

Universidad de San Carlos de Guatemala

Facultad de Ingeniería

Escuela de Ciencias y Sistemas

LABORATORIO INTRODUCCION A LA PROGRAMACION Y COMPUTACION 2 Sección D

OSCAR ROBERTO VELÁSQUEZ LEÓN

Practica auxiliar

Nombre	Carné	Sección
Brandon Orlando Seijas Morales	202010035	D

Link del repositorio

https://github.com/brandonseijas/IPC2_PracticaAuxiliar_2023.git

Top 10

Pantallazos de los resultados de los top 10 de los 3 niveles con mayor rentabilidad y el top 10 de productos con mayor valor del inventario

MarginLevel1



```
Menu para el primer proyecto de ipc2 en el 2023
1. Cargar informacion
2. Calcular Margenes y Valor del inventario
3. Mostrar el Top 10 dependiendo el valor a evaluar
4. Acerca del creador
5.Salida

Ingres opcion: 3
Dado las siguientes opciones:
->MarginLevel1:
->MarginLevel2:
->MarginLevel3:
->ValorInventario:

Ingrese que valor se utilizara como parametro para que se ordene la lista
y muestre el Top 10 MarginLevel1
->Top 10 del : MarginLevel1

1
->ItemCode: MALH0005
->QuantityOnHand: 6
->PriceLevel1: 67
->PriceLevel2: 66
->PriceLevel3: 65.25
->LastTotalUnitCost: 60.99
->MarginLevel1: 91.03658714236866
->MarginLevel2: 92.468898936789
->MarginLevel3: 93.4712643678161
->ValorInventario: 365.94

2
->ItemCode: MALH0005-1
->QuantityOnHand: 10
->PriceLevel1: 79.9
->PriceLevel2: 79.8
->PriceLevel3: 79.7
->LastTotalUnitCost: 73.99
->MarginLevel1: 92.6832548675847
->MarginLevel2: 92.71929282456104
->MarginLevel3: 92.8356336269786
->ValorInventario: 739.9

3
->ItemCode: MUELA-001
->QuantityOnHand: 10
->PriceLevel1: 69.8
->PriceLevel2: 69.6
->PriceLevel3: 69.1
->LastTotalUnitCost: 64
->MarginLevel1: 91.6965441268745
->MarginLevel2: 91.824029889676
->MarginLevel3: 92.6193921852388
->ValorInventario: 640.0

4
->ItemCode: MALH0005
->QuantityOnHand: 6
->PriceLevel1: 67
->PriceLevel2: 66
->PriceLevel3: 65.25
->LastTotalUnitCost: 60.99
->MarginLevel1: 91.03658714236866
->MarginLevel2: 92.468898936789
->MarginLevel3: 93.4712643678161
->ValorInventario: 365.94

5
->ItemCode: MALH0005
->QuantityOnHand: 26
->PriceLevel1: 78
->PriceLevel2: 77
->PriceLevel3: 75.5
->LastTotalUnitCost: 69.99
->MarginLevel1: 89.73876923876923
->MarginLevel2: 90.8861838618388
->MarginLevel3: 92.78198675496688
->ValorInventario: 1819.7399999999998

6
->ItemCode: MSE10001
->QuantityOnHand: 34
->PriceLevel1: 43.25
->PriceLevel2: 43
->PriceLevel3: 42.3
->LastTotalUnitCost: 38.79
->MarginLevel1: 89.6378612716763
->MarginLevel2: 90.209302325814
->MarginLevel3: 91.78212765957447
->ValorInventario: 1318.86

7
->ItemCode: DIFR2022
->QuantityOnHand: 188
->PriceLevel1: 39.75
->PriceLevel2: 39.5
->PriceLevel3: 39
->LastTotalUnitCost: 35.61
->MarginLevel1: 89.5949056037736
->MarginLevel2: 90.15189873417722
->MarginLevel3: 91.3876923876923
->ValorInventario: 3596.61

8
->ItemCode: SOWE6002
->QuantityOnHand: 0
->PriceLevel1: 16.1
->PriceLevel2: 16
->PriceLevel3: 15.9
->LastTotalUnitCost: 14.35
->MarginLevel1: 89.13843478260867
->MarginLevel2: 89.6875
->MarginLevel3: 90.25157232784402
->ValorInventario: 0.0

9
->ItemCode: GLUC2003
->QuantityOnHand: 54
->PriceLevel1: 57
->PriceLevel2: 57
->PriceLevel3: 56
->LastTotalUnitCost: 50.75
->MarginLevel1: 89.83588771929825
->MarginLevel2: 89.83588771929825
->MarginLevel3: 90.625
->ValorInventario: 2740.5

10
->ItemCode: GLUC2004
->QuantityOnHand: 0
->PriceLevel1: 57
->PriceLevel2: 57
->PriceLevel3: 56
->LastTotalUnitCost: 50.75
->MarginLevel1: 89.83588771929825
->MarginLevel2: 89.83588771929825
->MarginLevel3: 90.625
->ValorInventario: 0.0

Menu para el primer proyecto de ipc2 en el 2023
1. Cargar informacion
2. Calcular Margenes y Valor del inventario
3. Mostrar el Top 10 dependiendo el valor a evaluar
4. Acerca del creador
5.Salida

Ingres opcion: 4
```

