

UNIVERSIDAD AUTONOMA DE NUEVO LEON.

FACULTAD DE INGENIERIA MECÁNICA Y ELECTRICA.

TAREA 4: ALGORITMO HEURISTICO PARA PROBLEMA DE
MOCHILA

PRESENTADO POR:

JESUS JAVIER MORENO VAZQUEZ 1619830

HORA: V4-V6

PROFESOR: DR. MARIA ANGELICA SALAZAR

SAN NICOLAS DE LOS GARZA, NUEVO LEON, A 5 DE FEBRERO DEL 2016

DESCRIPCION GENERAL DEL PROBLEMA

Se tiene una lista de 'n' objetos, cada uno tiene un peso y un beneficio respectivamente, también se tiene una "mochila" la cual tiene una capacidad 'K_max'. El objetivo del problema es introducir tantos objetos a la mochila buscando aumentar al máximo el beneficio sin exceder la capacidad. Se presentan 9 casos para testear el algoritmo desarrollado:

CASO CLASE: Propuesto en Clase

Capacidad =	2000	
Objeto	Peso	Beneficio
Obj_1	100	15
Obj_2	2	10
Obj_3	150	12
Obj_4	56	20
Obj_5	2000	8
Obj_6	10	2
Obj_7	1000	6
Obj_8	50	20
Obj_9	150	6
Obj_10	5	8
Obj_11	10	6
Obj_12	300	8
Obj_13	500	10
Obj_14	100	1
Obj_15	20	5
Obj_16	15	1
Obj_17	30	6
Obj_18	15	1
Obj_19	9	2
Obj_20	150	6

CASO 1: 10 Objetos con una capacidad máxima de 165

Capacidad =	165	
Objeto	Peso	Beneficio
Obj_1	23	92
Obj_2	31	57
Obj_3	29	49
Obj_4	44	68
Obj_5	53	60
Obj_6	38	43
Obj_7	63	67
Obj_8	85	84
Obj_9	89	87
Obj_10	82	72

CASO 2: 5 objetos y una capacidad 26

Capacidad =	26	
Objeto	Peso	Beneficio
Obj_1	12	24
Obj_2	7	13
Obj_3	11	23
Obj_4	8	15
Obj_5	9	16

CASO 3: 6 objetos y una capacidad de 190

Capacidad =	190	
Objeto	Peso	Beneficio
Obj_1	56	50
Obj_2	59	50
Obj_3	80	64
Obj_4	64	46
Obj_5	75	50
Obj_6	17	5

CASO 4: 7 objetos y una capacidad de 50

Capacidad =	50	
Objeto	Peso	Beneficio
Obj_1	31	70
Obj_2	10	20
Obj_3	20	39
Obj_4	19	37
Obj_5	4	7
Obj_6	3	5
Obj_7	6	10

CASO 5: 8 objetos y una capacidad de 104

Capacidad =	104	
Objeto	Peso	Beneficio
Obj_1	25	350
Obj_2	35	400
Obj_3	45	450
Obj_4	5	20
Obj_5	25	70
Obj_6	3	8
Obj_7	2	5
Obj_8	2	5

CASO 6: 7 Objetos con una capacidad de 170

Capacidad =	170	
Objeto	Peso	Beneficio
Obj_1	41	442
Obj_2	50	525
Obj_3	49	511
Obj_4	59	593
Obj_5	55	546
Obj_6	57	564
Obj_7	60	617

