

TALLER DE CPLEX

Mario César López Locés



5 de febrero del 2020

CONTENTS

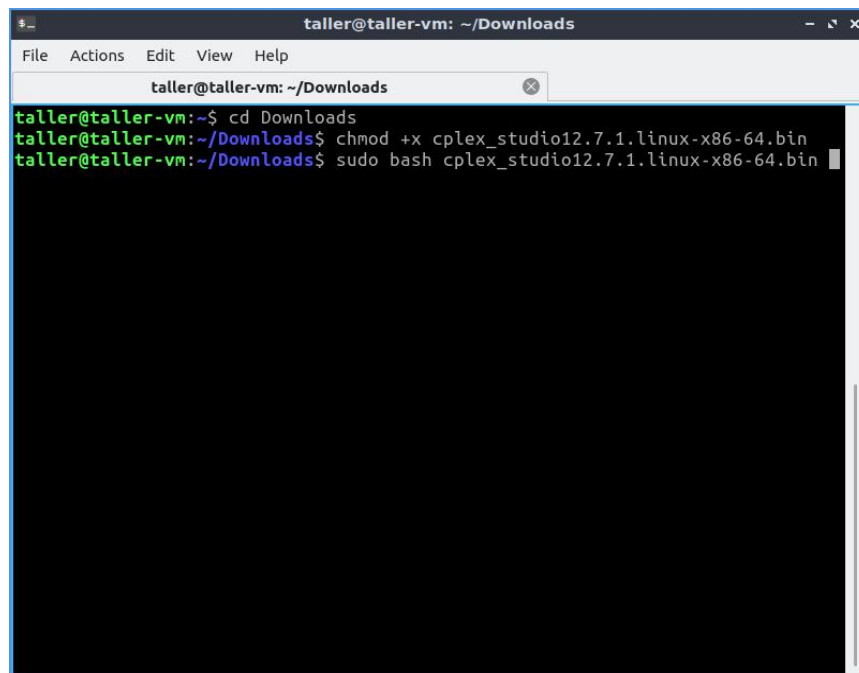
1	Instalar CPLEX (GNU/Linux)	2
2	Conexión al servidor yalma	2
2.1	Transferir archivos	3
3	El Problema de la Dieta	4
3.1	Modelos	4
3.1.1	Conjuntos	4
3.1.2	Parámetros	4
3.1.3	Variable de decisión	5
3.1.4	LP	5
3.1.5	MIP	5
3.1.6	Instancia JSON	5
4	Python	7
4.1	Instalar Miniconda	7
4.2	Problema de la dieta en Python	8
4.2.1	Ejecutar desde terminal	9
5	C++	10
5.1	Organización del proyecto	10
5.2	Problema de la dieta en C++	10
5.2.1	Código	10
5.2.2	Makefile	13
5.3	Ejecutar desde la terminal	13
6	Java	15
6.1	Crear proyecto en Eclipse	15
6.1.1	Configurar proyecto	16
6.2	Problema de la dieta en Java	17
6.3	Generar ejecutable en Eclipse	19
6.3.1	Generar archivo jar	19
6.3.2	Ejecutar desde terminal	21
7	Material adicional	22

1 INSTALAR CPLEX (GNU/LINUX)

Proceso para instalar CPLEX en GNU/Linux. Se recomienda aceptar las opciones por defecto del instalador.

```
1 cd Downloads
2 chmod +x cplex_studio12.7.1.linux-x86-64.bin
3 sudo bash cplex_studio12.7.1.linux-x86-64.bin
```

1. Cambiarse al directorio que contiene el instalador.
2. Asignarle permisos de ejecución.
3. Ejecutar con permisos de administrador.

A screenshot of a terminal window titled 'taller@taller-vm: ~/Downloads'. The terminal shows the following commands and their outputs: 'cd Downloads', 'chmod +x cplex_studio12.7.1.linux-x86-64.bin', and 'sudo bash cplex_studio12.7.1.linux-x86-64.bin'. The terminal has a menu bar with 'File', 'Actions', 'Edit', 'View', and 'Help'. The window has standard Linux window controls (minimize, maximize, close) in the top right corner.

```
taller@taller-vm: ~/Downloads
File Actions Edit View Help
taller@taller-vm: ~/Downloads
taller@taller-vm:~$ cd Downloads
taller@taller-vm:~/Downloads$ chmod +x cplex_studio12.7.1.linux-x86-64.bin
taller@taller-vm:~/Downloads$ sudo bash cplex_studio12.7.1.linux-x86-64.bin
```