MarginLevel2

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

~>MarginLevel1:
~>MarginLevel2:
~>MarginLevel3:
~>ValorInventario:

Ingrese que valor se utilizara como parametro para que se ordene la lista y muestre el Top 10MarginLevel2
~>Top 10 del : MarginLevel2

----- 1 -----
->ItemCode: DUCA1804
->QuantityOnHand: 14
->PriceLevel1: 54.5
->PriceLevel2: 52
->PriceLevel3: 52
->LastTotalUnitCost: 48.4
->MarginLevel1: 88.80733944954127
->MarginLevel2: 93.07692307692388
->MarginLevel3: 93.07692307692388
->ValorInventario: 677.6
----- 2 -----
->ItemCode: DUCA1820
->QuantityOnHand: 17
->PriceLevel1: 54.5
->PriceLevel2: 52
->PriceLevel3: 52
->LastTotalUnitCost: 48.4
->MarginLevel1: 88.80733944954127
->MarginLevel2: 93.07692307692388
->MarginLevel3: 93.07692307692388
->ValorInventario: 822.8
----- 3 -----
->ItemCode: MALH2003-1
->QuantityOnHand: 10
->PriceLevel1: 79.9
->PriceLevel2: 79.8
->PriceLevel3: 79.7
->LastTotalUnitCost: 73.99
->MarginLevel1: 92.60325406758447
->MarginLevel2: 92.71929824561404
->MarginLevel3: 92.83563362689786
->ValorInventario: 739.9
----- 4 -----
->ItemCode: MALH2006

+

...

×

Brandon Orlando Seijas Morales
202010035
LABORATORIO INTRODUCCION A LA
PROGRAMACION Y COMPUTACION 2
Sección D

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

~>ValorInventario: 739.9

----- 4 -----
->ItemCode: MALH2006
->QuantityOnHand: 6
->PriceLevel1: 67
->PriceLevel2: 66
->PriceLevel3: 65.25
->LastTotalUnitCost: 60.99
->MarginLevel1: 91.02985074626866
->MarginLevel2: 92.46908098989899
->MarginLevel3: 93.4712643678161
->ValorInventario: 365.94
----- 5 -----
->ItemCode: CARO-2002
->QuantityOnHand: 345
->PriceLevel1: 26
->PriceLevel2: 25.5
->PriceLevel3: 23.46
->MarginLevel1: 90.23076923076924
->MarginLevel2: 92.0
->MarginLevel3: 92.0
->ValorInventario: 8093.700000000001
----- 6 -----
->ItemCode: MUTEA-001
->QuantityOnHand: 10
->PriceLevel1: 69.6
->PriceLevel2: 69.6
->PriceLevel3: 69.1
->LastTotalUnitCost: 64
->MarginLevel1: 91.69054441260745
->MarginLevel2: 91.95402298850576
->MarginLevel3: 92.6183921852368
->ValorInventario: 640.0
----- 7 -----
->ItemCode: DWQD2002-1
->QuantityOnHand: 24
->PriceLevel1: 2.25
->PriceLevel2: 2.1
->PriceLevel3: 2.1
->LastTotalUnitCost: 1.9300000000000002
->MarginLevel1: 85.77777777777779

+

...

×

Brandon Orlando Seijas Morales
202010035
LABORATORIO INTRODUCCION A LA
PROGRAMACION Y COMPUTACION 2
Sección D

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

~>QuantityOnHand: 24
~>PriceLevel1: 2.25
~>PriceLevel2: 2.1
~>PriceLevel3: 2.1
~>LastTotalUnitCost: 1.9300000000000002
~>MarginLevel1: 85.77777777777779
~>MarginLevel2: 91.90476190476191
~>MarginLevel3: 91.90476190476191
~>ValorInventario: 46.320000000000001
----- 8 -----
->ItemCode: SOQU1851
->QuantityOnHand: 73
->PriceLevel1: 29
->PriceLevel2: 28
->PriceLevel3: 28.5
->LastTotalUnitCost: 25.72
->MarginLevel1: 88.68965517241378
->MarginLevel2: 91.85714285714285
->MarginLevel3: 90.24561403508771
->ValorInventario: 1877.56
----- 9 -----
->ItemCode: COLAT-001
->QuantityOnHand: 102
->PriceLevel1: 23.95
->PriceLevel2: 22.95
->PriceLevel3: 22.5
->LastTotalUnitCost: 20.97
->MarginLevel1: 87.35741127348642
->MarginLevel2: 91.37254901960785
->MarginLevel3: 93.19999999999999
->ValorInventario: 2138.94
----- 10 -----
->ItemCode: CANO1001
->QuantityOnHand: 4
->PriceLevel1: 29.95
->PriceLevel2: 28.5
->PriceLevel3: 28.3
->LastTotalUnitCost: 26
->MarginLevel1: 86.81135225375625
->MarginLevel2: 91.22807017543859
->MarginLevel3: 91.87279151943463
->ValorInventario: 104.0

+

...

