DSA/ISE 5113
Advanced Analytics and Metaheuristics
Homework #2

February 8th, 2023

Claudia Corrales

Marshall Baldwin

Question 1: Grapes of Wrath, Inc. (60 points)

purchased at 28 cents per pound — 930,000 lbs. — 5,270,000

1) 6,200,000 pound crop. 15% Grade A. 85% Grade B.

Raisins can be sold infinitely. Juice max, 190,000. Jelly max, 210,000.

Raisins → \$8.29 → 6.5 pounds

Juice → \$16.20 → 14 pounds

Jelly → \$13.89 → 18 pounds

Grade A score: 9 per pound. Grade B score: 5 per pound.

a) Grimes must have done some variation on the following to find the upper bound on grapes that can be used for raisins:

Note that we want to use all Grade A grapes. This totals $(6{,}200{,}000) \times .15 = 930{,}000$ pounds. Next we determine how many pounds of Grade B grapes can be added to maintain a score of 8.

$$\frac{(1 \text{ pound grade A}) \times 9 + (x \text{ pound grade B}) \times 5}{1+x} = 8$$

Then $9 + 5x = 8 + 8x \rightarrow 1 = 3x \rightarrow x = \frac{1}{3}$. We can then add $(930{,}000) \times \frac{1}{3} = 310{,}000$ pounds of Grade B grapes. This gives a total of 930,000 pounds Grade A + 310,000 grade B = 1,240,000 pounds.

b) Bollman first computes the price per pound for Grade A and B, 45 and 25 cents respectively. For raisins, she then follows Grimes' procedure to find the ideal proportion of each: 1 pound A to 1/3 pound B. That is, 3/4 pound A to 1/4 pound B. She then finds the cost of a raisin product by multiplying the total weight (6.5 pounds) by the two proportions and multiplying by their respective costs:

$$\left(\frac{3}{4}\right)(6.5\,lb.)(\$.45\ lb.^{-1}) + \left(\frac{1}{4}\right)(6.5\,lb.)(\$.25\ lb^{-1}) = \$2.60$$

The same procedure is done for juice and jelly.

c) Assume that we want to have the lowest proportion of grade A grapes in each product. This allows us to not have to deal with a quadratic programming problem.
For raisins, we know .25 (grade B) and .75 (grade A) from b).

For juice: $\frac{(1)(9)+b(5)}{1+b}=6 \Rightarrow 9+5b=6+6b$

$3=b$

So for 1 pound of grade A grapes, there must be 3 lbs. of grade B.
That is .75 (grade B) and .25 (grade A) by proportion.

For jelly, use all grade B.

i) Jelly: 177037 lbs., Juice: 190000 lbs., Raisins: 450636 lbs.
ii) Juice has the max contribution to profit of \$3,078,000
iii) There were no grapes leftover.
iv) As per our assumption, Jelly: 5, Juice: 6, and Raisins: 8.
v) Yes, they should! Profit increases by over \$200,000. (See output).
vi) Manually plugging in values for purchasing price, I find \$1.44 to be the maximum price the company should pay for additional grade A grapes. Following the same adjustments to the model for grade B (also taking away the ability to purchase grade A grapes), I find \$.77 to be the max price that the company should pay for a pound of grade B grapes.

LP Model Formulation with objective function and constraints:

$$\text{Maximize} \quad \sum_{p=1}^{P_r} x_p P_p$$

$$\text{s.t.} \quad x_p \geq 0 \; p$$

$$W_A \geq \sum_{p=1} Aprop_p \cdot w_p \cdot x_p$$

$$W_B \geq \sum_{p=1}^{P} (1-Aprop_p) \cdot w_p \cdot x_p$$

$$d_p \geq x_p$$

Where x_p = Pounds of product $p \in P_r$

P_p = Selling price of product p

W_A, W_B = Total pounds of grade A, B grapes

w_p = Pounds of grapes required for producing product p.

d_p = Maximum number of units that should be produced of product p

$Aprop_p$ = Minimum proportion of total weight that must be grade A to satisfy quality requirement for product P.

In plain English, we want to maximize total revenue. This is subject to a non-negativity constraint (we assume there is no such thing as a negative number of products produced); weight constraints, stipulating that the total amount of each grape grade must not exceed the total amount available; and a demand constraint, stipulating that the total amount of each product produced must not exceed a demand threshold.

