








Programmazione Lineare con GLPK  
**Problemi di Soldi**  
*Giovanni Di Maria*

Abbiamo le seguenti entità:

			
<b>m1</b>	<b>m2</b>	<b>m3</b>	<b>m4</b>
			
<b>m5</b>	<b>m6</b>	<b>m7</b>	<b>m8</b>

			
<b>b1</b>	<b>b2</b>	<b>b3</b>	<b>b4</b>
			
<b>b5</b>	<b>b6</b>	<b>b7</b>	

$S$  = La somma da pagare

# 1° Problema

Dobbiamo pagare al negoziante la somma di € 1498.66. Egli (per non riempire il registratore di cassa di monetine o banconote, vuole il **NUMERO PIU' BASSO** di pezzi (sia di carta che di metallo).

Dal momento che possiamo usare eventualmente fino a 5 pezzi per ogni taglio, quanti pezzi per taglio dobbiamo consegnare, per pagare la somma? E' possibile evitare l'utilizzo di qualche taglio.

## Soluzione

Si scriva con un editor di testo il seguente listato, nel file "euro.txt":

```
var m1 >=0,<=5,integer;      /* moneta da 1 cent */
var m2 >=0,<=5,integer;      /* moneta da 2 cent */
var m3 >=0,<=5,integer;      /* moneta da 5 cent */
var m4 >=0,<=5,integer;      /* moneta da 10 cent */
var m5 >=0,<=5,integer;      /* moneta da 20 cent */
var m6 >=0,<=5,integer;      /* moneta da 50 cent */
var m7 >=0,<=5,integer;      /* moneta da 1 euro */
var m8 >=0,<=5,integer;      /* moneta da 2 euro */
var b1 >=0,<=5,integer;      /* banconota da 5 cent */
var b2 >=0,<=5,integer;      /* banconota da 10 cent */
var b3 >=0,<=5,integer;      /* banconota da 20 cent */
var b4 >=0,<=5,integer;      /* banconota da 50 cent */
var b5 >=0,<=5,integer;      /* banconota da 100 cent */
var b6 >=0,<=5,integer;      /* banconota da 200 cent */
var b7 >=0,<=5,integer;      /* banconota da 500 cent */

minimize QTPezzi: m1+m2+m3+m4+m5+m6+m7+m8+b1+b2+b3+b4+b5+b6+b7;

S: (m1*0.01)+(m2*0.02)+(m3*0.05)+(m4*0.10)+(m5*0.20)+(m6*0.50)+
(m7*1)+(m8*2)+(b1*5)+(b2*10)+(b3*20)+(b4*50)+(b5*100)+(b6*200)+
(b7*500)=1498.66;

end;
```

Alla fine, al prompt del sistema operativo, si digiti il seguente comando:

```
C:\Programmi\GnuWin32\bin> glpsol.exe -m euro.txt -o CON
```

Dopo qualche istante l'output a Video (CON) sarà il seguente (tralasciando le parti non interessanti):

```
INTEGER OPTIMAL SOLUTION FOUND
Time used: 0.2 secs
Memory used: 0.2 Mb (190819 bytes)
lp*_print_mip: writing MIP problem solution to `CON'...
Problem: euro
Rows: 2
Columns: 15 (15 integer, 0 binary)
Non-zeros: 30
Status: INTEGER OPTIMAL
Objective: QTpezzi = 14 (MINimum)
```

No.	Row name	Activity	Lower bound	Upper bound
1	QTpezzi	<b>14</b>		
2	S	1498.66	1498.66	=

No.	Column name	Activity	Lower bound	Upper bound	
1	m1	*	<b>1</b>	0	5
2	m2	*	0	0	5
3	m3	*	<b>1</b>	0	5
4	m4	*	<b>1</b>	0	5
5	m5	*	0	0	5
6	m6	*	<b>1</b>	0	5
7	m7	*	<b>1</b>	0	5
8	m8	*	<b>1</b>	0	5
9	b1	*	<b>1</b>	0	5
10	b2	*	0	0	5
11	b3	*	<b>2</b>	0	5
12	b4	*	<b>1</b>	0	5
13	b5	*	0	0	5
14	b6	*	<b>2</b>	0	5
15	b7	*	<b>2</b>	0	5

Questa soluzione di MINIMIZZAZIONE ha pertanto decretato il numero “minimo” di pezzi che consentono di soddisfare la somma di € 1498.66, utilizzando solo 14 pezzi, così ripartiti:

- 1 moneta da 1 cent
- 1 moneta da 5 cent
- 1 moneta da 10 cent
- 1 moneta da 50 cent
- 1 moneta da 1 euro
- 1 moneta da 2 euro
- 1 banconota da 5 euro
- 2 banconote da 20 euro
- 1 banconota da 50 euro
- 2 banconote da 200 euro
- 2 banconote da 500 euro

---

TOTALE:           € 1498.66

Meno di così non si può!