×

Brandon Orlando Seijas Morales
202010035
LABORATORIO INTRODUCCION A LA
PROGRAMACION Y COMPUTACION 2
Sección D

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE

~>LastTotalUnitCost: 25.72
~>MarginLevel1: 88.68965517241378
~>MarginLevel2: 91.85714285714285
~>MarginLevel3: 90.24561403508771
~>ValorInventario: 1877.56
----- 9 -----
->ItemCode: COLAT-001
->QuantityOnHand: 102
->PriceLevel1: 23.95
->PriceLevel2: 22.95
->PriceLevel3: 22.5
->LastTotalUnitCost: 20.97
->MarginLevel1: 87.35741127348642
->MarginLevel2: 91.37254901960785
->MarginLevel3: 93.19999999999999
->ValorInventario: 2138.94
----- 10 -----
->ItemCode: CANO1001
->QuantityOnHand: 4
->PriceLevel1: 29.95
->PriceLevel2: 28.5
->PriceLevel3: 28.3
->LastTotalUnitCost: 26
->MarginLevel1: 86.81135225375625
->MarginLevel2: 91.22807017543859
->MarginLevel3: 91.87279151943463
->ValorInventario: 104.0

+

...

×

Brandon Orlando Seijas Morales
202010035
LABORATORIO INTRODUCCION A LA
PROGRAMACION Y COMPUTACION 2
Sección D

Menú para el primer proyecto de ipc2 en el 2023
1. Cargar información
2. Calcular Margenes y Valor del inventario
3. Mostrar el Top 10 dependiendo el valor a evaluar
4. Acerca del creador
5.Salida

Ingres opción: []

MarginLevel3

```
PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
| 4. Acerca del creador
| 5.Salida
-----
Ingres opción: 3
Dado las siguientes opciones:
->MarginLevel1:
->MarginLevel2:
->MarginLevel3:
->ValorInventario:
-----
Ingrese que valor se utilizara como parametro para que se ordene la lista
y muestre el Top 10MarginLevel3
->Top 10 del : MarginLevel3
-----
1
->ItemCode: JELLY1001
->QuantityOnHand: 15
->PriceLevel1: 47.25
->PriceLevel2: 45.4
->PriceLevel3: 39.65
->LastTotalUnitCost: 38
->MarginLevel1: 80.42328042328042
->MarginLevel2: 83.70044652863437
->MarginLevel3: 75.63650764186634
->ValorInventario: 570.0
-----
2
->ItemCode: DM0025001-1
->QuantityOnHand: 26
->PriceLevel1: 5
->PriceLevel2: 4.8
->PriceLevel3: 4.4
->LastTotalUnitCost: 4.14
->MarginLevel1: 82.8
->MarginLevel2: 86.25
->MarginLevel3: 94.09090909090908
->ValorInventario: 107.63999999999999
-----
3
->ItemCode: MALH2006
->QuantityOnHand: 6
->PriceLevel1: 67
->PriceLevel2: 66
->PriceLevel3: 65.25
->LastTotalUnitCost: 60.99
->MarginLevel1: 91.02985074626866
->MarginLevel2: 92.4090909090909
-----
4
->ItemCode: COLAT-001
->QuantityOnHand: 102
->PriceLevel1: 23.95
->PriceLevel2: 22.95
->PriceLevel3: 22.5
->LastTotalUnitCost: 20.97
->MarginLevel1: 87.55741127348642
->MarginLevel2: 91.37254901060785
->MarginLevel3: 93.19999999999999
->ValorInventario: 2138.94
-----
5
->ItemCode: GON30810
->QuantityOnHand: 576
->PriceLevel1: 47.95
->PriceLevel2: 45
->PriceLevel3: 44
->LastTotalUnitCost: 41
->MarginLevel1: 85.50573514077163
->MarginLevel2: 91.11111111111111
->MarginLevel3: 93.18181818181817
->ValorInventario: 23616.0
-----
6
->ItemCode: DUCA1020
->QuantityOnHand: 17
->PriceLevel1: 54.5
->PriceLevel2: 52
->PriceLevel3: 52
->LastTotalUnitCost: 48.4
->MarginLevel1: 88.80733944954127
->MarginLevel2: 93.07692307692308
->MarginLevel3: 93.07692307692308
->ValorInventario: 822.8
-----
7
->ItemCode: DUCA1004
->QuantityOnHand: 14
->PriceLevel1: 54.5
->PriceLevel2: 52
->PriceLevel3: 52
->LastTotalUnitCost: 48.4
->MarginLevel1: 88.80733944954127
->MarginLevel2: 93.07692307692308
->MarginLevel3: 93.07692307692308
->ValorInventario: 677.6
-----
8
->ItemCode: MALH2003-1
->QuantityOnHand: 10
->PriceLevel1: 79.9
->PriceLevel2: 79.8
->PriceLevel3: 79.7
->LastTotalUnitCost: 73.99
->MarginLevel1: 92.60325406758447
->MarginLevel2: 92.71929824561404
->MarginLevel3: 92.83563362609786
->ValorInventario: 739.9
-----
9
->ItemCode: MALH2005
->QuantityOnHand: 26
->PriceLevel1: 78
->PriceLevel2: 77
->PriceLevel3: 75.5
->LastTotalUnitCost: 69.99
->MarginLevel1: 89.73076923076923
->MarginLevel2: 90.89610389610388
->MarginLevel3: 92.70198675496688
->ValorInventario: 1819.7399999999998
-----
10
->ItemCode: NUTELA-001
->QuantityOnHand: 10
->PriceLevel1: 69.8
->PriceLevel2: 69.6
->PriceLevel3: 69.1
->LastTotalUnitCost: 64
->MarginLevel1: 91.69054441260745
->MarginLevel2: 91.9540229850576
->MarginLevel3: 92.6193921892388
->ValorInventario: 640.0
-----
Menú para el primer proyecto de ipc2 en el 2023
1. Cargar información
2. Calcular Margenes y Valor del inventario
3. Mostrar el Top 10 dependiendo el valor a evaluar
4. Acerca del creador
5.Salida
-----
Ingres opción: []
```

