Pontificia Universidad Católica de Chile
Departamento de Ingenieria Industrial y de Sistemas
ICS1113- Optimización

# Tarea 1

**Integrantes: Jacinta Ortiz y Vicente Lavagnino**

---

## Índice

# 1. Pregunta 1:

## 1). Simplex y Programación lineal

### a. Solución gráfica

En esta imagen, podemos ver como dominio de la solución, toda el área contenida por el polígono formado con los vértices $VF_1$, $VF_2$, $VF_3$, $VF_4$ y $VF_5$.

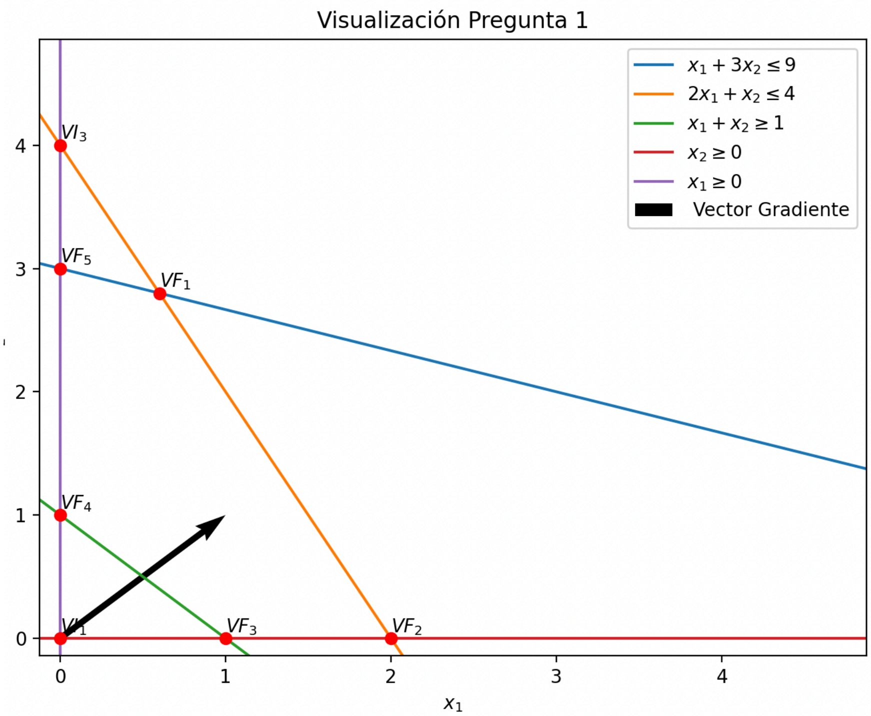



Gráfico 1 : Visualización Pregunta 1.

En la sección de Anexos Pregunta 1, se incorporan más vistas donde se incluyen los puntos de las tablas que no son visualizados en el dominio de la imagen. Además se incluye el script de python utilizado para hacer la visualización

| Vértice | Coordenadas $(x_1, x_2)$ | Restricciones que se intersectan |
|---|---|---|
| $VF_1$ | (0.6, 2.8) | $x_1 + 3x_2 \leq 9$, $2x_1 + x_2 \leq 4$ |
| $VF_2$ | (2, 0) | $2x_1 + x_2 \leq 4$, $x_1 \geq 0$ |
| $VF_3$ | (1, 0) | $x_1 + x_2 \geq 1$, $x_1 \geq 0$ |
| $VF_4$ | (0, 1) | $x_1 + x_2 \geq 1$, $x_2 \geq 0$ |
| $VF_5$ | (0, 3) | $x_1 + 3x_2 \leq 9$, $x_2 \geq 0$ |

Cuadro 1: Vértices de la región factible y sus restricciones

| Vértice | Coordenadas $(x_1, x_2)$ | Restricciones que se intersectan |
|---|---|---|
| $VI_1$ | (0, 0) | $x_1 \geq 0$, $x_2 \geq 0$ |
| $VI_2$ | (3, -2) | $2x_1 + x_2 \leq 4$, $x_1 + x_2 \geq 1$ |
| $VI_3$ | (0, 4) | $2x_1 + x_2 \leq 4$, $x_1 \geq 0$ |
| $VF_4$ | (9, 0) | $x_1 + 3x_2 \leq 9$, $x_2 \geq 0$ |

Cuadro 2: Vértices de la región infactible y sus restricciones

De esta forma, visualmente identificamos que la **solución óptima es (0.6, 2.8)** y que el **valor óptimo es 3.4**. En este mismo sentido, logramos ver que las variables que se activan son $x_1 + 3x_2 \leq 9$ y $2x_1 + x_2 \leq 4$.

### b. Forma estándar

Para encontrar la forma estándar realizaremos ciertas modificaciones al problema inicial:

- Ocupamos equivalencias para transformar el problema incial a uno de minimización. Para ello, multiplicamos por -1 la función, obteniendo:

$$\begin{aligned} &\min -x_1 - x_2 \\ &\quad \text{s.a} \quad x_1 + 3x_2 \leqslant 9 \\ &\qquad\qquad 2x_1 + x_2 \leqslant 4 \\ &\qquad\qquad x_1 + x_2 \geqslant 1 \\ &\qquad\qquad x_1, x_2 \geqslant 0 \end{aligned}$$

- Ahora incluímos las variables de holgura: $\{x3, x4\}$ y la variable de exceso: $\{x5\}$, obteniendo:

$$\begin{aligned} &\min -x_1 - x_2 \\ &\quad \text{s.a} \quad x_1 + 3x_2 + x_3 = 9 \\ &\qquad\qquad 2x_1 + x_2 + x_4 = 4 \\ &\qquad\qquad x_1 + x_2 - x_5 = 1 \\ &\qquad\qquad x_1, x_2, x_3, x_4, x_5 \geqslant 0 \end{aligned}$$

- A partir de esto, podemos definir lo siguiente:

$$A = \begin{bmatrix} 1 & 3 & 1 & 0 & 0 \\ 2 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & -1 \end{bmatrix},\ b = \begin{bmatrix} 9 \\ 4 \\ 1 \end{bmatrix},\ C^\top = \begin{bmatrix} -1 & -1 & 0 & 0 & 0 \end{bmatrix}$$