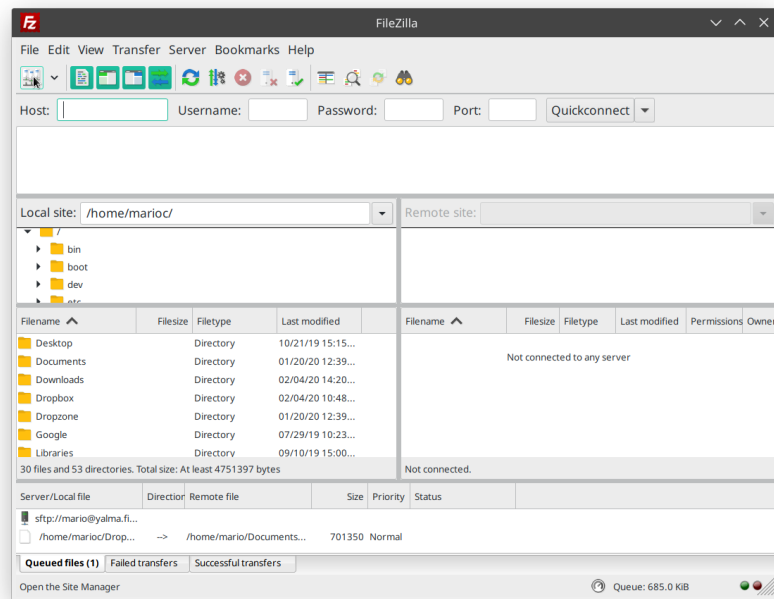
2 CONEXIÓN AL SERVIDOR YALMA

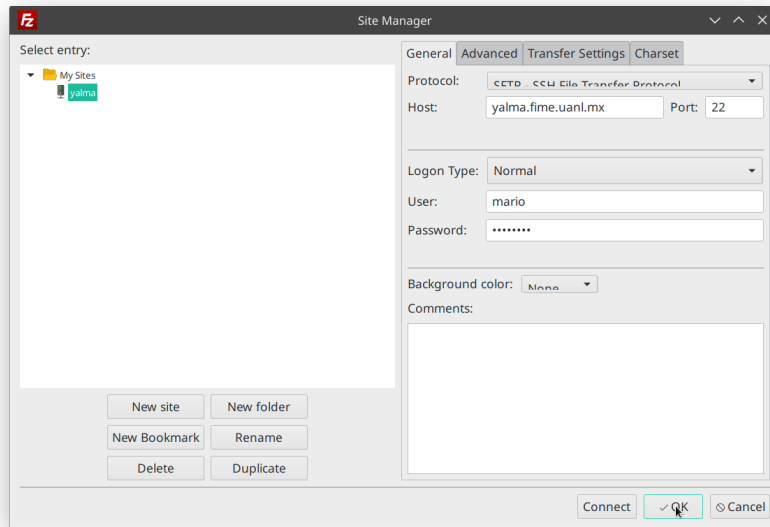
Desde una terminal en GNU/Linux o MacOS o desde una terminal de Putty:

```
1 ssh -l $USUARIO -p22 yalma.fime.uanl.mx
```

Posteriormente, escribir la contraseña cuando el servidor la solicite.

2.1 Transferir archivos





3 EL PROBLEMA DE LA DIETA

3.1 Modelos

La instancia de prueba en formato json se encuentra en el archivo `instancia.json`.

3.1.1 Conjuntos

- I = conjunto de comidas.
- J = conjunto de nutrientes.

3.1.2 Parámetros

- a_{ij} = cantidad del nutriente j en la comida i .
- c_i = costo de porción de la comida i .
- b_j = requerimiento mínimo del nutriente j .
- B_j = cantidad máxima permitida del nutriente j .
- d_j = requerimiento mínimo de porciones de la comida i .
- D_j = cantidad máxima permitida de porciones de la comida i .

3.1.3 Variable de decisión

- x_i = número de porciones compradas de la comida i .

3.1.4 LP

$$\min \sum_{i \in I} c_i x_i \quad (1)$$

$$\text{sujeto a: } \sum_{i \in I} a_{ij} x_i \geq b_j, \quad \forall j \in J \quad (2)$$

$$\sum_{i \in I} a_{ij} x_i \leq B_j, \quad \forall j \in J \quad (3)$$

$$x_i \geq d_i, \quad \forall i \in I \quad (4)$$

$$x_i \leq D_i, \quad \forall i \in I \quad (5)$$

$$x_i \in \mathbb{R}^+, \quad \forall i \in I \quad (6)$$

3.1.5 MIP

$$\min \sum_{i \in I} c_i x_i \quad (7)$$

$$\text{sujeto a: } \sum_{i \in I} a_{ij} x_i \geq b_j, \quad \forall j \in J \quad (8)$$

$$\sum_{i \in I} a_{ij} x_i \leq B_j, \quad \forall j \in J \quad (9)$$

$$x_i \geq d_i, \quad \forall i \in I \quad (10)$$

$$x_i \leq D_i, \quad \forall i \in I \quad (11)$$

$$x_i \in \mathbb{N}_0, \quad \forall i \in I \quad (12)$$

3.1.6 Instancia JSON

```
1 {
2   "nutrientes": 4,
3   "comidas": 9,
4   "costo_comida": [
5     249.0,
6     289.0,
7     150.0,
8     189.0,
9     209.0,
```

```

10     199.0,
11     249.0,
12     89.0,
13     159.0
14 ],
15 "comida_minima": [
16     0,
17     0,
18     0,
19     0,
20     0,
21     0,
22     0,
23     0,
24     0
25 ],
26 "comida_maxima": [
27     10,
28     10,
29     10,
30     10,
31     10,
32     10,
33     10,
34     10,
35     10
36 ],
37 "nutr_minimo": [
38     1800,
39     91,
40     0,
41     0
42 ],
43 "nutr_maximo": [
44     2200,
45     99999,
46     65,
47     1779
48 ],
49 "nutr_comida": [
50     [
51         410,
52         420,
53         560,
54         380,
55         320,
56         320,
57         320,
58         100,
59         330
60     ],
61     [
62         24,
63         32,
64         20,
65         4,

```

```

66     12,
67     15,
68     31,
69     8,
70     8
71 ],
72 [
73     26,
74     10,
75     32,
76     19,
77     10,
78     12,
79     12,
80     2.5,
81     10
82 ],
83 [
84     730,
85     1190,
86     1800,
87     270,
88     930,
89     820,
90     1230,
91     125,
92     180
93 ]
94 ]
95 }

```

4 PYTHON

4.1 Instalar Miniconda

- Descargar Miniconda de (<https://docs.conda.io/en/latest/miniconda.html>) e instalar siguiendo las instrucciones del sitio para la plataforma en la que se utilizará y transferirlo a yalma.

