent)):
            for posicion in range(len(categ_vent)-recorrido):
                if categ_vent[posicion][1] > categ_vent[posicion+1][1]:
                    temp = categ_vent[posicion]
                    categ_vent[posicion] = categ_vent[posicion + 1]
                    categ_vent[posicion+1]=temp

        #Imprimir titulo de la tabla señalando
        print(f'Lista de Ventas de Productos por Categoría de Menor a Mayor (descartando
devoluciones)')

        #Imprimir tabla de los 50 productos más vendidos
        print(tabulate(categ_vent,headers=['#','Categ_Prod','Ventas'],tablefmt='grid',
showindex='always'))

        """
        Productos más vendidos y productos rezagados:
        1.4 Por categoría, generar un listado con los 100 productos con menores
búsquedas.
        """

        #Lista total de productos agrupados con número total de ventas
        prod_men = []
        #Lista de categorías
        categor=[]
        #Lista de ventas por categoría
        categ_vent=[]
        #Variable para el conteo de venta
        cont = 0

        #Listar todas las categorías presentes en lifestore_products (sin duplicar)
        for cat in lifestore_products:
            if cat[3] not in categor:
                categor.append(cat[3])
        #print(categor)

        #Lista total de productos agrupados con número total de búsquedas
        prod_bus = []

        #Función para enlistar productos y contabilizar el número de búsquedas
        for prod in lifestore_products:
            for busq in lifestore_searches:
                if prod[0] == (busq[1]):
                    cont +=1
                prod_bus.append([prod[3],cont])
            cont=0
        #print(prod_bus)

        #Cruzar lista de categorías con lista de productos con ventas (Merge)
        for i in categor:
            for p in prod_bus:
                if p[0] == i:
                    cont += p[1]
            categ_vent.append([i,cont])

        #Función burbuja para reordenar el listado anterior de menor a mayor tomando el
valor listado en la columna 2 (cont)

```

```

for recorrido in range(1,len(categ_vent)):
    for posicion in range(len(categ_vent)-recorrido):
        if categ_vent[posicion][1] > categ_vent[posicion+1][1]:
            temp = categ_vent[posicion]
            categ_vent[posicion] = categ_vent[posicion + 1]
            categ_vent[posicion+1]=temp

#Imprimir titulo de la tabla señalando
print(f'Lista de Búsquedas de Productos por Categoría de Menor a Mayor')

#Imprimir tabla de los 50 productos más vendidos

print(tabulate(categ_vent,headers=['#', 'Categ_Prod', 'Búsquedas'],tablefmt='grid',
showindex='always'))

```

Sección 2 – Productos por reseña

```

elif opcion=='2':
    print(f"\nReporte de productos por reseña\n")

    """ SECCIÓN 2
    2) Productos por reseña en el servicio a partir del análisis de categorías con
    mayores ventas y categorías con mayores búsquedas.
    """

    """
    Productos por reseña en el servicio:
    2.1 Mostrar dos listados de 20 productos cada uno, un listado para productos
    con las mejores reseñas y otro para las peores, considerando los productos con devolución.
    """

    #Lista total de productos agrupados con número total de reseñas
    prod_mas = []
    #Variable para el conteo de reseñas
    cont = 0

    #Función para enlistar productos y contabilizar el número de reseñas
    for prod in lifestore_products:
        for vent in lifestore_sales:
            if prod[0] == (vent[1]) and vent[4]==0:
                cont +=vent[2]
            prod_mas.append([prod[0],cont])
        cont=0

    #Función (MEJORES RESEÑAS) burbuja para reordenar el listado anterior de mayor a
    menor tomando el valor listado en la columna 2 (cont)
    for recorrido in range(1,len(prod_mas)):
        for posicion in range(len(prod_mas)-recorrido):
            if prod_mas[posicion][1] < prod_mas[posicion+1][1]:
                temp = prod_mas[posicion]
                prod_mas[posicion] = prod_mas[posicion + 1]
                prod_mas[posicion+1]=temp

    #Imprimir titulo de la tabla señalando
    print(f'Lista de Mejores Reseñas de Productos - Top 10 (descartando
    devoluciones)')