ValorInventario

PROBLEMS

OUTPUT

TERMINAL

DEBUG CONSOLE

```
-->MarginLevel2: 91.9540298850576
-->MarginLevel3: 92.6193921852388
-->ValorInventario: 640.0

-----
| Menú para el primer proyecto de ipc2 en el 2023
| 1. Cargar información
| 2. Calcular Margenes y Valor del inventario
| 3. Mostrar el Top 10 dependiendo el valor a evaluar
| 4. Acerca del creador
| 5.Salida
|
-----

Ingres opción: 3
Dado las siguientes opciones:
-->MarginLevel1:
-->MarginLevel2:
-->MarginLevel3:
-->ValorInventario:

-----

Ingrese que valor se utilizara como parametro para que se ordene la lista
y muestre el Top 10ValorInventario
-->Top 10 del : ValorInventario

----- 1 -----
->ItemCode: DLX1016
->QuantityOnHand: 26
->PriceLevel1: 47
->PriceLevel2: 46
->PriceLevel3: 45.25
->LastTotalUnitCost: 37.61
->MarginLevel1: 80.02127659574668
->MarginLevel2: 81.76086956521739
->MarginLevel3: 83.11602209944752
->ValorInventario: 977.86

----- 2 -----
->ItemCode: DIAN3002
->QuantityOnHand: 25
->PriceLevel1: 45
->PriceLevel2: 44
->PriceLevel3: 42.95
->LastTotalUnitCost: 39
->MarginLevel1: 86.66666666666667
->MarginLevel2: 88.63636363636364
->MarginLevel3: 90.80325960419091
->ValorInventario: 975.0

----- 3 -----
->ItemCode: LID01007
->QuantityOnHand: 39
->PriceLevel1: 29.85
->PriceLevel2: 29
->PriceLevel3: 28.5
->LastTotalUnitCost: 25
->MarginLevel1: 83.7500380234506
->MarginLevel2: 86.20689655172413
->MarginLevel3: 87.71929824561403
->ValorInventario: 975.0

----- 4 -----
->ItemCode: BOCA1007
->QuantityOnHand: 46
->PriceLevel1: 25.7
->PriceLevel2: 24.5
->PriceLevel3: 24
->LastTotalUnitCost: 21.15
->MarginLevel1: 82.29571894435798
->MarginLevel2: 86.3265306122449
->MarginLevel3: 88.125
->ValorInventario: 972.9

----- 5 -----
->ItemCode: DLX-36-0074
->QuantityOnHand: 9
->PriceLevel1: 17.85
->PriceLevel2: 17.2
->PriceLevel3: 16.4
->LastTotalUnitCost: 10.78
->MarginLevel1: 60.392156827451
->MarginLevel2: 62.67441860465116
->MarginLevel3: 65.73170731707317
->ValorInventario: 97.02

----- 6 -----
->ItemCode: CNC-1001
->QuantityOnHand: 47
->PriceLevel1: 33.9
->PriceLevel2: 33
->PriceLevel3: 24.5
->LastTotalUnitCost: 20.44
->MarginLevel1: 60.29498525073747
->MarginLevel2: 61.93939393939394
->MarginLevel3: 63.42857142857143
->ValorInventario: 960.6800000000001

----- 7 -----
->ItemCode: DM005007-1
->QuantityOnHand: 57
->PriceLevel1: 2.5
->PriceLevel2: 2.4
->PriceLevel3: 2.2
->LastTotalUnitCost: 1.69
->MarginLevel1: 67.6
->MarginLevel2: 70.41666666666667
->MarginLevel3: 76.81818181818181
->ValorInventario: 96.33

----- 8 -----
->ItemCode: TABO-011
->QuantityOnHand: 4
->PriceLevel1: 33.6
->PriceLevel2: 31.4
->PriceLevel3: 30.2
->LastTotalUnitCost: 24.06
->MarginLevel1: 71.60714285714285
->MarginLevel2: 76.62420382165604
->MarginLevel3: 79.66887417218544
->ValorInventario: 96.33

----- 9 -----
->ItemCode: FARMAC6000
->QuantityOnHand: 4
->PriceLevel1: 32
->PriceLevel2: 31
->PriceLevel3: 30.75
->LastTotalUnitCost: 24
->MarginLevel1: 75.0
->MarginLevel2: 77.41935483870968
->MarginLevel3: 78.84570848780488
->ValorInventario: 96.0

----- 10 -----
->ItemCode: B662005
->QuantityOnHand: 287
->PriceLevel1: 46.95
->PriceLevel2: 45.95
->PriceLevel3: 42.85
->LastTotalUnitCost: 33.39
->MarginLevel1: 71.1152108626198
->MarginLevel2: 72.65594124047879
->MarginLevel3: 77.9229216452743
->ValorInventario: 9582.93

-----

Menú para el primer proyecto de ipc2 en el 2023
1. Cargar información
2. Calcular Margenes y Valor del inventario
3. Mostrar el Top 10 dependiendo el valor a evaluar
4. Acerca del creador
5.Salida
```