O descargar directo en yalma con wget.

```

1  wget
   ↪ https://repo.anaconda.com/miniconda/Miniconda2-latest-Linux-x86_64.sh
   ↪ -O miniconda.sh
2  bash miniconda.sh

```

- Crear un ambiente de trabajo, activarlo e instalar el paquete para utilizar CPLEX en Python.


```

1 conda create --name taller
2 source activate taller
3 conda install python=3.6
4 conda install -c ibmdecisionoptimization cplex

```

4.2 Problema de la dieta en Python

```

1 import sys
2 import json
3 import cplex
4 from cplex.exceptions import CplexError
5
6 def dieta(fileName):
7     #fileName = "/home/taller/Desktop/instancia.json"
8     with open(fileName) as file:
9         data = json.load(file)
10        nutrientes = data['nutrientes']
11        comidas = data['comidas']
12        nutr_comida = data['nutr_comida']
13        costo_comida = data['costo_comida']
14        comida_minima = data['comida_minima']
15        comida_maxima = data['comida_maxima']
16        nutr_minimo = data['nutr_minimo']
17        nutr_maximo = data['nutr_maximo']
18        print(nutr_comida)
19        try:
20            modelo = cplex.Cplex()
21            modelo.objective.set_sense(
22                ↪ modelo.objective.sense.minimize)
23            nombres = ["x" + str(j) for j in range(comidas)]
24            print(costo_comida)
25            print(comida_minima)
26            print(comida_maxima)
27            modelo.variables.add(obj=costo_comida, lb=comida_minima,
28                ↪ ub=comida_maxima, names=nombres, types =
29                ↪ ['I']*comidas)
30
31            for n in range(nutrientes):
32                modelo.linear_constraints.add(
33                    lin_expr=[[nombres, nutr_comida[n]]],
34                    senses=["R"],
35                    rhs=[nutr_minimo[n]],
36                    range_values=[nutr_maximo[n] -
37                        ↪ nutr_minimo[n]])
38
39            modelo.parameters.tune.timelimit.set(1.0)
40            print("Time limit =
41                ↪ ",modelo.parameters.tune.timelimit.get())
42
43            modelo.solve()
44
45            solution = modelo.solution

```

```

41
42         print("Status = ", solution.get_status(), ":", end=' ')
43         print(solution.status[solution.get_status()])
44         print("Costo = ", solution.get_objective_value())
45
46         x = solution.get_values(0, modelo.variables.get_num() -
47             ↪ 1)
48         for i in range(modelo.variables.get_num()):
49             print("Comida", i, "=", x[i])
50     except CplexError as exc:
51         print(exc)
52         return
53
54 if __name__ == "__main__":
55     dieta(str(sys.argv[1]))

```

4.2.1 Ejecutar desde terminal

```

1 python dieta.py "instancia.json"

```

Verificar que está activo el ambiente taller, si no lo está, hacerlo con la instrucción `source activate taller`.

```

taller@taller-vm: /opt/ibm/ILOG/CPLEX_Studio1271/cplex/examples/src/python
File Actions Edit View Help
taller@taller-vm: /opt/ibm/ILOG/CPLEX_Studio1271/cplex/examples/src/python
(taller) taller@taller-vm: /opt/ibm/ILOG/CPLEX_Studio1271/cplex/examples/src/p
ython$ python diet.py -r
CPXPARAM_Read_DataCheck          1
Tried aggregator 1 time.
LP Presolve eliminated 0 rows and 5 columns.
Reduced LP has 7 rows, 11 columns, and 60 nonzeros.
Presolve time = 0.00 sec. (0.01 ticks)
Initializing dual steep norms . . .

Iteration log . . .
Iteration:    1   Dual objective    =          10.940000
Solution status = 1 : optimal
Objective value = 14.855737704918033
Buy 0 =          4.385245902
Buy 1 =              0
Buy 2 =              0
Buy 3 =              0
Buy 4 =              0
Buy 5 =          6.147540984
Buy 6 =              0
Buy 7 =          3.422131148
Buy 8 =              0
(taller) taller@taller-vm: /opt/ibm/ILOG/CPLEX_Studio1271/cplex/examples/src/p
ython$

```

5 C++

5.1 Organización del proyecto

Crear estructura del proyecto de la siguiente forma

- Dieta
 - src
 - include

Descargar la librería rapidjson de <https://github.com/Tencent/rapidjson/>. Descomprimir el archivo y copiar la carpeta rapidjson-master/include/rapidjson dentro de la carpeta include del proyecto.