CASO 7: 15 Objetos con una capacidad de 750

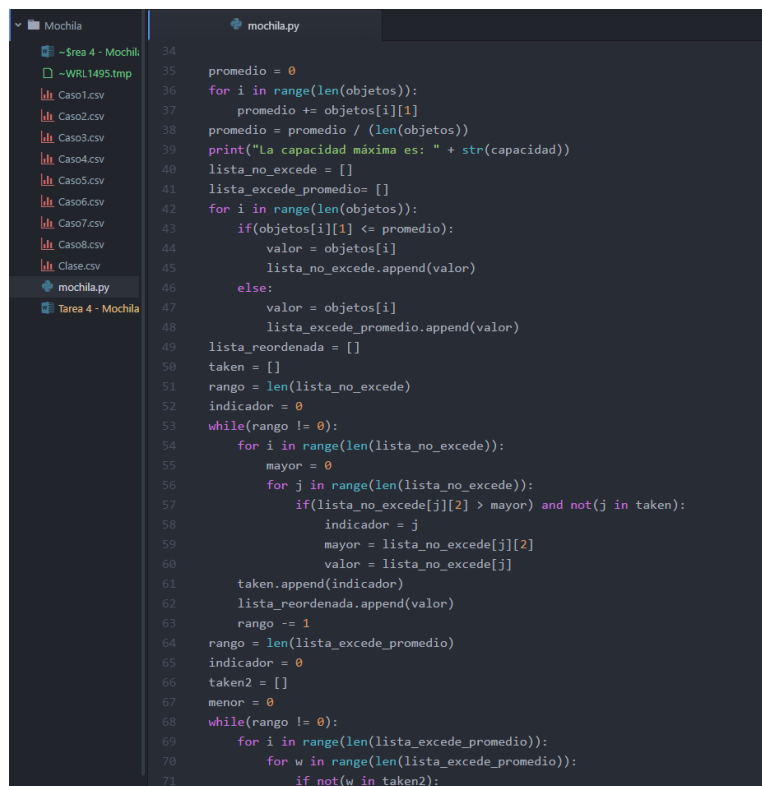
Capacidad =	750	
Objeto	Peso	Beneficio
Obj_1	70	135
Obj_2	73	139
Obj_3	77	149
Obj_4	80	150
Obj_5	82	156
Obj_6	87	163
Obj_7	90	173
Obj_8	94	184
Obj_9	98	192
Obj_10	106	201
Obj_11	110	210
Obj_12	113	214
Obj_13	115	221
Obj_14	118	229
Obj_15	120	240

CASO 8: 24 Objetos y una capacidad de 6404180

Capacidad =	6404180	
Objeto	Peso	Beneficio
Obj_1	382745	825594
Obj_2	799601	1677009
Obj_3	909247	1676628
Obj_4	729069	1523970
Obj_5	467902	943972
Obj_6	44328	97426
Obj_7	34610	69666
Obj_8	698150	1296457
Obj_9	823460	1679693
Obj_10	903959	1902996
Obj_11	853665	1844992
Obj_12	551830	1049289
Obj_13	610856	1252836
Obj_14	670702	1319836
Obj_15	488960	953277
Obj_16	951111	2067538
Obj_17	323046	675367
Obj_18	446298	853655
Obj_19	931161	1826027
Obj_20	31385	65731
Obj_21	496951	901489
Obj_22	264724	577243
Obj_23	224916	466257
Obj_24	169684	369261

ALGORITMO PROPUESTO:

- 1.- Lectura de Datos
- 2.- Sacar el promedio de los pesos.
- 3.- Ordenar objetos en una nueva lista respecto al beneficio de manera descendiente sin tomar en cuenta aquellos objetos cuyos pesos excedan el promedio.
- 4.- Agregar los objetos restantes (aquellos que exceden el promedio) pero ahora respecto al peso de manera ascendente.
- 5.- Seleccionar el primer elemento que no haya sido seleccionado.
- 6.- Asignar dicho elemento si y solo si no excede la capacidad
- 7.- Si aún hay espacio, sigue introduciendo objetos (paso 5), si no, termina la ejecución.
- 8.- Reportar resultados.



```
34 promedio = 0
35 for i in range(len(objetos)):
36     promedio += objetos[i][1]
37 promedio = promedio / (len(objetos))
38 print("La capacidad máxima es: " + str(capacidad))
39 lista_no_excede = []
40 lista_excede_promedio = []
41 for i in range(len(objetos)):
42     if(objetos[i][1] <= promedio):
43         valor = objetos[i]
44         lista_no_excede.append(valor)
45     else:
46         valor = objetos[i]
47         lista_excede_promedio.append(valor)
48 lista_reordenada = []
49 taken = []
50 rango = len(lista_no_excede)
51 indicador = 0
52 while(rango != 0):
53     for i in range(len(lista_no_excede)):
54         mayor = 0
55         for j in range(len(lista_no_excede)):
56             if(lista_no_excede[j][2] > mayor and not(j in taken):
57                 indicador = j
58                 mayor = lista_no_excede[j][2]
59                 valor = lista_no_excede[j]
60         taken.append(indicador)
61         lista_reordenada.append(valor)
62         rango -= 1
63 rango = len(lista_excede_promedio)
64 indicador = 0
65 taken2 = []
66 menor = 0
67 while(rango != 0):
68     for i in range(len(lista_excede_promedio)):
69         for w in range(len(lista_excede_promedio)):
70             if not(w in taken2):
```