+

Brandon Orlando Seijas Morales
202010035
LABORATORIO INTRODUCCION A LA
PROGRAMACION Y COMPUTACION 2
Sección D

×

PROBLEMS

OUTPUT

TERMINAL

DEBUG CONSOLE

```
-->ValorInventario: 977.86

----- 2 -----
->ItemCode: DIAN3002
->QuantityOnHand: 25
->PriceLevel1: 45
->PriceLevel2: 44
->PriceLevel3: 42.95
->LastTotalUnitCost: 39
->MarginLevel1: 86.66666666666667
->MarginLevel2: 88.63636363636364
->MarginLevel3: 90.80325960419091
->ValorInventario: 975.0

----- 3 -----
->ItemCode: LID01007
->QuantityOnHand: 39
->PriceLevel1: 29.85
->PriceLevel2: 29
->PriceLevel3: 28.5
->LastTotalUnitCost: 25
->MarginLevel1: 83.7500380234506
->MarginLevel2: 86.20689655172413
->MarginLevel3: 87.71929824561403
->ValorInventario: 975.0

----- 4 -----
->ItemCode: BOCA1007
->QuantityOnHand: 46
->PriceLevel1: 25.7
->PriceLevel2: 24.5
->PriceLevel3: 24
->LastTotalUnitCost: 21.15
->MarginLevel1: 82.29571894435798
->MarginLevel2: 86.3265306122449
->MarginLevel3: 88.125
->ValorInventario: 972.9

----- 5 -----
->ItemCode: DLX-36-0074
->QuantityOnHand: 9
->PriceLevel1: 17.85
->PriceLevel2: 17.2
->PriceLevel3: 16.4
->LastTotalUnitCost: 10.78
->MarginLevel1: 60.392156827451
->MarginLevel2: 62.67441860465116
->MarginLevel3: 65.73170731707317
->ValorInventario: 97.02

----- 6 -----
->ItemCode: CNC-1001
->QuantityOnHand: 47
->PriceLevel1: 33.9
->PriceLevel2: 33
->PriceLevel3: 24.5
->LastTotalUnitCost: 20.44
->MarginLevel1: 60.29498525073747
->MarginLevel2: 61.93939393939394
->MarginLevel3: 63.42857142857143
->ValorInventario: 960.6800000000001

----- 7 -----
->ItemCode: DM005007-1
->QuantityOnHand: 57
->PriceLevel1: 2.5
->PriceLevel2: 2.4
->PriceLevel3: 2.2
->LastTotalUnitCost: 1.69
->MarginLevel1: 67.6
->MarginLevel2: 70.41666666666667
->MarginLevel3: 76.81818181818181
->ValorInventario: 96.33

----- 8 -----
->ItemCode: TABO-011
->QuantityOnHand: 4
->PriceLevel1: 33.6
->PriceLevel2: 31.4
->PriceLevel3: 30.2
->LastTotalUnitCost: 24.06
->MarginLevel1: 71.60714285714285
->MarginLevel2: 76.62420382165604
->MarginLevel3: 79.66887417218544
->ValorInventario: 96.33

----- 9 -----
->ItemCode: FARMAC6000
->QuantityOnHand: 4
->PriceLevel1: 32
->PriceLevel2: 31
->PriceLevel3: 30.75
->LastTotalUnitCost: 24
->MarginLevel1: 75.0
->MarginLevel2: 77.41935483870968
->MarginLevel3: 78.84570848780488
->ValorInventario: 96.0

----- 10 -----
->ItemCode: B662005
->QuantityOnHand: 287
->PriceLevel1: 46.95
->PriceLevel2: 45.95
->PriceLevel3: 42.85
->LastTotalUnitCost: 33.39
->MarginLevel1: 71.1152108626198
->MarginLevel2: 72.65594124047879
->MarginLevel3: 77.9229216452743
->ValorInventario: 9582.93

-----

Menú para el primer proyecto de ipc2 en el 2023
1. Cargar información
2. Calcular Margenes y Valor del inventario
3. Mostrar el Top 10 dependiendo el valor a evaluar
4. Acerca del creador
5.Salida
```