- Dieta
 - src
 - include
 - * rapidjson

5.2 Problema de la dieta en C++

5.2.1 Código

```
1  #pragma GCC diagnostic ignored "-Wignored-attributes"
2  #include <iostream>
3  #include "rapidjson/filereadstream.h"
4  #include "rapidjson/document.h"
5  #include <stdio>
6  #include <vector>
7  #include <sstream>
8
9  using namespace rapidjson;
10 IOSTLBEGIN
11
12 IloNumArray getDataArray(const Value& data,
13                           IloEnv env,
14                           int size)
15 {
16     IloNumArray array(env, size);
17     int k = 0;
18     for(auto& c : data.GetArray()){
19         array[k] = c.GetDouble();
20         k++;
```

```

21     }
22     return array;
23 }
24
25 IloNumArray2 getDataMatrix(const Value& data,
26                             IloEnv env,
27                             int sizeX,
28                             int sizeY)
29 {
30     IloNumArray2 array(env, sizeX);
31     int k = 0;
32     for(auto& f : data.GetArray()){
33         array[k] = getDataArray(f, env, sizeY);
34         k++;
35     }
36     return array;
37 }
38
39 int main(int argc, char **argv)
40 {
41     IloEnv env;
42
43     try {
44         IloNumVar::Type varType = ILOINT;
45
46         std::string filename = argv[1];
47
48         IloInt i, j;
49
50         FILE* archivo = fopen(filename.c_str(), "r");
51
52         char read_buffer[65536];
53
54         rapidjson::FileReadStream is(archivo, read_buffer,
55                                     ↪ sizeof(read_buffer));
56
57         rapidjson::Document d;
58         d.ParseStream(is);
59
60         int nutrientes = d["nutrientes"].GetInt();
61         int comidas = d["comidas"].GetInt();
62
63         const Value& cc = d["costo_comida"];
64         IloNumArray foodCost = getDataArray(cc, env, comidas);
65
66         const Value& cm = d["comida_minima"];
67         IloNumArray foodMin = getDataArray(cm, env, comidas);
68
69         const Value& cma = d["comida_maxima"];
70         IloNumArray foodMax = getDataArray(cma, env, comidas);
71
72         const Value& nm = d["nutr_minimo"];
73         IloNumArray nutrMin = getDataArray(nm, env, nutrientes);
74
75         const Value& nma = d["nutr_maximo"];
76         IloNumArray nutrMax = getDataArray(nma, env, nutrientes);

```

```

76
77     const Value& nc = d["nutr_comida"];
78     IloNumArray2 nutrPer = getDataMatrix(nc, env,
79                                     nutrientes, comidas);
80
81     IloInt n = comidas;
82     IloInt m = nutrientes;
83
84     fclose(archivo);
85
86     IloModel      mod(env);
87     IloNumVarArray Buy(env);
88
89     IloNumVarArray tmp(env, foodMin, foodMax, varType);
90     Buy.add(tmp);
91     tmp.end();
92
93     mod.add(IloMinimize(env, IloScalProd(Buy, foodCost)));
94     for (i = 0; i < m; i++) {
95         IloExpr expr(env);
96         for (j = 0; j < n; j++) {
97             expr += Buy[j] * nutrPer[i][j];
98         }
99         mod.add(nutrMin[i] <= expr <= nutrMax[i]);
100        expr.end();
101    }
102    // Solve model
103
104    IloCplex cplex(mod);
105    cplex.solve();
106    cplex.out() << "solution status = " << cplex.getStatus() << endl;
107
108    cplex.out() << endl;
109    cplex.out() << "cost   = " << cplex.getObjValue() << endl;
110    for (i = 0; i < foodCost.getSize(); i++) {
111        cplex.out() << "   Buy" << i << " = " << cplex.getValue(Buy[i])
112        << endl;
113    }
114    catch (IloException& ex) {
115        cerr << "Error: " << ex << endl;
116    }
117    catch (...) {
118        cerr << "Error" << endl;
119    }
120
121    env.end();
122
123    return 0;
124 }

```

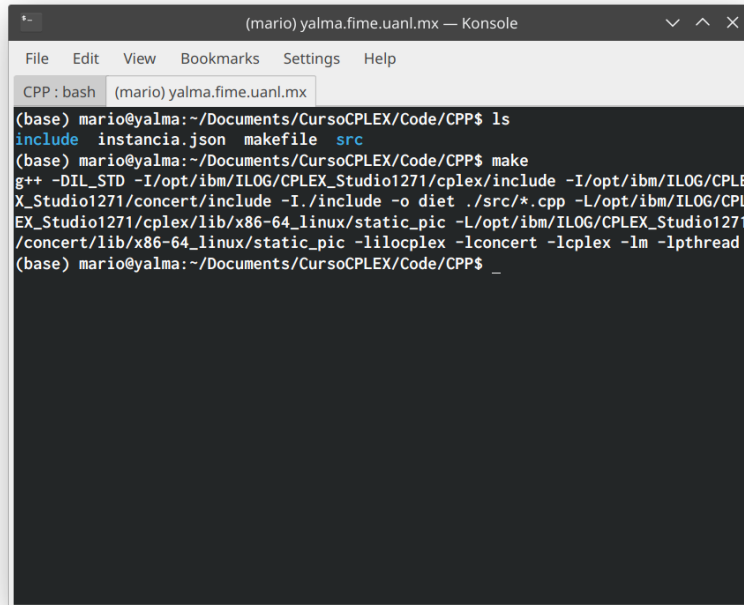
5.2.2 Makefile

```
1 CC = g++
2
3 INCLUDES = -I/opt/ibm/ILOG/CPLEX_Studio1271/cplex/include
4 ↪ -I/opt/ibm/ILOG/CPLEX_Studio1271/concert/include -I./include
5
6 LFLAGS =
7 ↪ -L/opt/ibm/ILOG/CPLEX_Studio1271/cplex/lib/x86-64_linux/static_pic
8 ↪ -L/opt/ibm/ILOG/CPLEX_Studio1271/concert/lib/x86-64_linux/static_pic
9
10 LIBS = -lilocplex -lconcert -lcplex -lm -lpthread
11
12 SRCS = ./src/*.cpp
13
14 OBJS = $(SRCS:.c=.o)
15
16 MAIN = diet
17
18 all: $(MAIN)
19
20 $(MAIN): $(OBJS)
21 ↪ $(CC) -DIL_STD $(INCLUDES) -o $(MAIN) $(OBJS) $(LFLAGS) $(LIBS)
```