AMPL Code, Part C (i-iv):

```
#clear memory
reset;

#choose solver;
option solver cplex;

#Set problem parameters
set Products; #grape products: raisins, juice, and jelly

param demand {p in Products}; #the maximum number of units that can be sold for each product
param weight {p in Products}; #the weight in pounds of grapes required to make each product
param price {p in Products}; #the selling price for each product
param A_prop {p in Products} ; #the proportion of grade A per pound for each product to guarantee quality

param A_grapes; #pounds of grade A grapes
param B_grapes; #pounds of grade B grapes

#Set variables to solve for
var numProduct {p in Products}; #number of each product to manufacture
var profits {p in Products} = numProduct[p] * price[p];
var A_grape_remainder = A_grapes - sum {p in Products} (A_prop[p] * weight[p] * numProduct[p]);
var B_grape_remainder = B_grapes - sum {p in Products} ((1 - A_prop[p]) * weight[p] * numProduct[p]);

#Set function to maximize
maximize totalProfit: sum {p in Products} numProduct[p] * price[p];

#Set constraints
s.t. nonNegative {p in Products}: numProduct[p] >= 0; #no such thing as negative product by assumption
s.t. A_weightConstraint: sum {p in Products} (A_prop[p] * weight[p] * numProduct[p]) <= A_grapes;
s.t. B_weightConstraint: sum {p in Products} ((1 - A_prop[p]) * weight[p] * numProduct[p]) <= B_grapes;
s.t. proportionBound {p in Products}: 0 <= A_prop[p] <= 1;
s.t. demandConstraint {p in Products}: demand[p] >= numProduct[p];

#load data in
data HW2/HW2_Q1_data.txt

#solve the model
solve;

#Display nicely
printf "Solving the model we can see that the following number of each product should be made: \n";
display numProduct;

printf "This creates profits of: \n";
display totalProfit;
display profits;

printf "The following amounts of grapes are leftover: \n";
display A_grape_remainder;
display B_grape_remainder;
```

AMPL Output: Part C (i-iv):

```
ampl: model HW2/HW2_Q1_model.txt
CPLEX 20.1.0.0: optimal solution; objective 5987680.342
2 dual simplex iterations (0 in phase I)
Solving the model we can see that the following number of each product should be made:
numProduct [*] :=
  jelly  177037
  juice  190000
raisins   54359
;

This creates profits of:
totalProfit = 5987680

profits [*] :=
  jelly  2459040
  juice  3078000
raisins   450636
;

The following amounts of grapes are leftover:
A_grape_remainder = 0

B_grape_remainder = 0
```

AMPL Code, Part C (v):

```
#Set variables to solve for
var numProduct {p in Products}; #number of each product to manufacture
var purchased_A_grapes <= 300000;
var profits {p in Products} = numProduct[p] * price[p];
var A_grape_remainder = A_grapes + purchased_A_grapes - sum {p in Products} (A_prop[p] * weight[p] * numProduct[p]);
var B_grape_remainder = B_grapes - sum {p in Products} ((1 - A_prop[p]) * weight[p] * numProduct[p]);


#Set function to maximize
maximize totalProfit: -purchased_A_grapes * .5 + sum {p in Products} numProduct[p] * price[p];

#Set constraints
s.t. nonNegative {p in Products}: numProduct[p] >= 0; #no such thing as negative product by assumption
s.t. nonNegativePurchase: purchased_A_grapes >= 0;
```

Here I add an additional variable, purchased_A_grapes, which subtracts 50 cents per pound purchased from the total profit objective function.