### c. Simplex Fase I

Para pasar nuestro problema a Fase I, le agregamos la variable auxiliar $t$. Quedando definido de la siguiente manera:

$$\begin{aligned} &\min t \\ &\text{s.a} \quad x_1 + 3x_2 + x_3 = 9 \\ &\qquad 2x_1 + x_2 + x_4 = 4 \\ &\qquad x_1 + x_2 - x_5 + t = 1 \\ &\qquad x_1, x_2, x_3, x_4, x_5, t \geqslant 0 \end{aligned}$$

Con $A$ y $C^\top$ :

$$A = \begin{bmatrix} 1 & 3 & 1 & 0 & 0 & 0 \\ 2 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & -1 & 1 \end{bmatrix},\ b = \begin{bmatrix} 9 \\ 4 \\ 1 \end{bmatrix},\ C^\top = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

**Iteración 1:** Partimos escogiendo las variables para nuestra base :
Base: $\{x_3, x_4, t\}$
Variables No básicas: $\{x_1, x_2, x_5\}$

- Sabiendo esto, podemos determinar las siguientes matrices:

$$B = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, B^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

  Como se puede observar la matriz $B$ es una matriz identidad. Por lo tanto, $B^{-1} = B$. Luego definimos $R$ como la matriz con las variables no básicas:

$$R = \begin{bmatrix} 1 & 3 & 0 \\ 2 & 1 & 0 \\ 1 & 1 & -1 \end{bmatrix}, \bar{R} = B^{-1} \cdot R = \begin{bmatrix} 1 & 3 & 0 \\ 2 & 1 & 0 \\ 1 & 1 & -1 \end{bmatrix}$$

- Ahora, realizamos el criterio de factibilidad:

$$\bar{b} \ = X_B = B^{-1} \cdot b = \begin{bmatrix} 9 \\ 4 \\ 1 \end{bmatrix}$$

  Como $\bar{b} \geqslant 0$ entonces este es un punto factible.

- Realizamos el criterio de optimalidad. Dado que :

$$C_R^\top = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}, C_B^\top = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$$

- Calculamos los costos reducidos:

$$\overline{C_R}^\top = C_R^\top - C_B^\top \cdot \bar{R} = \begin{bmatrix} -1 & -1 & 1 \end{bmatrix}$$

  Con esto, vemos que el valor asociado a la variable $x_1$ y $x_2$ es negativo, por lo elegimos que $x_2$ entre a la base.

- Por último, realizamos el criterio de salida:

$$\text{Min}_{u>0} \left\{ \frac{\bar{b}}{\bar{R}} \right\} = \text{Min}_{u>0} \begin{bmatrix} 3 & 4 & 1 \end{bmatrix}$$

  La variable que tenga asociada el valor más pequeño, no negativo y que no se indefina; es la que sale, por lo que en este caso es $t$. El punto (0,0) no es factible.

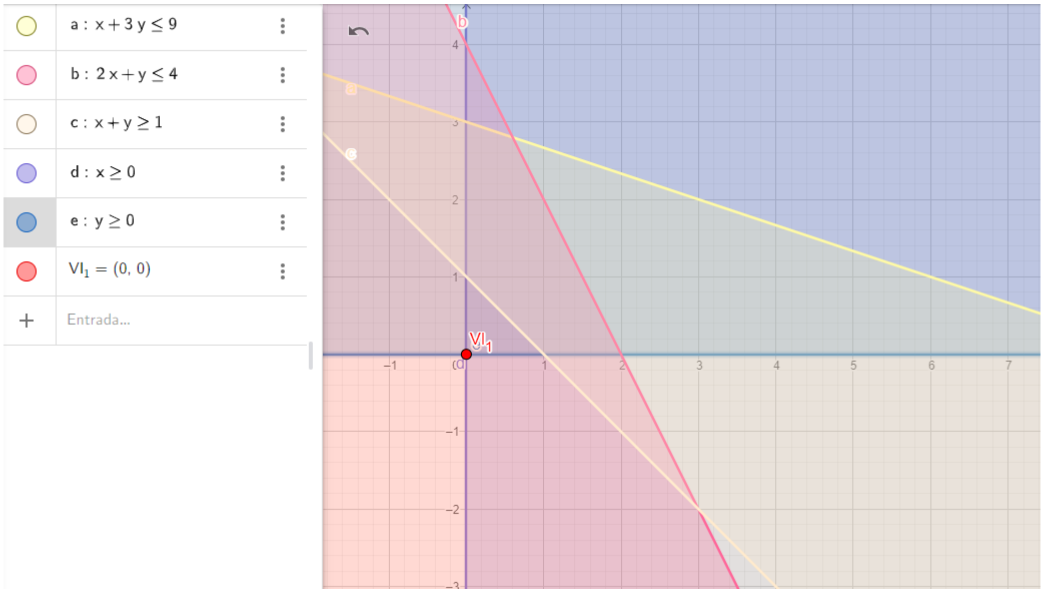



Gráfico 2 : Primera iteración Fase I con punto $= (x1 = 0, x2 = 0)$.

**Iteración 2:** Como en la iteración vimos que la variable $x_2$ y $t$ sale, ahora tenemos lo siguiente:
Base: $\{x_2, x_3, x_4\}$
Variables No básicas: $\{x_1, x_5, t\}$

- Sabiendo esto, podemos determinar las siguientes matrices:

$$B = \begin{bmatrix} 3 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}, B^{-1} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & -3 \\ 0 & 1 & -1 \end{bmatrix}$$

Luego definimos $R$ como la matriz con las variables no básicas:

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 0 & 0 \\ 1 & -1 & 1 \end{bmatrix}, \bar{R} = B^{-1} \cdot R = \begin{bmatrix} 1 & -1 & 1 \\ -2 & 3 & -3 \\ 1 & 1 & -1 \end{bmatrix}$$

- Ahora, realizamos el criterio de factibilidad:

$$\bar{b} \ = X_B = B^{-1} \cdot b = \begin{bmatrix} 1 \\ 6 \\ 3 \end{bmatrix}$$

Como $\bar{b} \geqslant 0$ entonces este es un punto factible.

- Realizamos el criterio de optimalidad. Dado que :

$$C_R^{\top} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}, C_B^{\top} = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}$$

- Calculamos los costos reducidos:

$$\overline{C_R}^{\top} = C_R^{\top} - C_B^{\top} \cdot \bar{R} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}$$

Con esto, vemos que la Base: $\{x_2, x_3, x_4\}$ es óptima y el punto es factible.