+

Brandon Orlando Seijas Morales
202010035
LABORATORIO INTRODUCCION A LA
PROGRAMACION Y COMPUTACION 2
Sección D

×

- Código de la lista simplemente enlazada

```

class nodoempresa():

    def __init__(self):

        self.ItemCode = ""

        self.QuantityOnHand = ""

        self.PriceLevel1 = ""

        self.PriceLevel2 = ""

        self.PriceLevel3 = ""

        self.LastTotalUnitCost = ""

        self.siguiente = None

        self.antes = None

class Lista:

    def __init__(self):

        self.primeros = None

        self.ultimo = None

        self.size = 0

    def vacia(self):

        return self.primeros == self.ultimo == None

    def agregarNodo(self, nodoempresa):

        nuevamuestra = nodoempresa

        if self.vacia():

            self.primeros = self.ultimo = nuevamuestra

        elif self.primeros == self.ultimo:

            self.ultimo = nuevamuestra

            self.primeros.siguiente = self.ultimo

            self.ultimo.antes = self.primeros

        else:

            self.ultimo.siguiente = nuevamuestra

            nuevamuestra.antes = self.ultimo

            self.ultimo = nuevamuestra

        self.size = self.size + 1

    def mostrardato(self):

        actual = self.primeros

        while (actual != None):

            print(f"    ->ItemCode: ",str(actual.ItemCode))

```

```

print(f" ->QuantityOnHand: ",str(actual.QuantityOnHand))
print(f" ->PriceLevel1: ", str(actual.PriceLevel1))
print(f" ->PriceLevel2: ",str(actual.PriceLevel2))
print(f" ->PriceLevel3: ",str(actual.PriceLevel3))
print(f" ->LastTotalUnitCost: ", str(actual.LastTotalUnitCost))
print(" -----")
actual = actual.siguiente

```

```

class nodoempresa():
    def __init__(self):
        self.ItemCode = ""
        self.QuantityOnHand = ""
        self.PriceLevel1 = ""
        self.PriceLevel2 = ""
        self.PriceLevel3 = ""
        self.LastTotalUnitCost = ""
        self.siguiente = None
        self.antes = None

class Lista:
    def __init__(self):
        self.primeros = None
        self.ultimo = None
        self.size = 0

    def vacia(self):
        return self.primeros == self.ultimo == None

    def agregarNodo(self, nodoempresa):
        nuevamuestra = nodoempresa
        if self.vacia():
            self.primeros = self.ultimo = nuevamuestra
        elif self.primeros == self.ultimo:
            self.ultimo = nuevamuestra
            self.primeros.siguiente = self.ultimo
            self.ultimo.antes = self.primeros
        else:
            self.ultimo.siguiente = nuevamuestra
            nuevamuestra.antes = self.ultimo
            self.ultimo = nuevamuestra
        self.size = self.size +1

    def mostrardato(self):
        actual = self.primeros
        while (actual != None):
            print(f" ->ItemCode: ",str(actual.ItemCode))
            print(f" ->QuantityOnHand: ",str(actual.QuantityOnHand))
            print(f" ->PriceLevel1: ", str(actual.PriceLevel1))
            print(f" ->PriceLevel2: ",str(actual.PriceLevel2))
            print(f" ->PriceLevel3: ",str(actual.PriceLevel3))
            print(f" ->LastTotalUnitCost: ", str(actual.LastTotalUnitCost))
            print(" -----")
            actual = actual.siguiente

```

- Código para ordenar el arreglo
- class Nodo:

```

def __init__(self, data):

    self.data = data

    self.next = None

```

```
self.prev = None
```

```
class Datos:
```

```
def __init__(self, ItemCode, QuantityOnHand, PriceLevel1, PriceLevel2, PriceLevel3,  
LastTotalUnitCost, MarginLevel1,MarginLevel2,MarginLevel3,ValorInventario):
```

```
    self.ItemCode = ItemCode
```

```
    self.QuantityOnHand = QuantityOnHand
```

```
    self.PriceLevel1 = PriceLevel1
```

```
    self.PriceLevel2 = PriceLevel2
```

```
    self.PriceLevel3 = PriceLevel3
```

```
    self.LastTotalUnitCost = LastTotalUnitCost
```

```
    self.MarginLevel1 = MarginLevel1
```

```
    self.MarginLevel2 = MarginLevel2
```

```
    self.MarginLevel3 = MarginLevel3
```

```
    self.ValorInventario = ValorInventario
```

```
class ListaDoble:
```

```
def __init__(self):
```

```
    self.head = None
```

```
def agregar(self, data):
```

```
    nuevo_nodo = Nodo(data)
```

```
    if self.head is None:
```

```
        self.head = nuevo_nodo
```

```
    else:
```

```
        actual = self.head
```

```
        while actual.next is not None:
```

```
            actual = actual.next
```

```
        actual.next = nuevo_nodo
```

```
        nuevo_nodo.prev = actual
```



```

def mostrar(self,respuesta):

    i=0

    if self.head is None:

        print("La lista está vacía")

    else:

        actual = self.head

        print(f" ->Top 10 del : {respuesta}")

        while actual is not None:

            if (i<10):

                print(f"----- {i+1}")

                print(f" ->ItemCode: ",str(actual.data.ItemCode))

                print(f" ->QuantityOnHand: ",str(actual.data.QuantityOnHand))

                print(f" ->PriceLevel1: ", str(actual.data.PriceLevel1))

                print(f" ->PriceLevel2: ",str(actual.data.PriceLevel2))

                print(f" ->PriceLevel3: ",str(actual.data.PriceLevel3))

                print(f" ->LastTotalUnitCost: ", str(actual.data.LastTotalUnitCost))

                print(f" ->MarginLevel1: ", str(actual.data.MarginLevel1))

                print(f" ->MarginLevel2: ", str(actual.data.MarginLevel2))

                print(f" ->MarginLevel3: ", str(actual.data.MarginLevel3))

                print(f" ->ValorInventario: ", str(actual.data.ValorInventario))

                print(" -----")

                i = i+1

            actual = actual.next