5.3 Ejecutar desde la terminal

- El comando make compila el proyecto usando el makefile.

```
1 make
2 ./diet "instancia.json"
```



```
(mario) yalma.fime.uanl.mx — Konsole
File Edit View Bookmarks Settings Help
CPP: bash (mario) yalma.fime.uanl.mx
(base) mario@yalma:~/Documents/CursoCPLEX/Code/CPP$ ls
include instancia.json makefile src
(base) mario@yalma:~/Documents/CursoCPLEX/Code/CPP$ make
g++ -DIL_STD -I/opt/ibm/ILOG/CPLEX_Studio1271/cplex/include -I/opt/ibm/ILOG/CPLEX_Studio1271/concert/include -I./include -o diet ./src/*.cpp -L/opt/ibm/ILOG/CPLEX_Studio1271/cplex/lib/x86-64_linux/static_pic -L/opt/ibm/ILOG/CPLEX_Studio1271/concert/lib/x86-64_linux/static_pic -lilocplex -lconcert -lcplex -lm -lpthread
(base) mario@yalma:~/Documents/CursoCPLEX/Code/CPP$ _
```

```
(mario) yalma.fime.uanl.mx — Konsole
File Edit View Bookmarks Settings Help
CPP: bash (mario) yalma.fime.uanl.mx
Mixed integer rounding cuts applied: 1
Gomory fractional cuts applied: 1

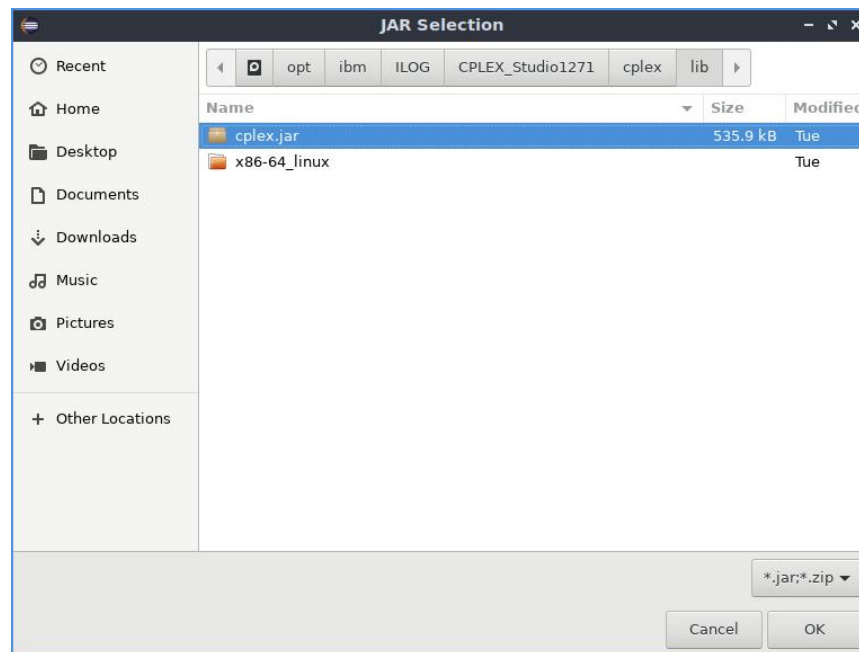
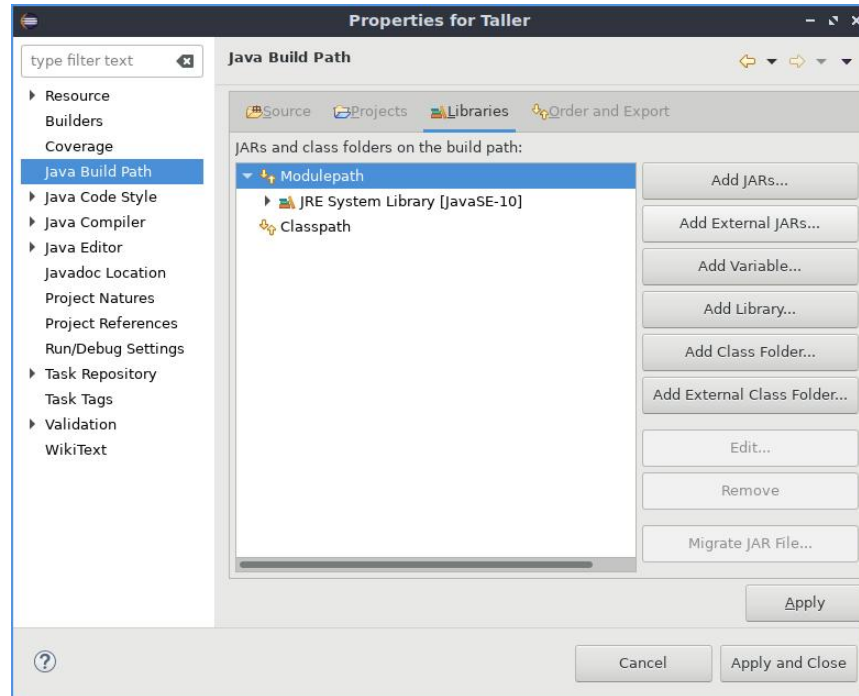
Root node processing (before b&c):
  Real time           = 0.03 sec. (0.22 ticks)
Parallel b&c, 8 threads:
  Real time           = 0.00 sec. (0.00 ticks)
  Sync time (average) = 0.00 sec.
  Wait time (average) = 0.00 sec.
-----
Total (root+branch&cut) = 0.03 sec. (0.22 ticks)
solution status = Optimal

cost = 1278
Buy0 = 0
Buy1 = -0
Buy2 = 0
Buy3 = 0
Buy4 = -0
Buy5 = 0
Buy6 = -0
Buy7 = 9
Buy8 = 3
(base) mario@yalma:~/Documents/CursoCPLEX/Code/CPP$ ./diet instancia.json _
```