RESULTADOS

Clase:

```
C:\WINDOWS\system32\cmd.exe
Los objetos en la mochila son:

Obj_4 con un peso de objeto de 56 y con un beneficio de 20
Obj_8 con un peso de objeto de 50 y con un beneficio de 20
Obj_1 con un peso de objeto de 100 y con un beneficio de 15
Obj_3 con un peso de objeto de 150 y con un beneficio de 12
Obj_2 con un peso de objeto de 2 y con un beneficio de 10
Obj_10 con un peso de objeto de 5 y con un beneficio de 8
Obj_9 con un peso de objeto de 150 y con un beneficio de 6
Obj_11 con un peso de objeto de 10 y con un beneficio de 6
Obj_17 con un peso de objeto de 30 y con un beneficio de 6
Obj_20 con un peso de objeto de 150 y con un beneficio de 6
Obj_15 con un peso de objeto de 20 y con un beneficio de 5
Obj_6 con un peso de objeto de 10 y con un beneficio de 2
Obj_19 con un peso de objeto de 9 y con un beneficio de 2
Obj_14 con un peso de objeto de 100 y con un beneficio de 1
Obj_16 con un peso de objeto de 15 y con un beneficio de 1
Obj_18 con un peso de objeto de 15 y con un beneficio de 1
Obj_12 con un peso de objeto de 300 y con un beneficio de 8
Obj_13 con un peso de objeto de 500 y con un beneficio de 10

Con un peso en la mochila de 1672 y un beneficio maximo de 139
Tiempo de ejecución : 0.003427608602223943

C:\Users\PC\Documents\Semestre Enero-Junio 2017\TSO\Mochila>
```

Caso1:

$$Eficiencia = 100 - \left(\frac{309-277}{309} \times 100 \right) = 89.6 \%$$

```
C:\WINDOWS\system32\cmd.exe
C:\Users\PC\Documents\Semestre Enero-Junio 2017\TSO\Mochila>python mochila.py

Ingrese el nombre del archivo [Clase],[Caso1],[Caso2]....[Caso'n']
Caso1
La capacidad máxima es: 165
Los objetos en la mochila son:

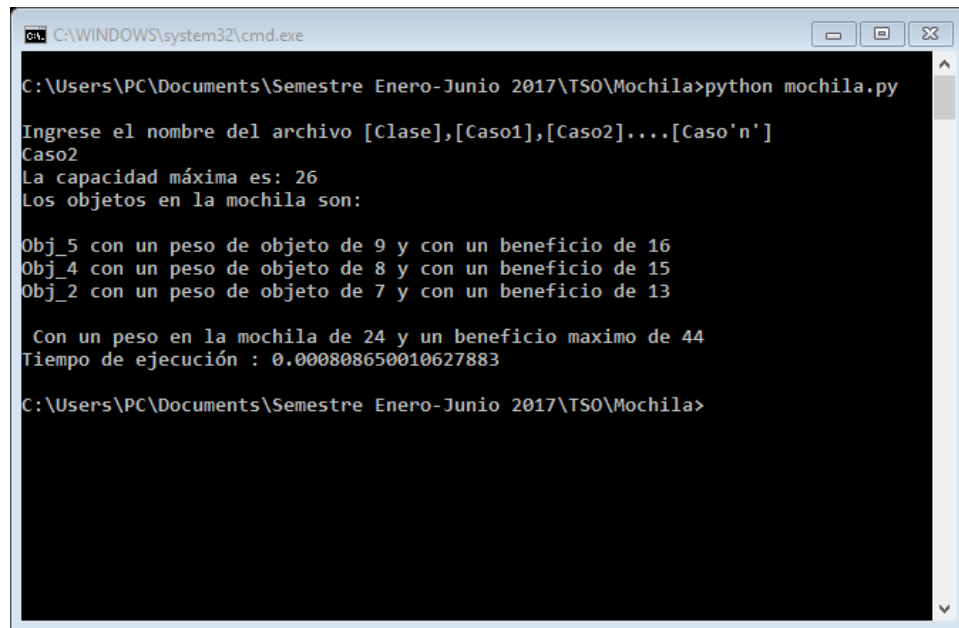
Obj_1 con un peso de objeto de 23 y con un beneficio de 92
Obj_4 con un peso de objeto de 44 y con un beneficio de 68
Obj_5 con un peso de objeto de 53 y con un beneficio de 60
Obj_2 con un peso de objeto de 31 y con un beneficio de 57

Con un peso en la mochila de 151 y un beneficio maximo de 277
Tiempo de ejecución : 0.0012042865277232591

C:\Users\PC\Documents\Semestre Enero-Junio 2017\TSO\Mochila>
```