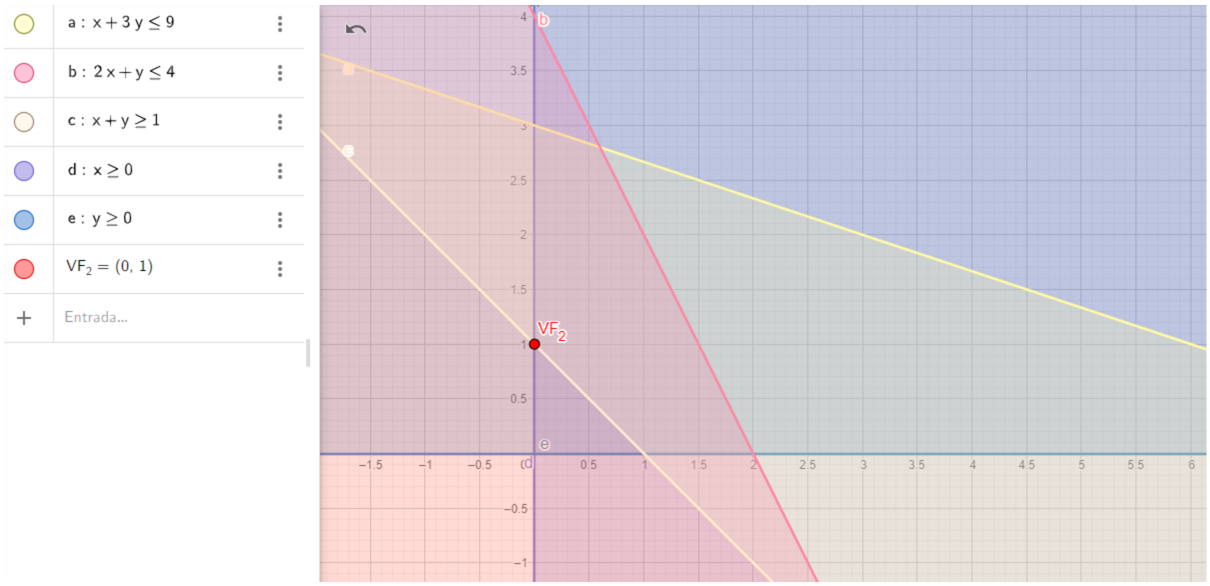



Gráfico 3 : Segunda iteración Fase I con punto $=(x1 = 0, x2 = 1)$.

## d. Simplex Fase II

Ahora para la Fase II volvemos a utilizar el problema incial en su forma estándar:

$$A = \begin{bmatrix} 1 & 3 & 1 & 0 & 0 \\ 2 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & -1 \end{bmatrix}, \; b = \begin{bmatrix} 9 \\ 4 \\ 1 \end{bmatrix}, \; C^{\top} = \begin{bmatrix} -1 & -1 & 0 & 0 & 0 \end{bmatrix}$$

**Iteración 1:**

- Como se obtuvo en la Fase I, partimos con la siguiente base:
  Base: $\{x_2, x_3, x_4\}$
  Variables No básicas: $\{x_1, x_5\}$
  Por lo que tenemos las siguientes matrices:

$$B = \begin{bmatrix} 3 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}, B^{-1} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & -3 \\ 0 & 1 & -1 \end{bmatrix}, R = \begin{bmatrix} 1 & 0 \\ 2 & 0 \\ 1 & -1 \end{bmatrix}, \bar{R} = B^{-1} \cdot R = \begin{bmatrix} 1 & -1 \\ -2 & 3 \\ 1 & 1 \end{bmatrix}$$

- Ahora, realizamos el criterio de factibilidad:

$$\bar{b} \; = X_B = B^{-1} \cdot b = \begin{bmatrix} 1 \\ 6 \\ 3 \end{bmatrix}$$

Como $\bar{b} \geqslant 0$ entonces este es un punto factible.

- Realizamos el criterio de optimalidad. Dado que :

$$C_R^\top = \begin{bmatrix} -1 & 0 \end{bmatrix}, C_B^\top = \begin{bmatrix} -1 & 0 & 0 \end{bmatrix}$$

  Calculamos los costos reducidos:

$$\overline{C_R}^\top = C_R^\top - C_B^\top \cdot \bar{R} = \begin{bmatrix} -1 & 1 \end{bmatrix}$$

  Podemos observar que el valor asociado a $x_5$ es negativo por lo que esta variable entra la base.

- Por último, realizamos el criterio de salida:

$$\text{Min}_{u>0} \left\{ \frac{\bar{b}}{\bar{R}} \right\} = \text{Min}_{u>0} \begin{bmatrix} -1 & 2 & 3 \end{bmatrix}$$

  La variable que tenga asociada el valor más pequeño, no negativo y que no se indefina; es la que sale, por lo que en este caso es $x_3$. Se concluye que el punto (0,1) no es óptimo.