Come si vede, occorre pianificare bene il problema e formalizzarlo in una notazione matematico-logica.

## 2° Problema

Il secondo problema è un tantino più difficoltoso. Dobbiamo sempre pagare al negoziante la somma di € 1498.66. Egli (per non riempire il registratore di cassa di monetine e banconote, vuole il **NUMERO PIU' BASSO** di pezzi (sia di carta che di metallo).

In aggiunta c'è qualche condizione che il negoziante (molto pignolo) vuole rispettare: il numero delle monetine gialle (10 cent, 20 cent, 50 cent) deve essere maggiore delle monetine marroni (1 cent, 2 cent, 5 cent).

Dal momento che possiamo usare eventualmente fino a 5 pezzi per ogni taglio, quanti pezzi per taglio dobbiamo consegnare, per pagare la somma? E' possibile evitare l'utilizzo di qualche taglio.

## Soluzione

Si scriva con un editor di testo il seguente listato, nel file "euro.txt":

```
var m1 >=0,<=5,integer;      /* moneta da 1 cent */
var m2 >=0,<=5,integer;      /* moneta da 2 cent */
var m3 >=0,<=5,integer;      /* moneta da 5 cent */
var m4 >=0,<=5,integer;      /* moneta da 10 cent */
var m5 >=0,<=5,integer;      /* moneta da 20 cent */
var m6 >=0,<=5,integer;      /* moneta da 50 cent */
var m7 >=0,<=5,integer;      /* moneta da 1 euro */
var m8 >=0,<=5,integer;      /* moneta da 2 euro */
var b1 >=0,<=5,integer;      /* banconota da 5 cent */
var b2 >=0,<=5,integer;      /* banconota da 10 cent */
var b3 >=0,<=5,integer;      /* banconota da 20 cent */
var b4 >=0,<=5,integer;      /* banconota da 50 cent */
var b5 >=0,<=5,integer;      /* banconota da 100 cent */
var b6 >=0,<=5,integer;      /* banconota da 200 cent */
var b7 >=0,<=5,integer;      /* banconota da 500 cent */

minimize QTPezzi: m1+m2+m3+m4+m5+m6+m7+m8+b1+b2+b3+b4+b5+b6+b7;
MoneteGialle: (m4+m5+m6) - (m1+m2+m3) >=1;

S: (m1*0.01) + (m2*0.02) + (m3*0.05) + (m4*0.10) + (m5*0.20) + (m6*0.50) +
(m7*1) + (m8*2) + (b1*5) + (b2*10) + (b3*20) + (b4*50) + (b5*100) + (b6*200) +
(b7*500) = 1498.66;

end;
```

Alla fine, al prompt del sistema operativo, si digiti il seguente comando:

```
C:\Programmi\GnuWin32\bin> glpsol.exe -m euro.txt -o CON
```

Dopo qualche istante l'output a Video (CON) sarà il seguente (tralasciando le parti non interessanti):

```
INTEGER OPTIMAL SOLUTION FOUND
Time used: 0.2 secs
Memory used: 0.3 Mb (303274 bytes)
lp*_print_mip: writing MIP problem solution to `CON'...
Problem: euro
Rows: 3
Columns: 15 (15 integer, 0 binary)
Non-zeros: 36
Status: INTEGER OPTIMAL
Objective: QTpezzi = 15 (MINimum)
```

No.	Row name	Activity	Lower bound	Upper bound
1	QTpezzi	<b>15</b>		
2	MoneteGialle	2	1	
3	S	1498.66	1498.66	=

No.	Column name	Activity	Lower bound	Upper bound	
1	m1	*	<b>1</b>	0	5
2	m2	*	0	0	5
3	m3	*	<b>1</b>	0	5
4	m4	*	<b>1</b>	0	5
5	m5	*	0	0	5
6	m6	*	<b>3</b>	0	5
7	m7	*	0	0	5
8	m8	*	<b>1</b>	0	5
9	b1	*	<b>1</b>	0	5
10	b2	*	0	0	5
11	b3	*	<b>2</b>	0	5
12	b4	*	<b>1</b>	0	5
13	b5	*	0	0	5
14	b6	*	<b>2</b>	0	5
15	b7	*	<b>2</b>	0	5

Questa soluzione di MINIMIZZAZIONE ha pertanto decretato il numero “minimo” di pezzi che consentono di soddisfare la somma di € 1498.66, utilizzando solo 15 pezzi.

In più le monete gialle sono 4 mentre quelle marroni sono solo 2. Nel precedente problema tale numero era lo stesso per entrambe.

Meno di così non si può!

## 3° Problema

Il terzo problema è ancora più difficoltoso. Dobbiamo sempre pagare al negoziante la somma di € 1498.66. Egli (per non riempire il registratore di cassa di monetine e di banconote, vuole il **NUMERO PIU' BASSO** di pezzi (sia di carta che di metallo).