Caso2:

$$Eficiencia = 100 - \left(\frac{51-44}{51} \times 100 \right) = 86.27\%$$



```
C:\WINDOWS\system32\cmd.exe

C:\Users\PC\Documents\Semestre Enero-Junio 2017\TSO\Mochila>python mochila.py

Ingrese el nombre del archivo [Clase],[Caso1],[Caso2]....[Caso'n']
Caso2
La capacidad máxima es: 26
Los objetos en la mochila son:

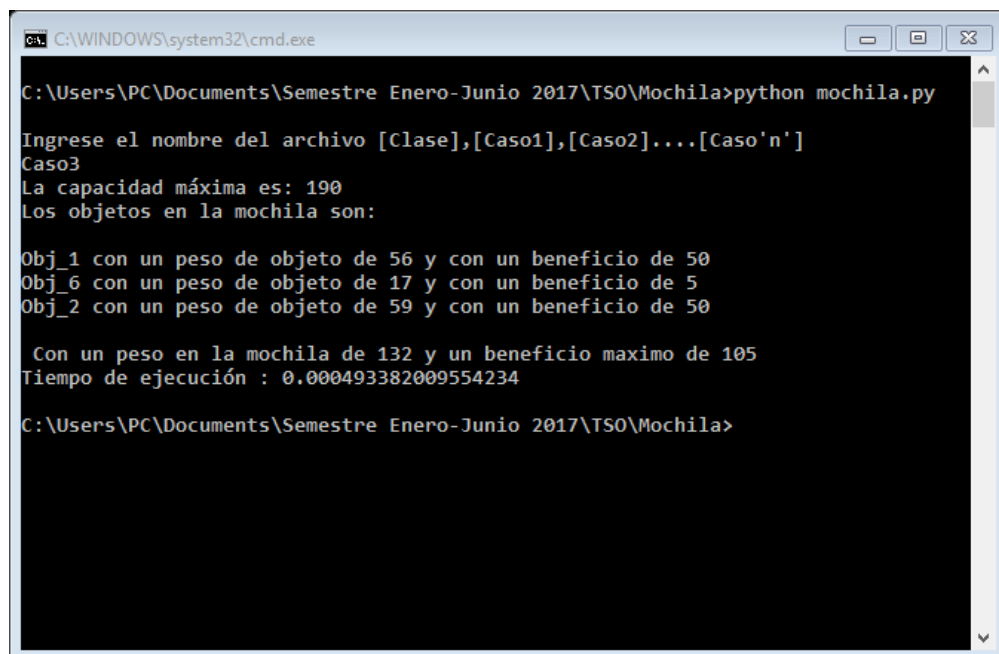
Obj_5 con un peso de objeto de 9 y con un beneficio de 16
Obj_4 con un peso de objeto de 8 y con un beneficio de 15
Obj_2 con un peso de objeto de 7 y con un beneficio de 13

Con un peso en la mochila de 24 y un beneficio maximo de 44
Tiempo de ejecución : 0.000808650010627883

C:\Users\PC\Documents\Semestre Enero-Junio 2017\TSO\Mochila>
```