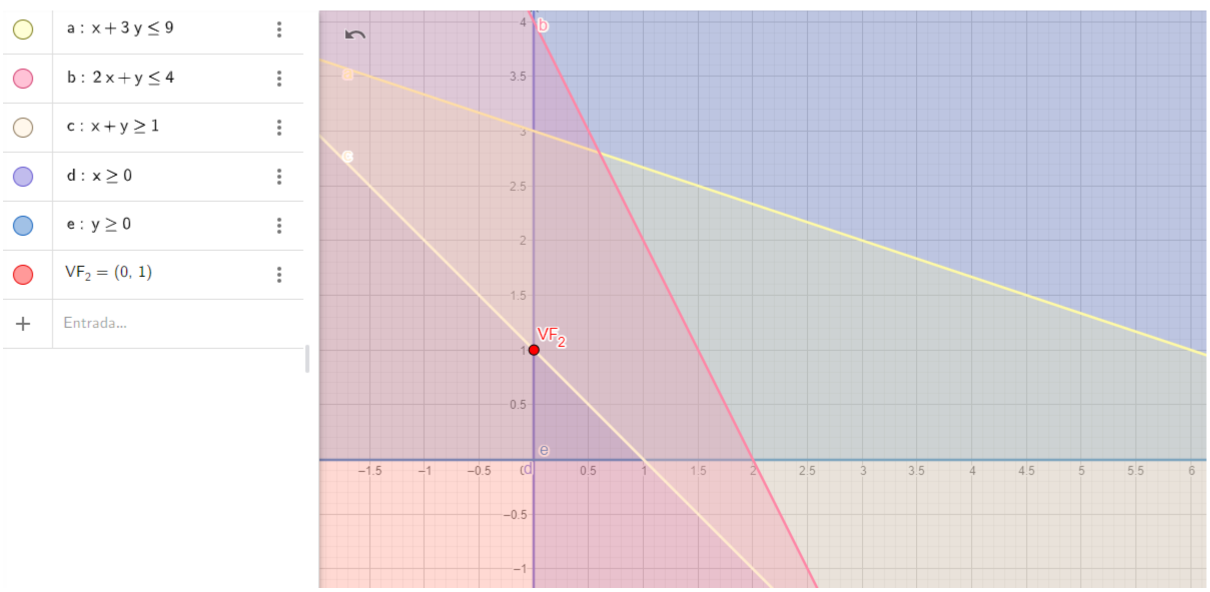



Gráfico 4 : Primera iteración Fase II con punto = $(x1 = 0, x2 = 1)$.

**Iteración 2:**

- Como se obtuvo en la iteración anterior vimos que la variable $x_5$ entra a la base y $x_3$ sale,por lo que tenemos lo siguiente para esta nueva iteración :
  Base: $\{x_2, x_4, x_5\}$
  Variables No básicas: $\{x_1, x_3\}$
  Con las siguientes matrices:

$$B = \begin{bmatrix} 3 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & -1 \end{bmatrix}, B^{-1} = \begin{bmatrix} 1/3 & 0 & 0 \\ -1/3 & 1 & 0 \\ 1/3 & 0 & -1 \end{bmatrix}, R = \begin{bmatrix} 1 & 1 \\ 2 & 0 \\ 1 & 0 \end{bmatrix}, \bar{R} = B^{-1} \cdot R = \begin{bmatrix} 1/3 & 1/3 \\ 5/3 & -1/3 \\ -2/3 & 1/3 \end{bmatrix}$$

- Ahora, realizamos el criterio de factibilidad:

$$\bar{b} = X_B = B^{-1} \cdot b = \begin{bmatrix} 3 \\ 1 \\ 2 \end{bmatrix}$$

Como $\bar{b} \geqslant 0$ entonces este es un punto factible.

- Realizamos el criterio de optimalidad. Dado que :

$$C_R^\top = \begin{bmatrix} -1 & 0 \end{bmatrix}, C_B^\top = \begin{bmatrix} -1 & 0 & 0 \end{bmatrix}$$

- Calculamos los costos reducidos:

$$\overline{C_R}^\top = C_R^\top - C_B^\top \cdot \bar{R} = \begin{bmatrix} -2/3 & 1/3 \end{bmatrix}$$

Podemos observar que el valor asociado a $x_1$ es negativo por lo que esta variable entra la base.

- Por último, realizamos el criterio de salida:

$$\text{Min}_{u>0} \left\{ \frac{\bar{b}}{\bar{R}} \right\} = \text{Min}_{u>0} \begin{bmatrix} 9 & 3/5 & -3 \end{bmatrix}$$

La variable que tenga asociada el valor más pequeño, no negativo y que no se indefina; es la que sale, por lo que en este caso es $x_4$. Se concluye que el punto $= (x_1 = 0, x_2 = 3)$ no es óptimo.

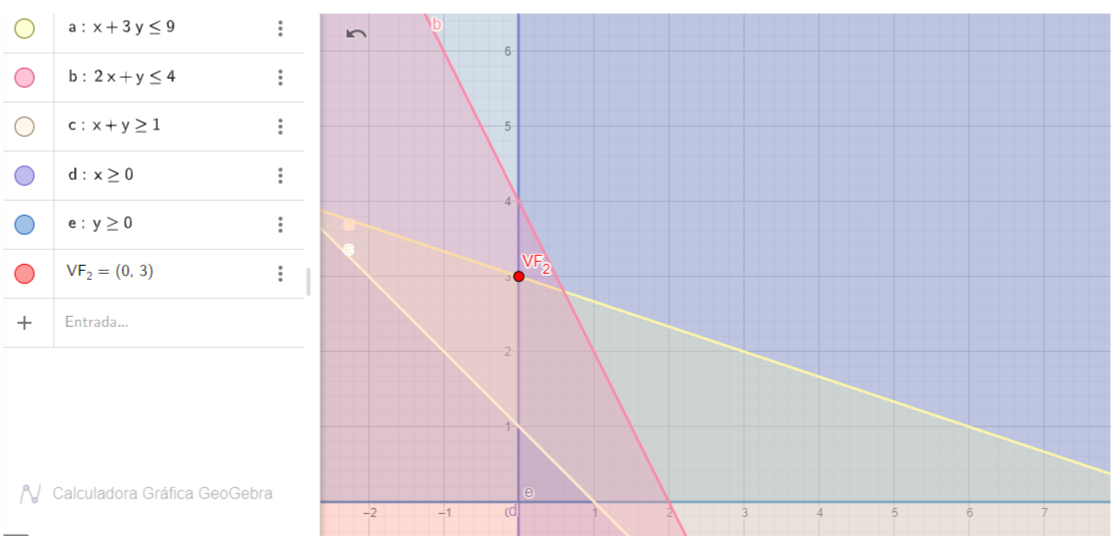



Gráfico 5 : Segunda iteración Fase II con punto $= (x1 = 0, x2 = 3)$.

**Iteración 3:**

- Como se obtuvo en la iteración anterior vimos que la variable $x_1$ entra a la base y $x_4$ sale,por lo que tenemos lo siguiente para esta nueva iteración :
  Base: $\{x_1, x_2, x_5\}$

Variables No básicas: $\{x_3, x_4\}$
Con las siguientes matrices:

$$B = \begin{bmatrix} 1 & 3 & 0 \\ 2 & 1 & 0 \\ 1 & 1 & -1 \end{bmatrix}, B^{-1} = \begin{bmatrix} -1/5 & 3/5 & 0 \\ 2/5 & -1/5 & 0 \\ 1/5 & 2/5 & -1 \end{bmatrix}, R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}, \bar{R} = B^{-1} \cdot R = \begin{bmatrix} -1/5 & 3/5 \\ 2/5 & -1/5 \\ 1/5 & 2/5 \end{bmatrix}$$

- Ahora, realizamos el criterio de factibilidad:

$$\bar{b} \ = X_B = B^{-1} \cdot b = \begin{bmatrix} 3/5 \\ 14/5 \\ 12/5 \end{bmatrix}$$

Como $\bar{b} \geqslant 0$ entonces este es un punto factible.