AMPL Output: Part C (v):

```
ampl: model HW2/HW2_Q1_model.txt
CPLEX 20.1.0.0: optimal solution; objective 6270667.521
2 dual simplex iterations (0 in phase I)
Solving the model we can see that the following number of each product should be made:
numProduct [*] :=
  jelly  171481
  juice  190000
raisins  115897
;

This creates profits of:
totalProfit = 6270670

profits [*] :=
  jelly  2381880
  juice  3078000
raisins   960790
;

The following amounts of grapes are leftover:
A_grape_remainder = 0

B_grape_remainder = 0

The following amount of additional grade A grapes were purchased:
purchased_A_grapes = 3e+05
```

AMPL Code, Part C (vi):

```
#clear memory
reset;

#choose solver;
option solver cplex;

#Set problem parameters
set Products; #grape products: raisins, juice, and jelly

param demand {p in Products}; #the maximum number of units that can be sold for each product
param weight {p in Products}; #the weight in pounds of grapes required to make each product
param price {p in Products}; #the selling price for each product
param A_prop {p in Products} ; #the proportion of grade A per pound for each product to guarantee quality

param A_grapes; #pounds of grade A grapes
param B_grapes; #pounds of grade B grapes

#Set variables to solve for
var numProduct {p in Products}; #number of each product to manufacture
var purchased_B_grapes <= 300000;
var profits {p in Products} = numProduct[p] * price[p];
var A_grape_remainder = A_grapes  - sum {p in Products} (A_prop[p] * weight[p] * numProduct[p]);
var B_grape_remainder = B_grapes + purchased_B_grapes - sum {p in Products} ((1 - A_prop[p]) * weight[p] * numProduct[p]);


#Set function to maximize
maximize totalProfit: -purchased_B_grapes * .78 + sum {p in Products} numProduct[p] * price[p];

#Set constraints
s.t. nonNegative {p in Products}: numProduct[p] >= 0; #no such thing as negative product by assumption
s.t. nonNegativePurchase: purchased_B_grapes >= 0;
s.t. A_weightConstraint: sum {p in Products} (A_prop[p] * weight[p] * numProduct[p]) <= A_grapes;
s.t. B_weightConstraint: sum {p in Products} ((1 - A_prop[p]) * weight[p] * numProduct[p]) <= B_grapes + purchased_B_grapes ;
s.t. proportionBound {p in Products}: 0 <= A_prop[p] <= 1;
s.t. demandConstraint {p in Products}: demand[p] >= numProduct[p];

#load data in
data HW2/HW2_Q1_data.txt

#solve the model
solve;

#Display nicely
printf "Solving the model we can see that the following number of each product should be made: \n";
display numProduct;

printf "This creates profits of: \n";
display totalProfit;
display profits;

printf "The following amounts of grapes are leftover: \n";
display A_grape_remainder;
display B_grape_remainder;

printf "The following amount of additional grade B grapes were purchased: \n";
display purchased_B_grapes;
```

Note that this code displays that at a threshold of 78 cents per pound, the optimal solution will purchase no additional grade B grapes. If this amount is changed to 77 cents per pound in the objective function, the grade B grapes will then be purchased. This was found by using a manual binary search of cost to refine bounds where cost is above and below the inflection point between purchase and no purchase. When one cent was the difference between the bounds, I stopped the process. The same procedure was followed for finding the maximum viable price for purchasing additional grade A grapes.

AMPL Output: Part C (vi):

```
ampl: model HW2/HW2_Q1_model_c.txt
CPLEX 20.1.0.0: optimal solution; objective 5987680.342
2 dual simplex iterations (0 in phase I)
Solving the model we can see that the following number of each product should be made:
numProduct [*] :=
  jelly  177037
  juice  190000
raisins   54359
;

This creates profits of:
totalProfit = 5987680

profits [*] :=
  jelly  2459040
  juice  3078000
raisins   450636
;

The following amounts of grapes are leftover:
A_grape_remainder = 0

B_grape_remainder = 0

The following amount of additional grade B grapes were purchased:
purchased_B_grapes = 0
```

d) i) For Thomas' contribution figures, the following product amounts were attained: Jelly, 210,000; Juice, 0; and Raisins, 252308. The revenue generated by Thomas' solution ($4,498,380) is much less than the revenue generated by the Part c solution ($5,987,680). Following the same manual approach as in c) vi), I found that the maximum price for grade A grapes is $.07 per pound under Thomas' figures.

ii) Using Bollman's profit figures yields the following product amounts: Jelly, 177037; Juice, 190000; and Raisins, 54359. The revenue is the same as my LP model at $5,987,680 with a profit of $959,540. Interestingly, the maximum price for grade A grapes using Bollman's figures is also $.07 per pound.

iii) I believe that the approach that Bollman and I took is most promising, since we each arrived at it in different ways and it appears to generate more revenue than Thomas'. However, my LP approach was naive in that I maximized revenue without considering variable costs. Nonetheless, if the variable costs assessed by Bollman are accurate, my solution is still ideal, with the caveat being that no more grapes should be purchased.