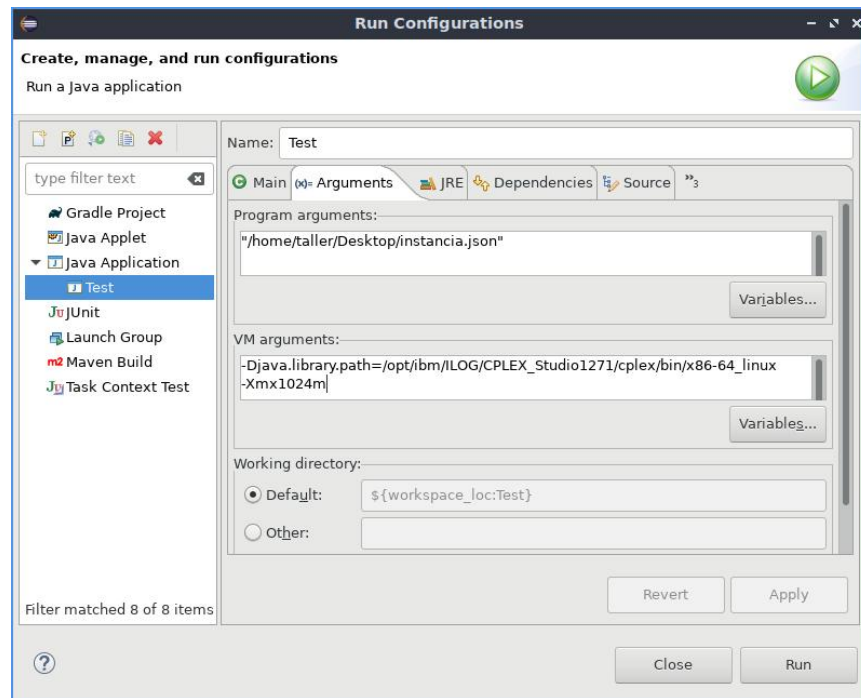
6 JAVA

6.1 Crear proyecto en Eclipse

6.1.1 Configurar proyecto



Descargar el paquete gson de (<https://search.maven.org/artifact/com.google.code.gson/gson/2.8.5/jar>) y agregar al proyecto actual siguiendo el mismo procedimiento.