Caso3:

$$Eficiencia = 100 - \left(\frac{150-105}{150} \times 100 \right) = 70 \%$$



```
C:\WINDOWS\system32\cmd.exe

C:\Users\PC\Documents\Semestre Enero-Junio 2017\TSO\Mochila>python mochila.py

Ingrese el nombre del archivo [Clase],[Caso1],[Caso2]....[Caso'n']
Caso3
La capacidad máxima es: 190
Los objetos en la mochila son:

Obj_1 con un peso de objeto de 56 y con un beneficio de 50
Obj_6 con un peso de objeto de 17 y con un beneficio de 5
Obj_2 con un peso de objeto de 59 y con un beneficio de 50

Con un peso en la mochila de 132 y un beneficio maximo de 105
Tiempo de ejecución : 0.000493382009554234

C:\Users\PC\Documents\Semestre Enero-Junio 2017\TSO\Mochila>
```


Caso4:

$$Eficiencia = 100 - \left(\frac{107-79}{107} \times 100 \right) = 73.83 \%$$

```
C:\WINDOWS\system32\cmd.exe

C:\Users\PC\Documents\Semestre Enero-Junio 2017\TSO\Mochila>python mochila.py

Ingrese el nombre del archivo [Clase],[Caso1],[Caso2]...[Caso'n']
Caso4
La capacidad máxima es: 50
Los objetos en la mochila son:

Obj_2 con un peso de objeto de 10 y con un beneficio de 20
Obj_7 con un peso de objeto de 6 y con un beneficio de 10
Obj_5 con un peso de objeto de 4 y con un beneficio de 7
Obj_6 con un peso de objeto de 3 y con un beneficio de 5
Obj_4 con un peso de objeto de 19 y con un beneficio de 37

Con un peso en la mochila de 42 y un beneficio maximo de 79
Tiempo de ejecución : 0.0005501674861255704

C:\Users\PC\Documents\Semestre Enero-Junio 2017\TSO\Mochila>
```

Caso5:

$$Eficiencia = 100 - \left(\frac{900-858}{900} \times 100 \right) = 95.33\%$$

```
C:\WINDOWS\system32\cmd.exe

C:\Users\PC\Documents\Semestre Enero-Junio 2017\TSO\Mochila>python mochila.py

Ingrese el nombre del archivo [Clase],[Caso1],[Caso2]...[Caso'n']
Caso5
La capacidad máxima es: 104
Los objetos en la mochila son:

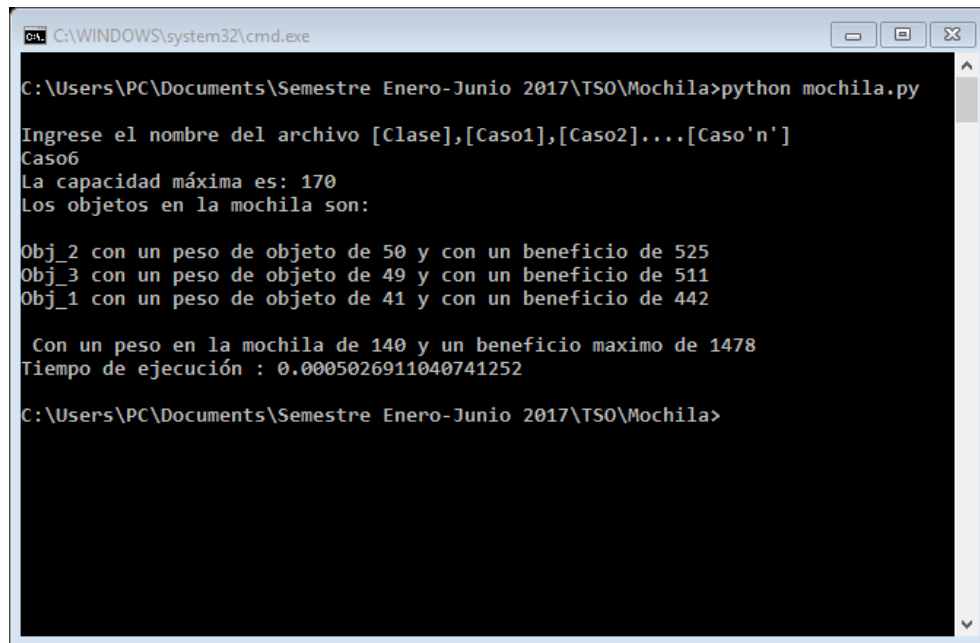
Obj_4 con un peso de objeto de 5 y con un beneficio de 20
Obj_6 con un peso de objeto de 3 y con un beneficio de 8
Obj_7 con un peso de objeto de 2 y con un beneficio de 5
Obj_8 con un peso de objeto de 2 y con un beneficio de 5
Obj_5 con un peso de objeto de 25 y con un beneficio de 70
Obj_1 con un peso de objeto de 25 y con un beneficio de 350
Obj_2 con un peso de objeto de 35 y con un beneficio de 400

Con un peso en la mochila de 97 y un beneficio maximo de 858
Tiempo de ejecución : 0.0010537894996516847

C:\Users\PC\Documents\Semestre Enero-Junio 2017\TSO\Mochila>
```