AMPL Code, Part D (i):

```
#clear memory
reset;

#choose solver;
option solver cplex;

#Set problem parameters
set Products; #grape products: raisins, juice, and jelly

param demand {p in Products}; #the maximum number of units that can be sold for each product
param weight {p in Products}; #the weight in pounds of grapes required to make each product
param price {p in Products}; #the selling price for each product
param profit {p in Products}; #Thomas' figures for each product's profit
param A_prop {p in Products} ; #the proportion of grade A per pound for each product to guarantee quality

param A_grapes; #pounds of grade A grapes
param B_grapes; #pounds of grade B grapes

#Set variables to solve for
var numProduct {p in Products}; #number of each product to manufacture
var profits {p in Products} = numProduct[p] * price[p];
var purchased_A_grapes <= 300000;
var A_grape_remainder = A_grapes + purchased_A_grapes - sum {p in Products} (A_prop[p] * weight[p] * numProduct[p]);
var B_grape_remainder = B_grapes - sum {p in Products} ((1 - A_prop[p]) * weight[p] * numProduct[p]);


#Set function to maximize
maximize totalProfit: - purchased_A_grapes * .07 + sum {p in Products} numProduct[p] * profit[p];

#Set constraints
s.t. nonNegative {p in Products}: numProduct[p] >= 0; #no such thing as negative product by assumption
s.t. nonNegativePurchase: purchased_A_grapes >= 0;
s.t. A_weightConstraint: sum {p in Products} (A_prop[p] * weight[p] * numProduct[p]) <= A_grapes + purchased_A_grapes;
s.t. B_weightConstraint: sum {p in Products} ((1 - A_prop[p]) * weight[p] * numProduct[p]) <= B_grapes ;
s.t. demandConstraint {p in Products}: demand[p] >= numProduct[p];

#load data in
data HW2/HW2_Q1_di_data.txt

#solve the model
solve;

#Display nicely
printf "Solving the model we can see that the following number of each product should be made: \n";
display numProduct;

printf "This creates revenue of: \n";
display sum {p in Products} numProduct[p] * price[p];
display profits;

printf "The following amounts of grapes are leftover: \n";
display A_grape_remainder;
display B_grape_remainder;

printf "The following amount of additional grade B grapes were purchased: \n";
display purchased_A_grapes;
```

AMPL Output: Part D (i):

```
ampl: model HW2/HW2_Q1_di_model.txt
CPLEX 20.1.0.0: optimal solution; objective 157607.6923
0 dual simplex iterations (0 in phase I)
Solving the model we can see that the following number of each product should be made:
numProduct [*] :=
  jelly  210000
  juice       0
raisins  252308
;

This creates revenue of:
sum{p in Products} numProduct[p]*price[p] = 5008530

profits [*] :=
  jelly  2916900
  juice        0
raisins  2091630
;

The following amounts of grapes are leftover:
A_grape_remainder = 0

B_grape_remainder = 1080000

The following amount of additional grade B grapes were purchased:
purchased_A_grapes = 3e+05
```