6.2 Problema de la dieta en Java

```
1 import java.io.FileNotFoundException;
2 import java.io.FileReader;
3
4 import com.google.gson.*;
5
6 import ilog.concert.IloException;
7 import ilog.concert.IloNumVar;
8 import ilog.concert.IloNumVarType;
9 import ilog.cplex.IloCplex;
10
11 public class Test {
12     public static void main(String[] args) throws FileNotFoundException,
13         ↳ IloException {
14         Gson gson = new Gson();
15         String fileName = args[0];
16         JsonObject reader = gson.fromJson(new FileReader(fileName),
17         ↳ JsonObject.class);
18         JsonElement data;
```

```

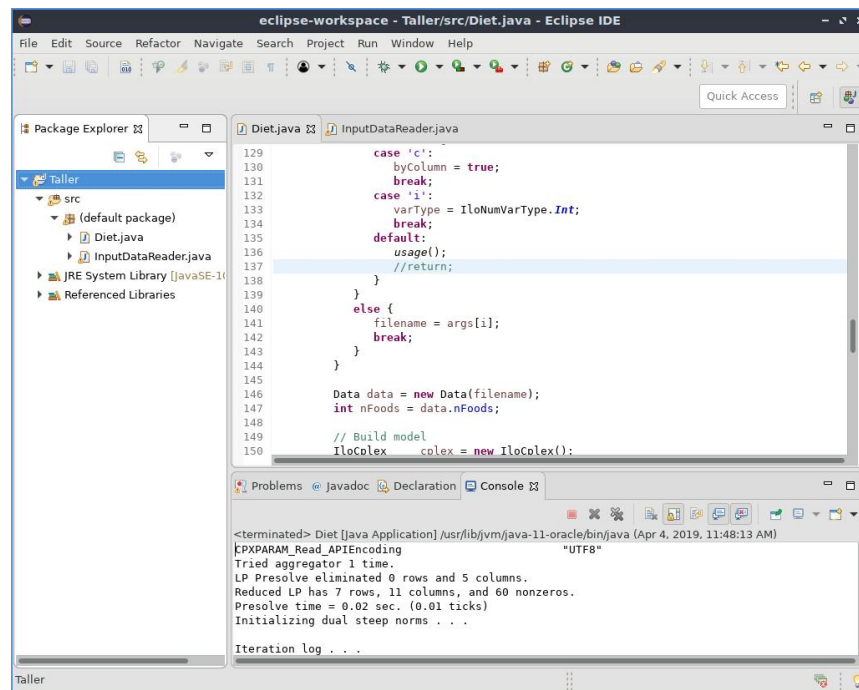
17
18     data = reader.get("nutrientes");
19     int nutrientes = gson.fromJson(data, int.class);
20
21     data = reader.get("comidas");
22     int comidas = gson.fromJson(data, int.class);
23
24     data = reader.get("nutr_comida");
25     double[][] nutr_comida = gson.fromJson(data, double[][].class);
26
27     data = reader.get("costo_comida");
28     double[] costo_comida = gson.fromJson(data, double[].class);
29
30     data = reader.get("comida_minima");
31     double[] comida_minima = gson.fromJson(data, double[].class);
32
33     data = reader.get("comida_maxima");
34     double[] comida_maxima = gson.fromJson(data, double[].class);
35
36     data = reader.get("nutr_minimo");
37     double[] nutr_minimo = gson.fromJson(data, double[].class);
38
39     data = reader.get("nutr_maximo");
40     double[] nutr_maximo = gson.fromJson(data, double[].class);
41
42     IloCplex modelo = new IloCplex();
43     IloNumVar[] x = new IloNumVar[comidas];
44     IloNumVarType varType = IloNumVarType.Int;
45
46     for (int j = 0; j < comidas; j++) {
47         x[j] = modelo.numVar(comida_minima[j], comida_maxima[j],
48                               ↪ varType);
49     }
50     modelo.addMinimize(modelo.scalProd(costo_comida, x));
51
52     for (int i = 0; i < nutrientes; i++) {
53         modelo.addRange(nutr_minimo[i],
54                         modelo.scalProd(nutr_comida[i], x),
55                         nutr_maximo[i]);
56     }
57
58     modelo.setParam(IloCplex.Param.Tune.TimeLimit, 1800);
59
60     if (modelo.solve()) {
61         System.out.println("Status = " + modelo.getStatus());
62         System.out.println("Costo = " + modelo.getObjValue());
63         for (int i = 0; i < comidas; i++) {
64             System.out.println(" Comida " + i + " = " +
65                               ↪ modelo.getValue(x[i]));
66         }
67         modelo.end();
68     }
69 }

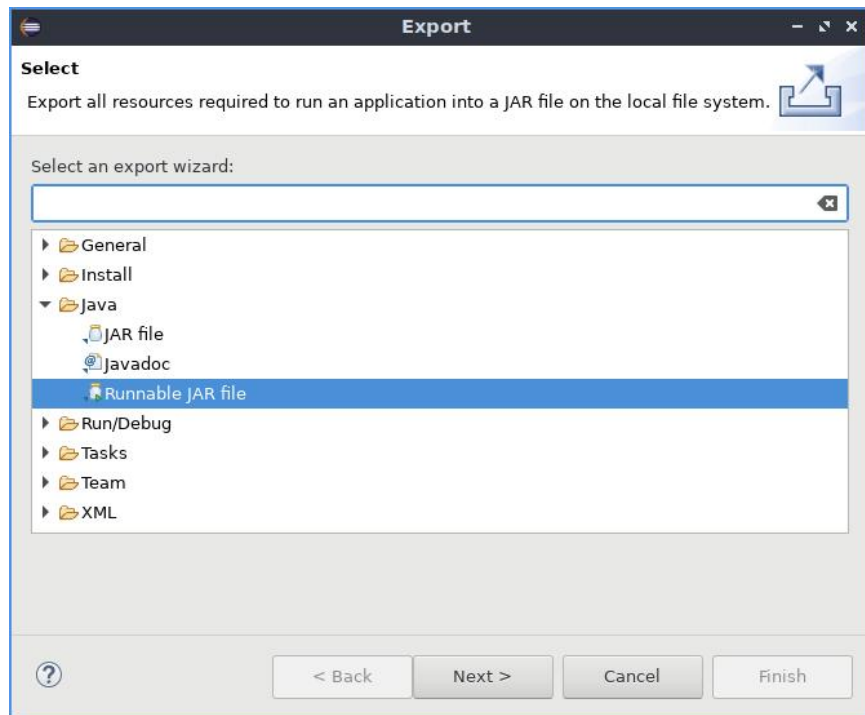
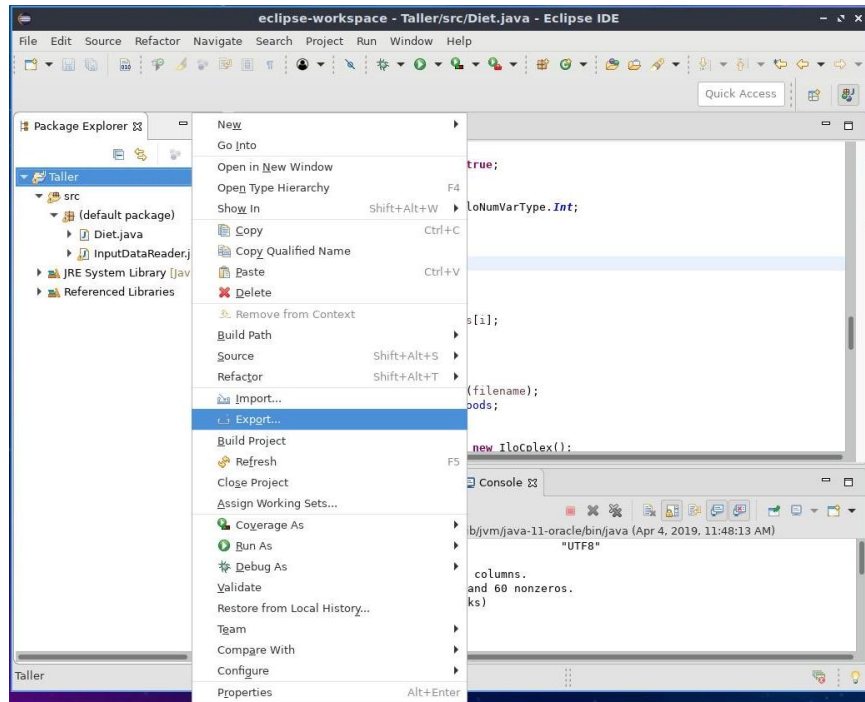
```