Caso6:

$$Eficiencia = 100 - \left(\frac{1735-1478}{1735} \times 100 \right) = 85.18 \%$$



```
C:\WINDOWS\system32\cmd.exe

C:\Users\PC\Documents\Semestre Enero-Junio 2017\TSO\Mochila>python mochila.py

Ingrese el nombre del archivo [Clase],[Caso1],[Caso2]...[Caso'n']
Caso6
La capacidad máxima es: 170
Los objetos en la mochila son:

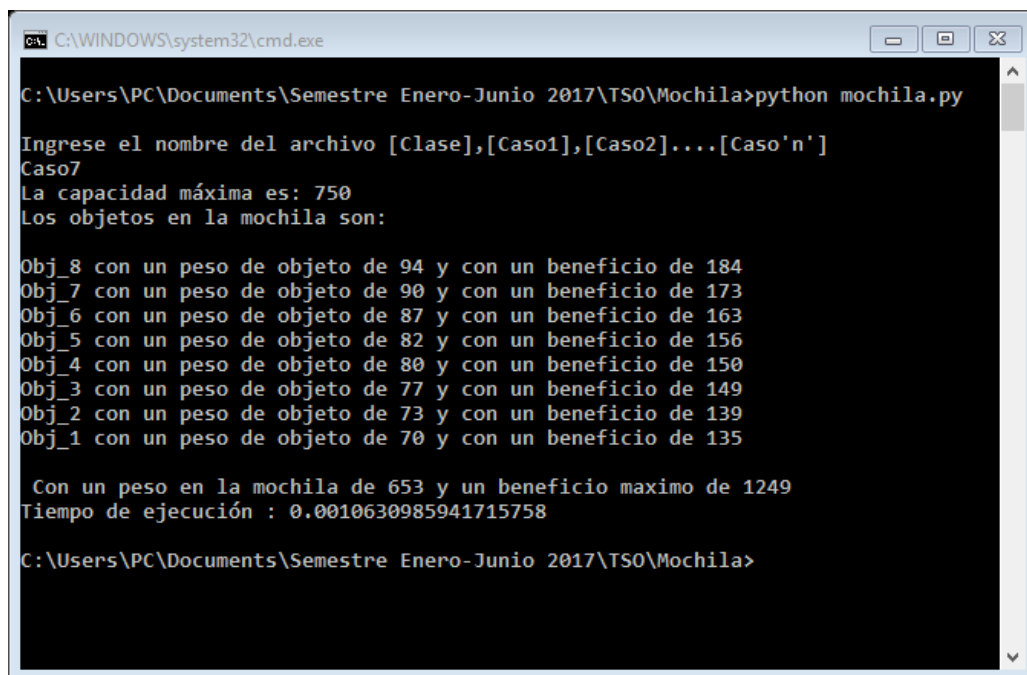
Obj_2 con un peso de objeto de 50 y con un beneficio de 525
Obj_3 con un peso de objeto de 49 y con un beneficio de 511
Obj_1 con un peso de objeto de 41 y con un beneficio de 442

Con un peso en la mochila de 140 y un beneficio maximo de 1478
Tiempo de ejecución : 0.0005026911040741252

C:\Users\PC\Documents\Semestre Enero-Junio 2017\TSO\Mochila>
```