- Realizamos el criterio de optimalidad. Dado que :

$$C_R^\top = \begin{bmatrix} 0 & 0 \end{bmatrix}, C_B^\top = \begin{bmatrix} -1 & -1 & 0 \end{bmatrix}$$

- Calculamos los costos reducidos:

$$\overline{C_R}^\top = C_R^\top - C_B^\top \cdot \bar{R} = \begin{bmatrix} 1/5 & 2/5 \end{bmatrix}$$

Como $\overline{C_R}^\top \geqslant 0$ no es necesario seguir iterando y por lo tanto hemos llegado a una solución óptima con la Base: $x_1, x_2, x_5$ y por lo tanto el punto $= (x_1 = 3/5, x_2 = 14/5$ es un punto óptimo.

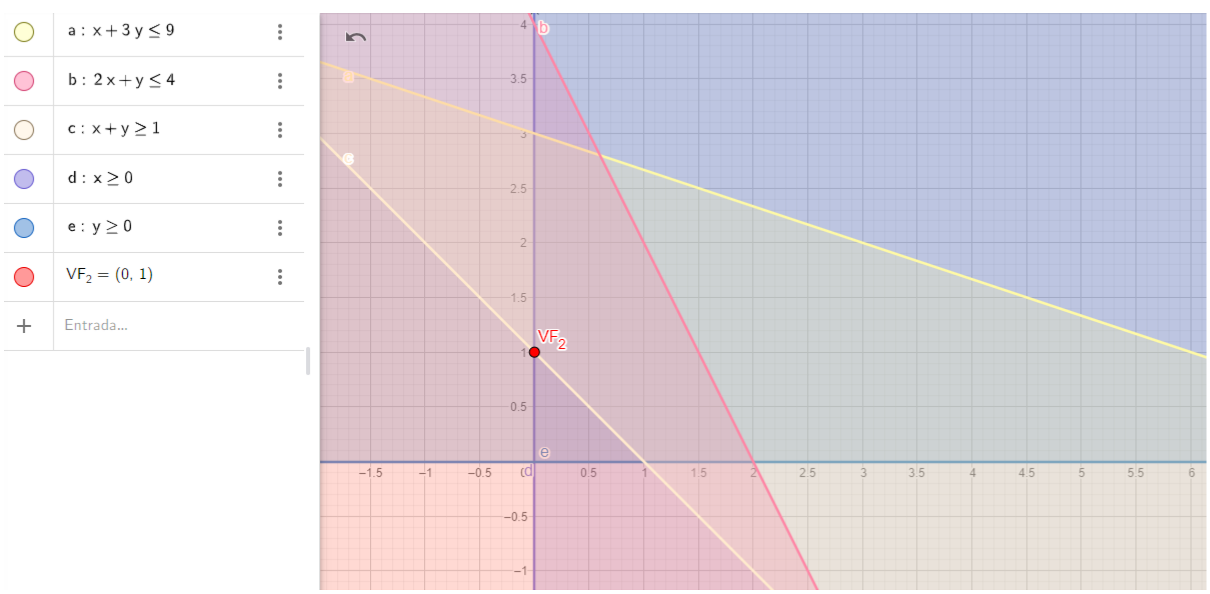



Gráfico 6 : Tercera iteración Fase II con punto $= (x1 = 3/5, x2 = 14/5)$.

La solución óptima resulta:

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} 3/5 \\ 14/5 \\ 0 \\ 0 \\ 12/5 \end{bmatrix}$$

Y el valor óptimo del problema original:

$$v* = 3,4$$

### e. Variación $\delta$

Con la restricción R2) $2x_1 + x_2 \leqslant 4 + \delta$ tenemos que:

$$b \ = \begin{bmatrix} 9 \\ 4+\delta \\ 1 \end{bmatrix}$$

Y con esto también obtenemos que :

$$\bar{b} = \begin{bmatrix} -9/5 + 3(4+\delta)/5 \\ 18/5 + (-4-\delta)/5 \\ 4/5 + 2(4+\delta)/5 \end{bmatrix}$$

Como se tiene que $\bar{b} \geqslant 0$ para que sea factible, obtenemos que $\delta \in$(-1,14).

## 2). Poliedro

### a. Existencia de escalares

Dado que $P = \{x \in \mathbb{R}^n : Ax = b, x \geq 0\}$, donde $A$ tiene $m$ restricciones $a_i^T x = b_i$ para $i = 1, \ldots, m$, y $A$ tiene filas linealmente dependientes, entonces sabemos hay una relación de dependencia lineal entre ellas. Es por esto que podemos suponer que la $m$-ésima restricción, puede ser expresada como una combinación lineal de las demás restricciones restantes $(m-1)$. Es por ello que la restricción m se puede expresar de la siguiente manera:

$$a_m = \sum_{i=1}^{m-1} \lambda_i a_i$$

donde $\lambda_i \in \mathbb{R}$ son los coeficientes de la combinación lineal.

### b. Remover restricción sin que P se vea alterado

Bajo la misma lógica de la parte a), es que si asumimos que $a_m$ puede ser descrito como una combinación lineal de otras variables, lo que conllevaría a incluir información redundante al problema. Es por ello que, sacar esta restricción no afectaría nuestro problema P).

### c. Relación del ítem anterior con el supuesto de filas LI

En el método simplex, se necesitan filas linealmente independientes para que las bases factibles en cada iteración sean únicas y se puedan definir en determinados puntos, con tal de poder avanzar por los vértices del poliedro sin recurrir a ciclos infinitos.

# 2. Pregunta 2:

## 1). Demostración 1

Por definición, vista en el Capítulo 5 con el profesor Gonzalo Pérez:

Un conjunto $S \subseteq \mathbb{R}^n$ se dice convexo si:

$$\forall x, y \in S \Rightarrow \lambda x + (1-\lambda)y \in S, \forall \lambda \in [0,1]$$


De esta forma, para este caso debemos demostrar que, sea $(x_1, x_2) \in P$, entonces para cualquier $\lambda \in [0,1]$, $\lambda x_1 + (1-\lambda)x_2 \in P$.