AMPL Code, Part D (ii):

```
#clear memory
reset;

#choose solver;
option solver cplex;

#Set problem parameters
set Products; #grape products: raisins, juice, and jelly

param demand {p in Products}; #the maximum number of units that can be sold for each product
param weight {p in Products}; #the weight in pounds of grapes required to make each product
param price {p in Products}; #the selling price for each product
param profit {p in Products}; #Thomas' figures for each product's profit
param A_prop {p in Products} ; #the proportion of grade A per pound for each product to guarantee quality

param A_grapes; #pounds of grade A grapes
param B_grapes; #pounds of grade B grapes

#Set variables to solve for
var numProduct {p in Products}; #number of each product to manufacture
var profits {p in Products} = numProduct[p] * price[p];
var purchased_A_grapes <= 300000;
var A_grape_remainder = A_grapes + purchased_A_grapes - sum {p in Products} (A_prop[p] * weight[p] * numProduct[p]);
var B_grape_remainder = B_grapes - sum {p in Products} ((1 - A_prop[p]) * weight[p] * numProduct[p]);


#Set function to maximize
maximize totalProfit: - purchased_A_grapes * .07 + sum {p in Products} numProduct[p] * profit[p];

#Set constraints
s.t. nonNegative {p in Products}: numProduct[p] >= 0; #no such thing as negative product by assumption
s.t. nonNegativePurchase: purchased_A_grapes >= 0;
s.t. A_weightConstraint: sum {p in Products} (A_prop[p] * weight[p] * numProduct[p]) <= A_grapes + purchased_A_grapes;
s.t. B_weightConstraint: sum {p in Products} ((1 - A_prop[p]) * weight[p] * numProduct[p]) <= B_grapes ;
s.t. demandConstraint {p in Products}: demand[p] >= numProduct[p];

#load data in
data HW2/HW2_Q1_d_ii_data.txt

#solve the model
solve;

#Display nicely
printf "Solving the model we can see that the following number of each product should be made: \n";
display numProduct;

printf "This creates revenue of: \n";
display sum {p in Products} numProduct[p] * price[p];
display totalProfit;
display profits;

printf "The following amounts of grapes are leftover: \n";
display A_grape_remainder;
display B_grape_remainder;

printf "The following amount of additional grade B grapes were purchased: \n";
display purchased_A_grapes;
```

AMPL Output: Part D (i):

```
ampl: model HW2/HW2_Q1_d_ii_model.txt
CPLEX 20.1.0.0: optimal solution; objective 960637.8917
2 dual simplex iterations (0 in phase I)
Solving the model we can see that the following number of each product should be made:
numProduct [*] :=
  jelly  171481
  juice  190000
raisins  115897
;

This creates revenue of:
sum{p in Products} numProduct[p]*price[p] = 6420670

totalProfit = 960638

profits [*] :=
  jelly  2381880
  juice  3078000
raisins   960790
;

The following amounts of grapes are leftover:
A_grape_remainder = 0

B_grape_remainder = 0

The following amount of additional grade B grapes were purchased:
purchased_A_grapes = 3e+05
```

Question 2: Titan Enterprises Case Study (40 points)

(a) Formulate and solve Titan's investment decision problem as a linear program (use AMPL).

Constraints:

 # Investment limitations: The investment limit of project A (subject to A_limit: xA <= 500000;), investment limit of investment B (subject to B_limit: xB <= 500000;) and investment limit of investment E (subject to E_limit: xE <= 750000;).

The investment conditions for 2021, 2022 and 2023

- **2021**: the total investments are equal to the money invested in investment A, C, D and/or saving in the Bank. subject to 2021_Inv: init_Inv = xA + xC + xD + xBK1;

- **2022**: the interest earned from investment A, investment C and Bank savings are equal to the money invested in investment B and/or saving in the Bank. subject to 2022_Inv: xA * Aint + xC * (1 + Cint) + xBK1 * (1 + BKint) = xB + xBK2;

- **2023**: the returned investment from investment A, interest earned from investment B and Bank savings are equal to the money invested in investment E and/or saving in the Bank. subject to 2023_Inv: xA * 1 + xB * Bint + xBK2 * (1 + BKint) = xE + xBK3;