Caso7:

$$Eficiencia = 100 - \left(\frac{1458-1249}{1458} \times 100 \right) = 85.66 \%$$



```
C:\WINDOWS\system32\cmd.exe

C:\Users\PC\Documents\Semestre Enero-Junio 2017\TSO\Mochila>python mochila.py

Ingrese el nombre del archivo [Clase],[Caso1],[Caso2]...[Caso'n']
Caso7
La capacidad máxima es: 750
Los objetos en la mochila son:

Obj_8 con un peso de objeto de 94 y con un beneficio de 184
Obj_7 con un peso de objeto de 90 y con un beneficio de 173
Obj_6 con un peso de objeto de 87 y con un beneficio de 163
Obj_5 con un peso de objeto de 82 y con un beneficio de 156
Obj_4 con un peso de objeto de 80 y con un beneficio de 150
Obj_3 con un peso de objeto de 77 y con un beneficio de 149
Obj_2 con un peso de objeto de 73 y con un beneficio de 139
Obj_1 con un peso de objeto de 70 y con un beneficio de 135

Con un peso en la mochila de 653 y un beneficio maximo de 1249
Tiempo de ejecución : 0.0010630985941715758

C:\Users\PC\Documents\Semestre Enero-Junio 2017\TSO\Mochila>
```

Caso8:

$$Eficiencia = 100 - \left(\frac{14,395,640 - 11,717,356}{14,395,640} \times 100 \right) = 81.39 \%$$

```
C:\WINDOWS\system32\cmd.exe
Caso8
La capacidad máxima es: 6404180
Los objetos en la mochila son:

Obj_15 con un peso de objeto de 488960 y con un beneficio de 953277
Obj_5 con un peso de objeto de 467902 y con un beneficio de 943972
Obj_21 con un peso de objeto de 496951 y con un beneficio de 901489
Obj_18 con un peso de objeto de 446298 y con un beneficio de 853655
Obj_1 con un peso de objeto de 382745 y con un beneficio de 825594
Obj_17 con un peso de objeto de 323046 y con un beneficio de 675367
Obj_22 con un peso de objeto de 264724 y con un beneficio de 577243
Obj_23 con un peso de objeto de 224916 y con un beneficio de 466257
Obj_24 con un peso de objeto de 169684 y con un beneficio de 369261
Obj_6 con un peso de objeto de 44328 y con un beneficio de 97426
Obj_7 con un peso de objeto de 34610 y con un beneficio de 69666
Obj_20 con un peso de objeto de 31385 y con un beneficio de 65731
Obj_12 con un peso de objeto de 551830 y con un beneficio de 1049289
Obj_13 con un peso de objeto de 610856 y con un beneficio de 1252836
Obj_14 con un peso de objeto de 670702 y con un beneficio de 1319836
Obj_8 con un peso de objeto de 698150 y con un beneficio de 1296457

Con un peso en la mochila de 5907087 y un beneficio maximo de 11717356
Tiempo de ejecución : 0.002826241096238971
C:\Users\PC\Documents\Semestre Enero-Junio 2017\TSO\Mochila>
```

CARACTERISTICAS DEL SISTEMA

Procesador: Intel® Core i5-4590 CPU @ 3.30 GHz

Memoria Instalada (RAM): 8.00 GB (7.88 GB utilizable)

Sistema Operativo: Windows 10 Pro

Tipo de sistema: Sistema operativo de 64 bits, procesador x64

CONCLUSIONES

Este algoritmo fue mucho más sencillo de codificar que los realizados anteriormente, no se presentó ninguna dificultad, sin embargo, recomiendo realizar estas tareas con tiempo y revisar siempre los resultados, ya que en este ejemplo tuve un error de lógica en el que no seleccionaba bien los objetos que se introducían a la mochila, error que solo ocurría en uno de los 9 casos revisados y que no hubiese notado sin revisar todos los resultados.

BIBLIOGRAFIA

- <https://www.python.org/doc/>