Sea $x_1 \in P \Rightarrow Ax_1 \leq b$ y $x_2 \in P \Rightarrow Ax_2 \leq b$, entonces podemos considerar $y = \lambda x_1 + (1-\lambda)x_2$.

Lo que nos permite hacer establecer las siguientes igualdades:

$$Ay \tag{1}$$

$$A(\lambda x_1 + (1-\lambda)x_2) \tag{2}$$

$$\lambda A x_1 + (1-\lambda)Ax_2 \tag{3}$$

Debido a las características de $x_1$ y $x_2$, vemos lo siguiente:

$$\lambda A x_1 + (1-\lambda)Ax_2 \leq \lambda b + (1-\lambda)b \leq b \tag{4}$$

$$A(\lambda x_1 + (1-\lambda)x_2) \leq b \tag{5}$$

Por lo tanto podemos demostrar que P es un conjunto convexo.

## 2). Demostración 2

Para este caso, demostraremos que no se cumplen los requerimientos con un contraejemplo.

Al no definir $z$ de la forma correcta, podemos llegar a sea el caso de 2 combinaciones de $x$ y $z$ tales que sean $(x = a, z = 0)$ y $(x = b, z = 1)$, además consideremos $\lambda = 0{,}5$.

Entonces, utilizando la misma definición del ejercicio anterior, podemos ver la siguiente secuencia:

$$\lambda \begin{pmatrix} a \\ 0 \end{pmatrix} + (1-\lambda)\begin{pmatrix} b \\ 1 \end{pmatrix} \tag{6}$$

$$\begin{pmatrix} 0{,}5a + 0{,}5b \\ 0{,}5 \end{pmatrix} \tag{7}$$

En este caso, $z$ toma valor, lo cual no es posible dentro de su definición de $z \in \{0,1\}$, por lo tanto $\begin{pmatrix} 0{,}5a + 0{,}5b \\ 0{,}5 \end{pmatrix} \notin P$.

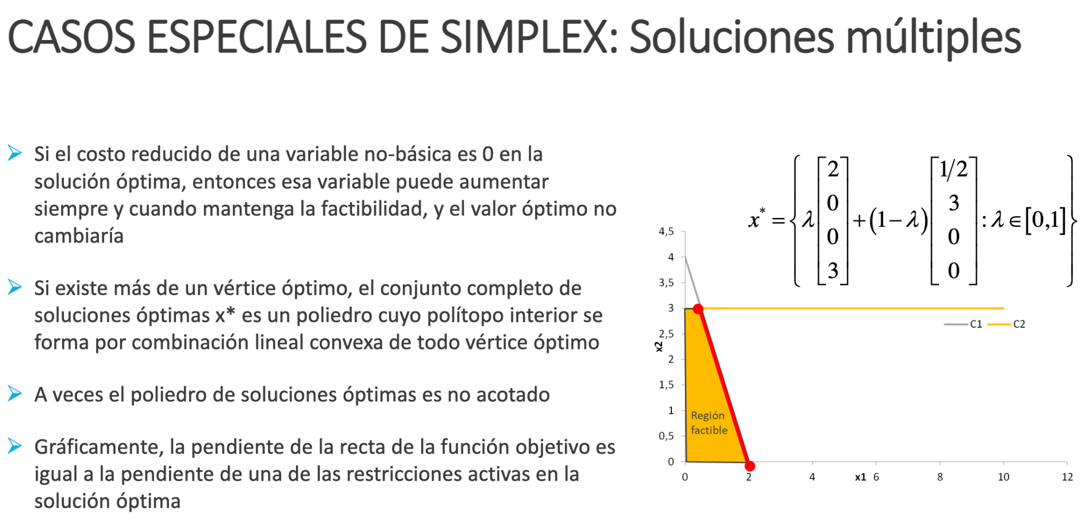
## 3). Demostración 3

Por definición sabemos que todo conjunto convexo tiene 3 posibilidades:

- No tener solución óptima
- Tener solución única
- Tener infinitas soluciones

Al tener más de una solución óptima, caemos en la tercera descripción, por lo que vemos que tiene infinitas soluciones óptimas para un único valor óptimo.

Para esto nos podemos basar en la información presentada en la Clase 18 de Camila Balbontin.




## 4). Poliedro

### a. Expresión

Podemos imaginar que la combinació posible es $3^n$, luego consideramos la cara vacía trivial y llegamos a la fórmula $3^n + 1$

### b. Demostración

Sea $P$, podemos ver que su expresión de caras es de $2^n + 1$ ya que contamos con una combinatoria de la cantida de caras posibles según la forma $2^n$, adicionalmente consideramos la cara vacía y llegamos $2^n + 1$.

## 5). $\mathbf{P} = \{x \in \mathbb{R}^n \mid Ax \geq b\} = \{x \in \mathbb{R}^n \mid a_i^\top x \geq b_i \ \forall i = 1, \ldots, m\}$