```

```

def ordenar_burbuja(self, valor):

```

```

    actual = self.head

    while actual is not None:

        siguiente = actual.next

        while siguiente is not None:

```

```
if getattr(actual.data, valor) < getattr(siguiete.data, valor):  
    actual.data, siguiete.data = siguiete.data, actual.data  
    siguiete = siguiete.next  
actual = actual.next
```

```
def ordenar(self, respuesta):  
    actual = self.primerero  
    MarginLevel1= 0  
    MarginLevel2= 0  
    MarginLevel3= 0  
    ValorInventario= 0  
    Lista = ListaDoble()  
    while (actual != None):  
        MarginLevel1 = ((float(actual.LastTotalUnitCost))/(float(actual.PriceLevel1)))*100  
        MarginLevel2 = ((float(actual.LastTotalUnitCost))/(float(actual.PriceLevel2)))*100  
        MarginLevel3 = ((float(actual.LastTotalUnitCost))/(float(actual.PriceLevel3)))*100  
        ValorInventario = (float(actual.LastTotalUnitCost))*(float(actual.QuantityOnHand))  
  
    Lista.agregar(Datos(str(actual.ItemCode),str(actual.QuantityOnHand),str(actual.PriceLevel1),  
str(actual.PriceLevel2),str(actual.PriceLevel3),str(actual.LastTotalUnitCost)  
,str(MarginLevel1),str(MarginLevel2),str(MarginLevel3),str(ValorInventario)))  
  
    actual = actual.siguiete  
  
    if ((respuesta == "MarginLevel1")or(respuesta == "MarginLevel2")or(respuesta ==  
"MarginLevel3")or(respuesta == "ValorInventario")):  
        Lista.ordenar_burbuja(respuesta)  
        Lista.mostrar(respuesta)
```

```

56
57 def ordenar_burbuja(self, valor):
58     actual = self.head
59     while actual is not None:
60         siguiente = actual.next
61         while siguiente is not None:
62             if getattr(actual.data, valor) < getattr(siguiente.data, valor):
63                 actual.data, siguiente.data = siguiente.data, actual.data
64                 siguiente = siguiente.next
65                 actual = actual.next
66
67 def ordenar(self, respuesta):
68     actual = self.primerNodo
69     MarginLevel1= 0
70     MarginLevel2= 0
71     MarginLevel3= 0
72     ValorInventario= 0
73     Lista = ListDoble()
74     while (actual != None):
75         MarginLevel1 = ((float(actual.LastTotalUnitCost))/(float(actual.PriceLevel1)))*100
76         MarginLevel2 = ((float(actual.LastTotalUnitCost))/(float(actual.PriceLevel2)))*100
77         MarginLevel3 = ((float(actual.LastTotalUnitCost))/(float(actual.PriceLevel3)))*100
78         ValorInventario = (float(actual.LastTotalUnitCost))*(float(actual.QuantityOnHand))
79         Lista.agregar(Datos(str(actual.ItemCode),str(actual.QuantityOnHand),str(actual.PriceLevel1), str(actual.PriceLevel2),str(actual.PriceLevel3),str(actual.LastTotalUnitCost)
80         ,str(MarginLevel1),str(MarginLevel2),str(MarginLevel3),str(ValorInventario)))
81         actual = actual.siguiente
82     if ((respuesta == "MarginLevel1")or(respuesta == "MarginLevel2")or(respuesta == "MarginLevel3")or(respuesta == "ValorInventario")):
83         Lista.ordenar_burbuja(respuesta)
84         Lista.mostrar(respuesta)

```

Código para ordenar el arreglo depende de una llave se puede ordenar y obtener el top 10 dependiendo si elige cual de los 3 niveles con mayor rentabilidad o los productos con mayor valor de inventario desea ordenar.