AMPL:

```
AMPL
ampl: #Group 22, DSA 5113, Spring 2023

# Titan Enterprises Case Study

reset;

#set-up options

option solver cplex;
option cplex_options 'sensitivity';

#parameters and sets

param init_Inv;          # initial investments in 2021
param Aint;              # interest rate of project A
param Bint;              # interest rate of project B
param Cint;              # interest rate of project C
param Dint;              # interest rate of project D
param Eint;              # interest rate of project E
param BKint;             # interest rate of Bank savings

#decision variables

var xA >= 0;             # investments in project A
var xB >= 0;             # investments in project B
var xC >= 0;             # investments in project C
var xD >= 0;             # investments in project D
var xE >= 0;             # investments in project E
var xBK1 >= 0;           # saving in the Bank in 2021
var xBK2 >= 0;           # saving in the Bank in 2022
var xBK3 >= 0;           # saving in the Bank in 2023
```

```
#objective

maximize Profit: xB * 1 + xD * (1 + Dint) + xE * (1 + Eint) +
                                 xBK3 * (1 + BKint) - init_Inv;
                # overall investment profits at the beginning of 2024

#constraints

# investment limitations
subject to A_limit: xA <= 600000;
subject to B_limit: xB <= 500000;
subject to E_limit: xE <= 750000;

subject to 2021_Inv: init_Inv = xA + xC + xD + xBK1;
subject to 2022_Inv: xA * Aint + xC * (1 + Cint) + xBK1 * (1 + BKint) = xB + xBK2;
subject to 2023_Inv: xA * 1 + xB * Bint + xBK2 * (1 + BKint) = xE + xBK3;

#data file
#data Q2a.dat;
data 'C:/Users/CC0481/OneDrive - AT&T Services, Inc/University of Oklahoma/DSA_5113/HW/HW#2/Q2a.dat';

solve;

display Profit;

display xA, xB, xC, xD, xE, xBK1, xBK2, xBK3;
```

Data:

welcome.txt | Q2a | .project | discussionEn... | Q2a.mod | *Q2.mod

```
# Titan Enterprises Case Study

param init_Inv := 1000000;         # initial investments in 2021

param Aint  := 0.30;               # interest rate of project A
param Bint  := 0.30;               # interest rate of project B
param Cint  := 0.10;               # interest rate of project C
param Dint  := 0.75;               # interest rate of project D
param Eint  := 0.40;               # interest rate of project E
param BKint := 0.06;               # interest rate of Bank savings

data;
```

Results:

- The profits are **$797,600** for the overall investment portfolio.
- With the initial investment of 1 million dollars, $500,000 are invested in investment A and $500,000 are invested in investment D at beginning of 2021.
- The investment interests of $150,000 received from investment A are saved in the Bank at beginning of 2022 and withdraw at beginning of 2023
- $659,000 consists of the interest earned from the Bank saving in 2022 plus the return of investment A investments at the beginning of 2023 are invested in investment E at beginning of 2023.

```
/* AMPL Execution
ampl: model Q2a.txt;
CPLEX 20.1.0.0: sensitivity
CPLEX 20.1.0.0: optimal solution; objective 797600
4 dual simplex iterations (3 in phase I)

suffix up OUT;
suffix down OUT;
suffix current OUT;
Profit = 797600

xA = 5e+05
xB = 0
xC = 0
xD = 5e+05
xE = 659000
xBK1 = 0
xBK2 = 150000
xBK3 = 0
*/
```

(b) Provide an interpretation of the shadow prices in the specific context of the Titan investment problem

```
ampl: display A_limit, A_limit.down, A_limit.up;
A_limit = 0.0952
A_limit.down = 0
A_limit.up = 0

ampl: display B_limit, B_limit.down, B_limit.up;
B_limit = 0
B_limit.down = 0
B_limit.up = 0

ampl: display E_limit, E_limit.down, E_limit.up;
E_limit = 0
E_limit.down = 0
E_limit.up = 0
```

- If we can invest more money in investment A, we will gain $0.0952 per one dollar invested in A
- We need to find the upper limit as we invest more in A but gain less than $0.0952
- We increase the investment limit in A to $550,000, the profits are increased to $802,360 (from $797,600). It is an increase of $4,760 with $50,000 more investment in A
- it confirms the shadow price of **0.0952** for investment A.

```
# 550,000 limit in A
ampl: display A_limit, A_limit.down, A_limit.up;
A_limit = 0.0952
A_limit.down = 0
A_limit.up = 0

ampl: display B_limit, B_limit.down, B_limit.up;
B_limit = 0
B_limit.down = 0
B_limit.up = 0

ampl: display E_limit, E_limit.down, E_limit.up;
E_limit = 0
E_limit.down = 0
E_limit.up = 0
```