In aggiunta c'è qualche condizione che il negoziante (molto pignolo) vuole rispettare:

1. Il numero delle monetine **gialle** (10 cent, 20 cent, 50 cent) deve essere maggiore delle monetine **marroni** (1 cent, 2 cent, 5 cent).
2. Il numero delle banconote deve risultare il **DOPPIO** di quello delle monete.

Dal momento che possiamo usare eventualmente fino a 5 pezzi per ogni taglio, quanti pezzi per taglio dobbiamo consegnare, per pagare la somma? E' possibile evitare l'utilizzo di qualche taglio.

## Soluzione

Si scriva con un editor di testo il seguente listato, nel file "euro.txt":

```
var m1 >=0,<=5,integer;      /* moneta da 1 cent */
var m2 >=0,<=5,integer;      /* moneta da 2 cent */
var m3 >=0,<=5,integer;      /* moneta da 5 cent */
var m4 >=0,<=5,integer;      /* moneta da 10 cent */
var m5 >=0,<=5,integer;      /* moneta da 20 cent */
var m6 >=0,<=5,integer;      /* moneta da 50 cent */
var m7 >=0,<=5,integer;      /* moneta da 1 euro */
var m8 >=0,<=5,integer;      /* moneta da 2 euro */
var b1 >=0,<=5,integer;      /* banconota da 5 cent */
var b2 >=0,<=5,integer;      /* banconota da 10 cent */
var b3 >=0,<=5,integer;      /* banconota da 20 cent */
var b4 >=0,<=5,integer;      /* banconota da 50 cent */
var b5 >=0,<=5,integer;      /* banconota da 100 cent */
var b6 >=0,<=5,integer;      /* banconota da 200 cent */
var b7 >=0,<=5,integer;      /* banconota da 500 cent */

minimize QTPezzi: m1+m2+m3+m4+m5+m6+m7+m8+b1+b2+b3+b4+b5+b6+b7;
MoneteGialle: (m4+m5+m6) - (m1+m2+m3) >=1;
BanconDoppio: (b1+b2+b3+b4+b5+b6+b7) = (m1+m2+m3+m4+m5+m6+m7+m8) * 2;

S: (m1*0.01) + (m2*0.02) + (m3*0.05) + (m4*0.10) + (m5*0.20) + (m6*0.50) +
(m7*1) + (m8*2) + (b1*5) + (b2*10) + (b3*20) + (b4*50) + (b5*100) + (b6*200) +
(b7*500) = 1498.66;

end;
```

Alla fine, al prompt del sistema operativo, si digiti il seguente comando:

```
C:\Programmi\GnuWin32\bin> glpsol.exe -m euro.txt -o CON
```

Dopo qualche istante (più lungo dei precedenti) l'output a Video (CON) sarà il seguente (tralasciando le parti non interessanti):

```
INTEGER OPTIMAL SOLUTION FOUND
Time used: 2.9 secs
Memory used: 1.2 Mb (1248589 bytes)
lp*_print_mip: writing MIP problem solution to `CON'...
Problem: euro
Rows: 4
Columns: 15 (15 integer, 0 binary)
Non-zeros: 51
Status: INTEGER OPTIMAL
Objective: QTPezzi = 21 (MINimum)
```

No.	Row name	Activity	Lower bound	Upper bound
1	QTPezzi	<b>21</b>		
2	MoneteGialle	1	1	
3	BanconDoppio	0	-0	=
4	S	1498.66	1498.66	=

No.	Column name	Activity	Lower bound	Upper bound	
1	m1	*	<b>1</b>	0	5
2	m2	*	0	0	5
3	m3	*	<b>1</b>	0	5
4	m4	*	0	0	5
5	m5	*	<b>3</b>	0	5
6	m6	*	0	0	5
7	m7	*	<b>1</b>	0	5
8	m8	*	<b>1</b>	0	5
9	b1	*	<b>1</b>	0	5
10	b2	*	<b>5</b>	0	5
11	b3	*	<b>2</b>	0	5
12	b4	*	<b>2</b>	0	5
13	b5	*	<b>1</b>	0	5
14	b6	*	<b>1</b>	0	5
15	b7	*	<b>2</b>	0	5

Questa soluzione di MINIMIZZAZIONE ha pertanto decretato il numero “minimo” di pezzi che consentono di soddisfare la somma di € 1498.66, utilizzando solo 21 pezzi.

In più le monete gialle sono 3 mentre quelle marroni sono solo 2. E ancora il numero di banconote è 14 mentre le monetine sono 7.

Meno di così non si può!

Grazie dell'attenzione.

Caltanissetta, 29 settembre 2010

*Giovanni Di Maria - Gruppo Eratostene*