    #Imprimir tabla de los 50 productos más vendidos
    print(tabulate(prod_mas[:10],headers=['#', 'Id_Prod', 'Ptos Total
    Reseñas'],tablefmt='grid', showindex='always'))

    #Función (PEORES RESEÑAS) burbuja para reordenar el listado anterior de mayor a
    menor tomando el valor listado en la columna 2 (cont)
    for recorrido in range(1,len(prod_mas)):
        for posicion in range(len(prod_mas)-recorrido):
            if prod_mas[posicion][1] > prod_mas[posicion+1][1]:
                temp = prod_mas[posicion]
                prod_mas[posicion] = prod_mas[posicion + 1]
                prod_mas[posicion+1]=temp

```

```

#Imprimir titulo de la tabla señalando
print(f'Lista de Peores Reseñas de Productos - Top 10 (descartando
devoluciones)')

#Imprimir tabla de los 50 productos más vendidos
#El resultado muestra todos en 0 ya que solo toma los primeros 10 ordenador por
id de producto
print(tabulate(prod_mas[:10],headers=['#','Id_Prod','Ptos Total
Reseñas'],tablefmt='grid', showindex='always'))

```

Sección 3 - Total de ingresos y ventas (Estrategia)

```

elif opcion=='3':
    print(f"\nTotal de ingresos y ventas promedio mensuales, total anual y meses con
    más ventas al año\n")

    """ SECCIÓN 3
    3) Sugerir una estrategia de productos a retirar del mercado así como sugerencia
    de cómo reducir la acumulación de inventario considerando los datos de ingresos y ventas mensuales.
    """

    """
    3.1 Total de ingresos y ventas promedio mensuales, total anual y meses con más
    ventas al año
    """

    #Total de ingresos de todos los productos vendidos
    total_ing = 0

    #Función para sumar los ingresos por venta de productos
    for prod in lifestore_products:
        for vent in lifestore_sales:
            if prod[0] == (vent[1]) and vent[4]==0:
                total_ing +=prod[2] #Sumarizamos el ingreso por producto vendido
    print(f"Total de Ingresos: {total_ing}\n")

    #Lista total de productos agrupados con número total de ventas
    prod_men = []
    #Lista de categorías
    meses=["01","02","03","04","05","06","07","08","09","10","11","12"]
    #Lista de ventas por categoría
    categ_vent=[]
    #Variable para el conteo de ingreso
    ing = 0

    #Listar ventas de productos y contabilizar el ingreso
    for vent in lifestore_sales:
        for prod in lifestore_products:
            if prod[0] == (vent[1]) and vent[4]==0: #Ultima sentencia para descartar
                ing =prod[2]
                cat= prod[3]
                prod_men.append([vent[3],cat,ing])

    #print(prod_men[0][0][3:5]) #Pruebas para validar salida
    #print(prod_men[0:5]) #Pruebas para validar salida

    suma=0
    #Cruzar lista de categorías con lista de productos con ventas (Merge)
    for i in meses:
        for p in prod_men:
            #print(f"{i} Fecha: {p[0][3:5]}")
            if str(i) == str(p[0][3:5]):
                #print(i) #Pruebas para validar salida
                suma += p[2]
                categ_vent.append([i,suma])
    suma=0

```

```

#Imprimir titulo de la tabla señalando
print(f'Ventas Mensuales de Productos (descartando devoluciones)')

#Imprimir tabla de los 50 productos más vendidos
print(tabulate(categ_vent,headers=['Mes','Total Ventas'],tablefmt='grid'))

#Función burbuja para reordenar el listado anterior de menor a mayor tomando el
valor listado en la columna 2 (cont)
for recorrido in range(1,len(categ_vent)):
    for posicion in range(len(categ_vent)-recorrido):
        if categ_vent[posicion][1] < categ_vent[posicion+1][1]:
            temp = categ_vent[posicion]
            categ_vent[posicion] = categ_vent[posicion + 1]
            categ_vent[posicion+1]=temp

#Imprimir titulo de la tabla señalando
print(f'Productos con más Ventas - Top 5 (descartando devoluciones)')