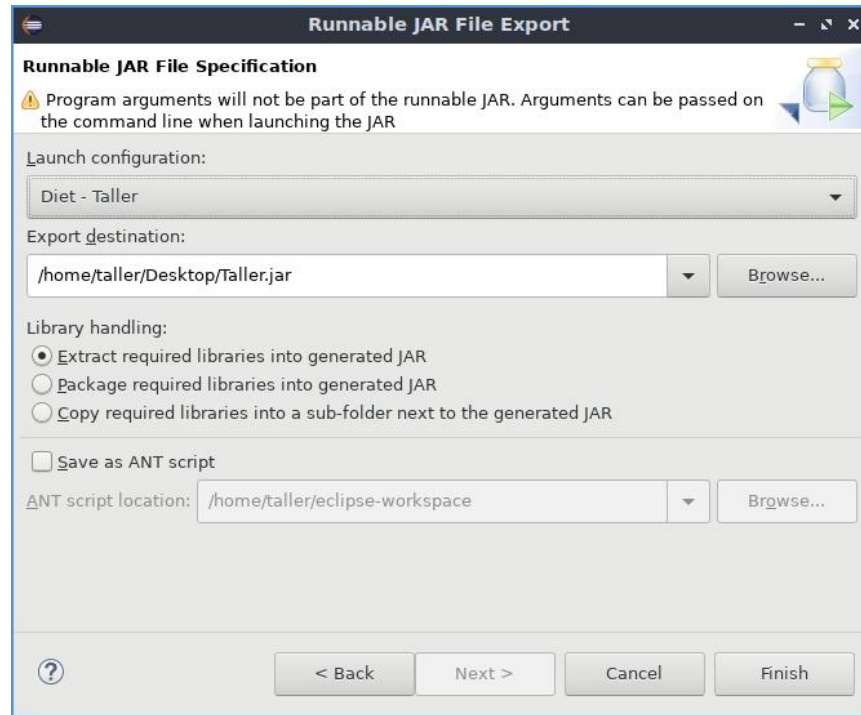
6.3 Generar ejecutable en Eclipse

Procedimiento para generar archivo ejecutable y portable en formato jar y como ejecutarlo desde una terminal.

6.3.1 Generar archivo jar







6.3.2 Ejecutar desde terminal

```
1 java -jar -Xms2048m -Djava.library.path=  
↪ /opt/ibm/ILOG/CPLEX_Studio1271/cplex/bin/x86-64_linux Taller.jar  
↪ "instancia.json"
```

- La opción `-Xms2048` aumenta la cantidad de memoria disponible para la máquina virtual de Java, en este caso a 2 GB.
- La ruta de la opción `-Djava.library.path` corresponde a la ruta de la instalación de CPLEX en la que se encuentra la librería `cplex.so`.

```
taller@taller-vm: ~/Desktop
File Actions Edit View Help
taller@taller-vm: ~/Desktop
taller@taller-vm:~/Desktop$ java -jar -Djava.library.path=/opt/ibm/ILOG/CPLEX_Studio1271/cplex/bin/x86-64_linux Taller.jar -c /opt/ibm/ILOG/CPLEX_Studio1271/cplex/examples/data/diet.dat
CPXPARAM_Read_APIEncoding "UTF8"
Tried aggregator 1 time.
LP Presolve eliminated 0 rows and 5 columns.
Reduced LP has 7 rows, 11 columns, and 60 nonzeros.
Presolve time = 0.00 sec. (0.01 ticks)
Initializing dual steep norms . . .

Iteration log . . .
Iteration: 1 Dual objective = 10.940000

Solution status = Optimal

cost = 14.855737704918033
Buy0 = 4.385245901639344
Buy1 = 0.0
Buy2 = 0.0
Buy3 = 0.0
Buy4 = 0.0
Buy5 = 6.147540983606555
Buy6 = 0.0
Buy7 = 3.4221311475409832
Buy8 = 0.0
```

7 MATERIAL ADICIONAL

- En el directorio `/opt/ibm/ILOG/CPLEX_Studio1271/cplex/examples/src` (depende de la versión instalada) se encuentran ejemplos más avanzados del uso de CPLEX, en Java, Python y otros lenguajes de programación. Para ejecutar proyectos de Java y Python se sigue el mismo procedimiento que el usado en este taller.
- En el sitio https://www.ibm.com/support/knowledgecenter/SSSA5P_12.9.0/ilog.odms.cplex.help/CPLEX/homepages/refparameterscplex.html (para la versión 12.9) se encuentra una lista de todos los parámetros que pueden ser modificados tal y como se hizo con el límite de tiempo en el ejemplo de la dieta para Java y Python.