- Código para realizar los cálculos de los márgenes

class Nodo:

```

def __init__(self, data):
    self.data = data
    self.next = None
    self.prev = None

```

class Datos:

```

def __init__(self, ItemCode, QuantityOnHand, PriceLevel1, PriceLevel2, PriceLevel3,
LastTotalUnitCost, MarginLevel1,MarginLevel2,MarginLevel3,ValorInventario):

```

```

    self.ItemCode = ItemCode
    self.QuantityOnHand = QuantityOnHand
    self.PriceLevel1 = PriceLevel1
    self.PriceLevel2 = PriceLevel2
    self.PriceLevel3 = PriceLevel3
    self.LastTotalUnitCost = LastTotalUnitCost
    self.MarginLevel1 = MarginLevel1
    self.MarginLevel2 = MarginLevel2
    self.MarginLevel3 = MarginLevel3
    self.ValorInventario = ValorInventario

```

class ListDoble:

```

def __init__(self):
    self.head = None

```

```

def agregar(self, data):

```

```

    nuevo_nodo = Nodo(data)
    if self.head is None:
        self.head = nuevo_nodo
    else:
        actual = self.head
        while actual.next is not None:
            actual = actual.next

```

```

actual.next = nuevo_nodo
nuevo_nodo.prev = actual

```

```

def realizarCal(self):
    actual = self.primerO
    MarginLevel1= 0
    MarginLevel2= 0
    MarginLevel3= 0
    ValorInventario= 0
    Lista = ListaDoble()
    while (actual != None):
        MarginLevel1 = ((float(actual.LastTotalUnitCost))/(float(actual.PriceLevel1)))*100
        MarginLevel2 = ((float(actual.LastTotalUnitCost))/(float(actual.PriceLevel2)))*100
        MarginLevel3 = ((float(actual.LastTotalUnitCost))/(float(actual.PriceLevel3)))*100
        ValorInventario =
(float(actual.LastTotalUnitCost))*(float(actual.QuantityOnHand))
        print(f"  ->ItemCode: ",str(actual.ItemCode))
        print(f"  ->QuantityOnHand: ",str(actual.QuantityOnHand))
        print(f"  ->PriceLevel1: ", str(actual.PriceLevel1))
        print(f"  ->PriceLevel2: ",str(actual.PriceLevel2))
        print(f"  ->PriceLevel3: ",str(actual.PriceLevel3))
        print(f"  ->LastTotalUnitCost: ", str(actual.LastTotalUnitCost))
        print(f"  ->MarginLevel1: ", str(MarginLevel1))
        print(f"  ->MarginLevel2: ", str(MarginLevel2))
        print(f"  ->MarginLevel3: ", str(MarginLevel3))
        print(f"  ->ValorInventario: ", str(ValorInventario))
        print("  -----")

Lista.agregar(Datos(str(actual.ItemCode),str(actual.QuantityOnHand),str(actual.PriceLevel
1), str(actual.PriceLevel2),str(actual.PriceLevel3),str(actual.LastTotalUnitCost)
,str(MarginLevel1),str(MarginLevel2),str(MarginLevel3),str(ValorInventario))))
actual = actual.siguiente

```

```

33     def realizarCal(self):
34         actual = self.primerO
35         MarginLevel1= 0
36         MarginLevel2= 0
37         MarginLevel3= 0
38         ValorInventario= 0
39         Lista = ListaDoble()
40         while (actual != None):
41             MarginLevel1 = ((float(actual.LastTotalUnitCost))/(float(actual.PriceLevel1)))*100
42             MarginLevel2 = ((float(actual.LastTotalUnitCost))/(float(actual.PriceLevel2)))*100
43             MarginLevel3 = ((float(actual.LastTotalUnitCost))/(float(actual.PriceLevel3)))*100
44             ValorInventario = (float(actual.LastTotalUnitCost))*(float(actual.QuantityOnHand))
45             print(f"  ->ItemCode: ",str(actual.ItemCode))
46             print(f"  ->QuantityOnHand: ",str(actual.QuantityOnHand))
47             print(f"  ->PriceLevel1: ", str(actual.PriceLevel1))
48             print(f"  ->PriceLevel2: ",str(actual.PriceLevel2))
49             print(f"  ->PriceLevel3: ",str(actual.PriceLevel3))
50             print(f"  ->LastTotalUnitCost: ", str(actual.LastTotalUnitCost))
51             print(f"  ->MarginLevel1: ", str(MarginLevel1))
52             print(f"  ->MarginLevel2: ", str(MarginLevel2))
53             print(f"  ->MarginLevel3: ", str(MarginLevel3))
54             print(f"  ->ValorInventario: ", str(ValorInventario))
55             print("  -----")
56             Lista.agregar(Datos(str(actual.ItemCode),str(actual.QuantityOnHand),str(actual.PriceLevel1), str(actual.PriceLevel2),str(actual.PriceLevel3),str(actual.LastTotalUnitCost) ,str(
MarginLevel1),str(MarginLevel2),str(MarginLevel3),str(ValorInventario))))
57             actual = actual.siguiente

```

Código para realizar los cálculos de los márgenes