#Imprimir tabla de los 50 productos más vendidos
print(tabulate(categ_vent[:5],headers=['Mes','Total Ventas'],tablefmt='grid'))

#Lista total de productos agrupados con número total de ventas
prod_mas = []
#Variable para el conteo de venta
cont = 0

#Función para enlistar productos y contabilizar el número de ventas
for prod in lifestore_products:
    for vent in lifestore_sales:
        if prod[0] == (vent[1]) and vent[4]==0:
            cont +=1
        if cont == 0:
            prod_mas.append([prod[0],cont])
            cont=0

#Función burbuja para reordenar el listado anterior de mayor a menor tomando el
valor listado en la columna 2 (cont)
for recorrido in range(1,len(prod_mas)):
    for posicion in range(len(prod_mas)-recorrido):
        if prod_mas[posicion][1] > prod_mas[posicion+1][1]:
            temp = prod_mas[posicion]
            prod_mas[posicion] = prod_mas[posicion + 1]
            prod_mas[posicion+1]=temp

#Imprimir titulo de la tabla señalando
print(f'Lista Productos con $0 Ventas que se recomienda como estrategia
retirarlos del mercado; tambien para reducir la acumulación de inventario se recomienda:')
sin_venta=[]
for pro in prod_mas:
    sin_venta.append(pro[0])
print(sin_venta)

```

Opción de cierre del menú de Sistema

```

elif opcion=='x':
    exit()

else:
    print("No existe registro o el nombre de usuario y contraseña son incorrectos")
    continue

```

Módulo de registro del Sistema

```

#registro
if eleccion == 'register':
    usr_valid = True
    print("Ingrese el nombre de usuario deseado")

```

```

usr_elegido = input("Usuario: ")
for combo in usrlist:
    if usr_elegido in combo:
        print("¡El nombre de usuario no está disponible! ¡Escoge otro!")
        usr_valid = False
if usr_valid == True:
    psw_elegido = getpass.getpass("Contraseña: ")
    psw_elegido2 = getpass.getpass("Confirma la contraseña: ")
    while psw_elegido != psw_elegido2:
        print("La contraseña no coincide")
        psw_elegido2 = getpass.getpass("Confirma la contraseña: ")
    combo = usr_elegido + ":" + psw_elegido
    f1 = open('usrlist.txt', 'a+')
    f1.writable()
    f1.write(combo + "\n")
    f1.close()
    print("¡Registro exitoso! Intente iniciar sesión.")
    #Inicializamos de nuevo la lectura para que valide el nuevo registro
    usrlist = open('usrlist.txt').read().splitlines()

```

Módulo en caso de opción incorrecta del Sistema

```

# error
if eleccion != "login" or "register" and usr_valid == False:
    eleccion = input("Escribe una opción 'login' o 'register': ")

```

Solución al Problema

Repositorio GIT: <https://github.com/Angel-Carrillo/PROYECTO-01-CARRILLO-ANGEL>

Tomando como referencia todos los conceptos aprendidos en el curso virtual, con excepción de las librerías para *user/password* para registrarlos en un archivo adjunto en la raíz y *tabulate* para presentar resultados, se desarrolló un Sistema que comprende un módulo de bienvenida. (ver *Módulo de bienvenida al Sistema*)

Se aplicó el bucle *While* para reiterar al menú inicial con las posibles opciones para que el usuario escogiera. Esto con el fin de darle una experiencia similar a los Sistemas.

Para dar solución al caso, se definió un menú en función de los puntos solicitados; los resultados se agruparon de la siguiente forma:

- Sección 1 - Resultados de productos más vendidos y rezagados
- Sección 2 - Productos por reseña
- Sección 3 - Total de ingresos y ventas (Estrategia)
- Opción de cierre del menú de Sistema

Conclusión

La información recolectada y bien presentada permitió identificar los productos que prácticamente no han sido vendidos en el transcurso del año, en ese sentido se expuso para que salieran del mercado.

Se tomo en cuenta las ventas y reseñas resultantes de procesar y presentar los datos tabulados, facilitando la toma de decisiones.

En general, el buen manejo de los datos ayuda y da soporte a la toma de decisiones permitiendo tener un respaldo fidedigno.