### a. Demostración $\hat{x}$ como solución óptima

### b. Suponiendo que $\hat{x}$ es vértice de P

## 3. Pregunta 3:

### a. Desarrollo Python

```python
# TAREA 2
# Jacinta Ortiz y Vicente Lavagnino

# Pregunta 3

## ______________________ CSV ______________________ ##
import pandas as pd

# CANTIDAD CUADRANTES
cuadrantes = pd.read_csv("cantidad_cuadrantes.csv", header=None).values.tolist()
cuadrantes = int(cuadrantes[0][0])
#print(cuadrantes)

# CAPACIDAD POR SACO
capacidad_por_saco = pd.read_csv("capacidad_por_saco.csv", header=None).values.tolist()

for i in range(len(capacidad_por_saco)):
    capacidad_por_saco[i] = int(capacidad_por_saco[i][0])
#print(capacidad_por_saco)

# CAPITAL INICIAL
capital_inicial = pd.read_csv("capital_inicial.csv", header=None).values.tolist()
capital_inicial = int(capital_inicial[0][0])
#print(capital_inicial)

# COSTO POR SACO HACERRRR
costos_por_saco = pd.read_csv("costo_saco.csv", header=None).values.tolist()
#print(costos_por_saco)

# KILOS FRUTA
kilos_fruta = pd.read_csv("kilos_fruta.csv", header=None).values.tolist()

for i in range(len(kilos_fruta)):
    kilos_fruta[i] = int(kilos_fruta[i][0])
#print(kilos_fruta)

# PRECIO VENTA HACERRRR
precio_venta = pd.read_csv("precio_venta.csv", header=None).values.tolist()
#print(precio_venta)


# TIEMPO DEMORA
tiempo_demora = pd.read_csv("tiempo_demora.csv", header=None).values.tolist()

for i in range(len(tiempo_demora)):
    tiempo_demora[i] = int(tiempo_demora[i][0])
#print(tiempo_demora)


## ______________________ GUROBI ______________________ ##
```

```python
from gurobipy import GRB, Model

#---------------------- Generacion del modelo ------------------------
model = Model()
model.setParam("TimeLimit", 60)  # Tiempo maximo en segundos

#---------------- Se instancian variables de decision -----------------

# Cantidades
J = len(kilos_fruta) # Cantidad de semillas
K = cuadrantes # Cantidad de cuadrantes
T = len(costos_por_saco[0]) # Cantidad de tiempos

x = model.addVars(J, K, T, vtype=GRB.BINARY, name="x")
y = model.addVars(J, K, T, vtype=GRB.BINARY, name="y")
i = model.addVars(T, vtype=GRB.CONTINUOUS, name="i")
u = model.addVars(J, T, vtype=GRB.INTEGER, name="u")
w = model.addVars(J, T, vtype=GRB.INTEGER, name="w")

#------------------ Agregar las variables al modelo -------------------
model.update()

#----------------------- Agregar Restricciones ------------------------

# Restriccion 1
for j in range(J):
    for k in range(K):
        for t in range(T):
            model.addConstr(sum(y[j, k, l] for l in range(t, min(t + tiempo_demora[j], T))) >=
            ↪  tiempo_demora[j] * x[j, k, t], "Activacion sembrado")

# Restriccion 2
for k in range(K):
    for t in range(T):
        model.addConstr(sum(y[j, k, t] for j in range(J)) <= 1, "Solo 1 sembrado por
        ↪  cuadrante")

# Restriccion 3
for t in range(1, T):
    model.addConstr(i[t] == i[t - 1] - sum(costos_por_saco[j][t] * w[j, t] for j in range(J))
    ↪  + sum(x[j, k, t - tiempo_demora[j]] * kilos_fruta[j] * precio_venta[j][t] for j in
    ↪  range(J) for k in range(K) if t - tiempo_demora[j] >= 0), "Inventario de dinero")


# Restriccion 4
model.addConstr(i[0] == capital_inicial - sum(costos_por_saco[j][0] * w[j, 0] for j in
↪  range(J)), "Condicion borde inventario dinero")


# Restriccion 5
for j in range(J):
    for t in range(1, T):
        model.addConstr(u[j, t] == u[j, t - 1] + capacidad_por_saco[j] * w[j, t] - sum(x[j, k,
        ↪  t] for k in range(K)), "Inventario semillas")

# Restriccion 6
```

```
for j in range(J):
    model.addConstr(u[j, 0] == capacidad_por_saco[j] * w[j, 0] - sum(x[j, k, 0] for k in
    ↪  range(K)), "Condicion borde semillas")

# Restriccion 7
for j in range(J):
    for k in range(K):
        for t in range(T - 1):
            model.addConstr(1 - x[j, k, t] >= sum(x[j, k, l] for l in range(t + 1, min(t +
            ↪  tiempo_demora[j], T))), "Terminar cosecha antes de volver a cosechar")

# Naturaleza de las Variables
for t in range(T):
    model.addConstr(i[t] >= 0, "Inventario de dinero positivo")

for j in range(J):
    for t in range(T):
        model.addConstr(u[j,t] >= 0, "positivo")
        model.addConstr(w[j,t] >= 0, "positivo")

#------------------------- Funcion Objetivo ---------------------------
objetivo = i[T - 1]
model.setObjective(objetivo, GRB.MAXIMIZE)
model.optimize()

#------------------------ Manejo Soluciones --------------------------\\
print("\n --------- SOLUCIÓN --------- \n")
print("Valor Objetivo (en pesos): ", int(model.objVal))


print("\n --------- PLANTACIONES EN CADA CUADRANTE --------- \n")

for k in range(K):
        plantaciones = int(sum(x[j, k, t].X for j in range(J) for t in range(T)))
        print(f"En el cuadrante {k+1}: se plantó {plantaciones} veces")

print("\n --------- CALENDARIO --------- \n")

calendario = pd.DataFrame(index=range(K), columns=range(T))

for k in range(K):
        for t in range(T):
            semillas_plantadas = [j for j in range(J) if x[j, k, t].X > 0]
            if semillas_plantadas:
                calendario.at[k, t] = semillas_plantadas[0]
            else:
                calendario.at[k, t] = 0
print(calendario)
```

## b. Resultados Obtenidos

Primero veremos la respuesta entregada debido al procesamiento del Software:

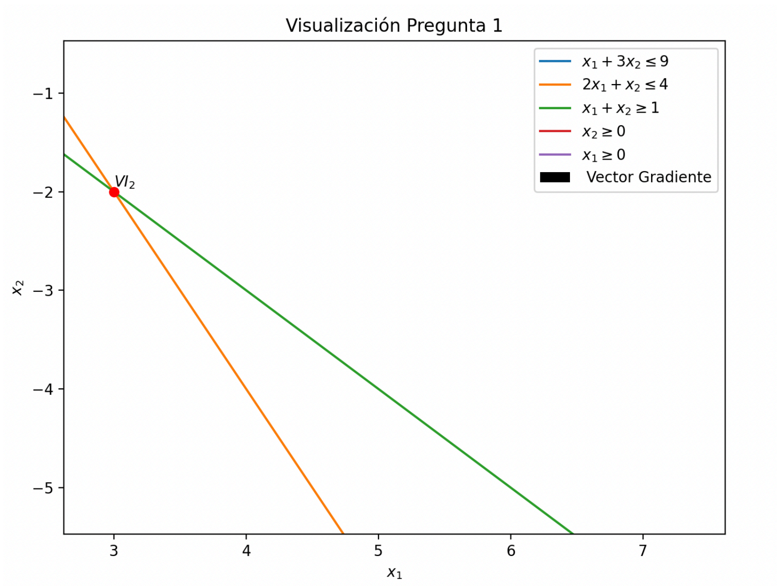
```
vicentelavagnino@Vicentes-MacBook-Pro ~/L/C/O/S/2/I/T/T/code> python3 main.py
Set parameter Username
Academic license - for non-commercial use only - expires 2025-04-08
Set parameter TimeLimit to value 60
Gurobi Optimizer version 11.0.1 build v11.0.1rc0 (mac64[arm] - Darwin 23.1.0 23B2073)

CPU model: Apple M3 Pro
Thread count: 11 physical cores, 11 logical processors, using up to 11 threads

Optimize a model with 439 rows, 363 columns and 1339 nonzeros
Model fingerprint: 0xc93ab897
Variable types: 11 continuous, 352 integer (264 binary)
Coefficient statistics:
  Matrix range     [1e+00, 2e+04]
  Objective range  [1e+00, 1e+00]
  Bounds range     [1e+00, 1e+00]
  RHS range        [1e+00, 2e+01]
Found heuristic solution: objective -0.0000000
Presolve removed 374 rows and 190 columns
Presolve time: 0.00s
Presolved: 65 rows, 173 columns, 552 nonzeros
Variable types: 0 continuous, 173 integer (108 binary)
Found heuristic solution: objective 32400.000000

Root relaxation: objective 1.584900e+05, 94 iterations, 0.00 seconds (0.00 work units)

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

*    0     0               0    158490.00000 158490.000  0.00%     -    0s

Explored 1 nodes (94 simplex iterations) in 0.01 seconds (0.00 work units)
Thread count was 11 (of 11 available processors)

Solution count 3: 158490 32400 -0

Optimal solution found (tolerance 1.00e-04)
Best objective 1.584900000000e+05, best bound 1.584900000000e+05, gap 0.0000%
```

Ahora, las soluciones printeadas por nosotros, incluyendo el valor objetivo, las especificaciones de los cuadrantes y el calendario, dan como resultado:

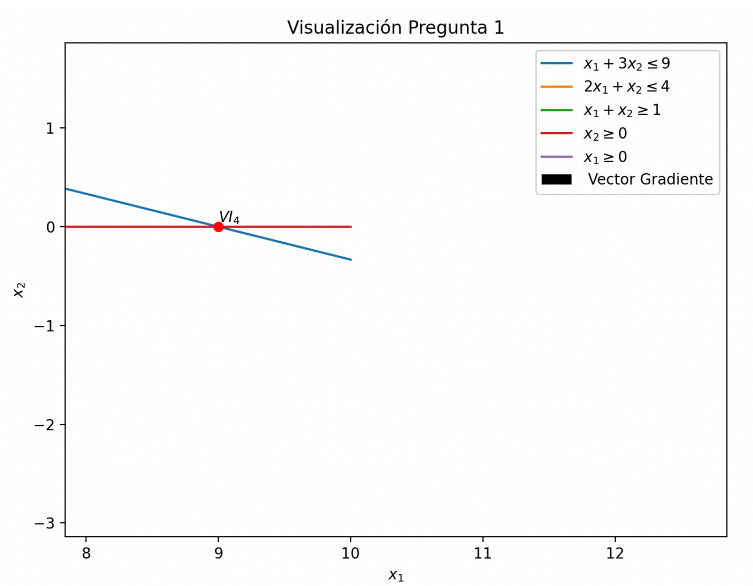
```
 --------- SOLUCIÓN ---------

Valor Objetivo (en pesos):  158490

 --------- PLANTACIONES EN CADA CUADRANTE ---------

En el cuadrante 1: se plantó 3 veces
En el cuadrante 2: se plantó 4 veces
En el cuadrante 3: se plantó 4 veces

 --------- CALENDARIO ---------

  0  1  2  3  4  5  6  7  8  9  10
0  0  0  1  0  2  0  0  2  0  0  0
1  1  0  1  0  2  0  0  2  0  0  0
2  1  0  1  0  2  0  0  2  0  0  0
```

# 4. Anexos

## 1). Anexos Pregunta 1




Vista de $VI_2$




Vista de $VI_4$

### Desarrollo Python

```
# TAREA 2 [JACINTA ORTIZ Y VICENTE LAVAGNINO]

# Pregunta 1
```

```python
import matplotlib.pyplot as plt
import numpy as np

# RANGOS
x1 = np.linspace(-10, 10, 1000)
x2_r1 = (9 - x1) / 3
x2_r2 = 4 - 2 * x1
x2_r3 = 1 - x1
x2_r4 = 0
x1_r5 = 0

# EJES Y RECTAS
plt.figure(figsize=(8, 6))

plt.plot(x1, x2_r1, label='$x_1 + 3x_2 \leq 9$')
plt.plot(x1, x2_r2, label='$2x_1 + x_2 \leq 4$')
plt.plot(x1, x2_r3, label='$x_1 + x_2 \geq 1$')
plt.plot(x1, x2_r4 * np.ones_like(x1), label='$x_2 \geq 0$')
plt.plot(x1_r5 * np.ones_like(x1), x1, label='$x_1 \geq 0$')

# VECTOR GRADIENTE
plt.quiver(0, 0, 1, 1, angles='xy', scale_units='xy', scale=1, color='Black', label=' Vector Gradiente')

# LEYENDAS
plt.legend()
plt.xlabel('$x_1$')
plt.ylabel('$x_2$')
plt.title('Visualización Pregunta 1')
plt.xlim(0, 5)
plt.ylim(0, 5)

# #VERTICES
vertices = [(0.6, 2.8), (2, 0), (1, 0), (0, 1), (0, 3) ]
infactibles = [(0,0), (3, -2), (0,4), (9,0)]

# PUNTOS
for i, vertice in enumerate(vertices):
    plt.plot(vertice[0], vertice[1], 'ro')
    plt.text(vertice[0], vertice[1] + 0.05, f'$VF_{i+1}$', fontsize=10)

for i, infactible in enumerate(infactibles):
    plt.plot(infactible[0], infactible[1], 'ro')
    plt.text(infactible[0], infactible[1] + 0.05, f'$VI_{i+1}$', fontsize=10)

# VISUALIZAR
plt.show()
```

---