- The profits are increased to $804,173 (from $797,600) when we further increase the investment in A to **$600,000**
- It is an increase of $6,573 with $100,000 more investment in A
- The shadow price decreases to $0.06573 for investment A
- The investment of A is at $569044 which is the most we can invest in it and get the more profits from the portfolio.
- We also found out that the shadow price of investment E is increased to 0.0722307 (from 0) as we increase the investment in A and decrease the investment in other investments.

```
# 600,000 limit in A
ampl: display A_limit, A_limit.down, A_limit.up;
A_limit = 0
A_limit.down = 0
A_limit.up = 0

ampl: display B_limit, B_limit.down, B_limit.up;
B_limit = 0
B_limit.down = 0
B_limit.up = 0

ampl: display E_limit, E_limit.down, E_limit.up;
E_limit = 0.0722307
E_limit.down = 0
E_limit.up = 0
```

- 10% hurdle rate is slightly higher that what we see from the shadow price of investment A (0.0952) and investment E (**0.0722307**).

(c) How might you determine how sensitive the investment decision is to changes in the projects' final payouts?

There is no change on the overall investment combinations/progresses after the interest rate change on both investment D & E. Profits decrease (from $797,600 to $733,060) due to the reduction on the interest payout from investment D & E.

Solution:

Changed the interest rate of investment D and E in the data file

- param Dint := 0.70; # interest rate of project D
- param Eint := 0.34; # interest rate of project E

```
#Group 22, DSA 5113, Spring 2023

# Titan Enterprises Case Study
# Investment change in D & E

param init_Inv := 1000000;          # initial investments in 2021

param Aint  := 0.30;                # interest rate of project A
param Bint  := 0.30;                # interest rate of project B
param Cint  := 0.10;                # interest rate of project C
param Dint  := 0.70;                # interest rate of project D
param Eint  := 0.34;                # interest rate of project E
param BKint := 0.06;                # interest rate of Bank savings
```

```
/* AMPL Execution
ampl: model Q2c.txt;
CPLEX 20.1.0.0: sensitivity
CPLEX 20.1.0.0: optimal solution; objective 733060
4 dual simplex iterations (3 in phase I)

suffix up OUT;
suffix down OUT;
suffix current OUT;
Profit = 733060

xA = 5e+05
xB = 0
xC = 0
xD = 5e+05
xE = 659000
xBK1 = 0
xBK2 = 150000
xBK3 = 0
```

(d) How might you use results to determine whether new projects should be included in the portfolio? Would you recommend that the portfolio be changed if F were available?

- The profits increase to \$802,311 (from \$797,600) if investment F is available
- The profits increase to \$800,129 if investment G is available
- The profits increase to \$802,311 if both investment F and G are available
- The overall investment combinations are the same for the case of investment F only and the case of both investment F and G are available
- In conclusion, that we only need to pick the portfolio with only investment F available

Overall investment combinations/progresses with investment F in the portfolio:

- With the initial investment of 1 million dollars, \$500,000 are invested in investment A, \$429,892 are invested in investment D and \$70107.9 are invested in investment F at beginning of 2021
- The investment interests of \$150,000 received from investment A and the return of \$56,086.32 from the investment F are saved in the Bank at beginning of 2022 and withdraw at beginning of 2023
- \$750,000 consists of the interest earned from the Bank saving in 2022 plus the interest earned with investment return of investment A as well as the investment return of investment F at the beginning of 2023 are invested in investment E at beginning of 2023

Solution:

a) Add a new variable for the investment F and G (var xF >= 0; # investments in project F; var xG >= 0; # investments in project G);
b) Add a new parameter for the interest of investment F and G (param Fint1; # 1st interest rate of project F; param Fint2; # 2nd interest rate of project F; param Gint1; # 1st interest rate of project G; param Gint2; # 2nd interest rate of project G);
c) Add the investment F and G to the constraints for each year based on the investment timeline (Table 4); and
d) Add the return of investment to the objective function (maximize Profit: xB * 1 + xD * (1 + Dint) + xE * (1 + Eint) + xG * Gint2 + xBK3 * (1 + BKint) - init_Inv;) for the cases when investment G only as well as investment F & G both available.

In the data file, add the interest rate of investment F and G

```
#Group 22, DSA 5113, Spring 2023

# Titan Enterprises Case Study
# Adding investment F and G

param init_Inv := 1000000;         # initial investments in 2021

param Aint  := 0.30;               # interest rate of project A
param Bint  := 0.30;               # interest rate of project B
param Cint  := 0.10;               # interest rate of project C
param Dint  := 0.75;               # interest rate of project D
param Eint  := 0.40;               # interest rate of project E
param BKint := 0.06;               # interest rate of Bank savings

param Fint1  := 0.80;              # interest rate of project F
param Fint2  := 0.45;              # interest rate of project F
param Gint1  := 0.10;              # interest rate of project G
param Gint2  := 0.15;              # interest rate of project G
```

Results:

- ✓ Investment F only

```
/* AMPL Execution
# investment F Only
ampl: model Q2d.txt;
CPLEX 20.1.0.0: sensitivity
CPLEX 20.1.0.0: optimal solution; objective 802311.2481
5 dual simplex iterations (3 in phase I)

suffix up OUT;
suffix down OUT;
suffix current OUT;
Profit = 802311

xA = 5e+05
xB = 0
xC = 0
xD = 429892
xE = 750000
xF = 70107.9
xBK1 = 0
xBK2 = 206086
xBK3 = 0
*/
```

- ✓ Investment G only

```
/* AMPL Execution
# investment G Only
ampl: model Q2d.txt;
CPLEX 20.1.0.0: sensitivity
CPLEX 20.1.0.0: optimal solution; objective 800128.6449
3 dual simplex iterations (2 in phase I)

suffix up OUT;
suffix down OUT;
suffix current OUT;
Profit = 800129

xA = 5e+05
xB = 0
xC = 0
xD = 421955
xE = 750000
xG = 78044.6
xBK1 = 0
xBK2 = 235849
xBK3 = 0
*/
```

- ✓ Investment F & G

```
/* AMPL Execution
# investment F & G
ampl: model Q2d.txt;
CPLEX 20.1.0.0: sensitivity
CPLEX 20.1.0.0: optimal solution; objective 802311.2481
3 dual simplex iterations (2 in phase I)

suffix up OUT;
suffix down OUT;
suffix current OUT;
Profit = 802311

xA = 5e+05
xB = 0
xC = 0
xD = 429892
xE = 750000
xF = 70107.9
xG = 0
xBK1 = 0
xBK2 = 206086
xBK3 = 0
*/
```

(e) Assuming that Project F is available, use the computer output to consider the sensitivity of the portfolio decision to the changes in projects D and E considered in question (c). Would the portfolio change if Project E pays only $1.34 per dollar invested? How? Does that make sense? Would the portfolio change if D pays only $1.70 per dollar invested (and E retains original payout)? How? Does that make sense?

- The profits are $ 758,060 for the overall investment portfolio
- The initial investment of 1 million dollars, $500,000 are invested in investment A and $500,000 are invested in investment D at beginning of 2021
- The investment interests of $150,000 received from investment A are saved in the Bank at beginning of 2022 and withdraw at beginning of 2023
- $659,000 consists of the interest earned from the Bank saving in 2022 plus the return of investment A investments at the beginning of 2023 are invested in investment E at beginning of 2023

Solution: Changed the interest rate of investment D

```
#Group 22, DSA 5113, Spring 2023

# Titan Enterprises Case Study
# Investment change in D & E

param init_Inv := 1000000;        # initial investments in 2021

param Aint  := 0.30;              # interest rate of project A
param Bint  := 0.30;              # interest rate of project B
param Cint  := 0.10;              # interest rate of project C
#param Dint  := 0.75;              # interest rate of project D
param Dint  := 0.70;              # interest rate change
param Eint  := 0.40;              # interest rate of project E
#param Eint  := 0.34;              # interest rate change
param BKint := 0.06;              # interest rate of Bank savings

param Fint1  := 0.80;             # interest rate of project F
param Fint2  := 0.45;             # interest rate of project F
```

Results:

```
/* AMPL Execution
# investment change in D Only
ampl: model Q2e.txt;
CPLEX 20.1.0.0: sensitivity
CPLEX 20.1.0.0: optimal solution; objective 782608.6957
6 dual simplex iterations (3 in phase I)

suffix up OUT;
suffix down OUT;
suffix current OUT;
Profit = 782609

xA = 5e+05
xB = 387681
xC = 0
xD = 202899
xE = 750000
xF = 297101
xBK1 = 0
xBK2 = 0
xBK3 = 0
*/ampl:
```