

Universidad Nacional del Altiplano
Facultad de Ingeniería Estadística e Informática
Docente: Fred Torres Cruz
Alumno: Clyde Neil Paricahua Pari

Trabajo Encargado - Restricciones y Sistemas de Ecuaciones

Ejercicio 1: Restricciones de Desarrollo de Software

Enunciado del Problema

Un desarrollador tiene 15 horas semanales para dedicar al desarrollo de software de front-end (x) y back-end (y). Además:

- Debe dedicar al menos 5 horas al desarrollo de front-end para cumplir con los entregables del cliente.
- El tiempo total no puede exceder 15 horas por restricciones de tiempo del sprint.

Se solicita formular las restricciones, representarlas gráficamente e identificar las combinaciones posibles de tiempo a invertir en cada actividad.

Formulación Matemática

Variables de decisión:

$$x = \text{horas dedicadas al desarrollo front-end} \quad (1)$$

$$y = \text{horas dedicadas al desarrollo back-end} \quad (2)$$

Restricciones:

$$x \geq 5 \quad (\text{mínimo 5 horas front-end}) \quad (3)$$

$$x + y \leq 15 \quad (\text{tiempo total máximo}) \quad (4)$$

$$x \geq 0, \quad y \geq 0 \quad (\text{restricciones de no negatividad}) \quad (5)$$

Ecuaciones de frontera:

Para la representación gráfica, las ecuaciones de frontera son:

$$\text{Ecuación 1: } x = 5 \quad (6)$$

$$\text{Ecuación 2: } y = 15 - x \quad (7)$$

Implementación del Software

```
1 class GraficadoraTexto:
2     def __init__(self, xmin=-1, xmax=20, ymin=-1, ymax=15):
3         self.xmin = xmin
4         self.xmax = xmax
5         self.ymin = ymin
6         self.ymax = ymax
7         self.funciones = []
8
9     def preparar_expression(self, expr: str) -> str:
10        expr = expr.replace(" ", "")
11        expr = expr.replace("^", "**")
12        if expr.startswith("x"):
13            expr = "1*" + expr
14        expr = expr.replace("-x", "-1*x")
```

```

15     expr = expr.replace("+x", "+1*x")
16     expr = expr.replace("x", "*x")
17     expr = expr.replace("**x", "*x")
18     return expr
19
20 def agregar_funcion(self, expresion: str, simbolo: str):
21     expr_preparada = self.preparar_expresion(expresion)
22     self.funciones.append((expr_preparada, simbolo))
23
24 def graficar(self):
25     print("\nGráfico en Plano Cartesiano (ASCII)\n")
26     for y in range(self.ymax, self.ymin - 1, -1):
27         linea = ""
28         for x in range(self.xmin, self.xmax + 1):
29             simbolo = " "
30             intersecciones = []
31             for expr, simb in self.funciones:
32                 try:
33                     y_eval = eval(expr, {"x": x, "y": y})
34                     if abs(y - y_eval) < 0.5:
35                         intersecciones.append(simb)
36                 except:
37                     pass
38             if len(intersecciones) > 1:
39                 simbolo = "#"
40             elif len(intersecciones) == 1:
41                 simbolo = intersecciones[0]
42             elif x == 0 and y == 0:
43                 simbolo = "+"
44             elif x == 0:
45                 simbolo = "|"
46             elif y == 0:
47                 simbolo = "-"
48             linea += simbolo
49         print(linea)
50
51
52
53 if __name__ == "__main__":
54     print("Ecuación 1:  $x = 5$  (restricción mínima front-end)")
55     print("Ecuación 2:  $y = 15 - x$  (tiempo total)")
56     print()
57
58     graf = GraficadoraTexto()
59     graf.agregar_funcion("(x==5)*y", "*")
60     graf.graficar()

```

Resultado de la Ejecución

Al ejecutar el programa, se obtiene la siguiente representación gráfica:

Interpretación de la Solución Gráfica

Análisis de las Restricciones

- **Línea vertical (*):** Representa la restricción $x = 5$, que establece el mínimo de 5 horas que debe dedicarse al desarrollo front-end.
- **Línea diagonal (@):** Representa la ecuación $y = 15 - x$, que corresponde a la frontera de la restricción $x + y \leq 15$ (tiempo total máximo).

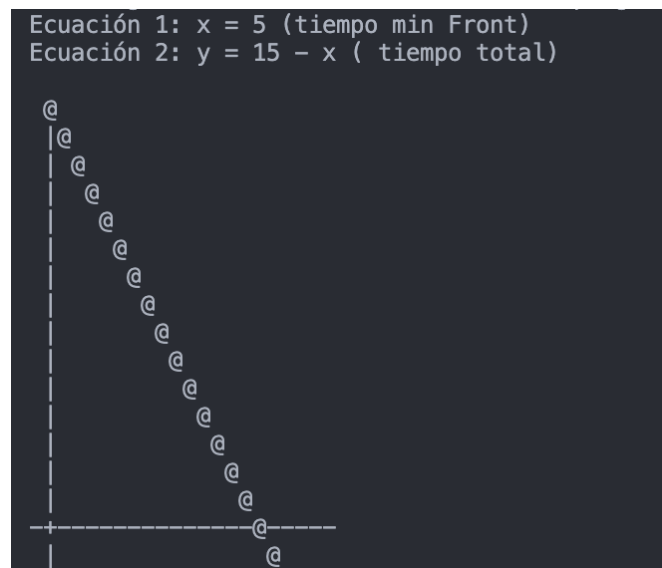


Figura 1: Compilacion Ejercicio 1

- **Intersección (#):** Indica el punto donde se cruzan ambas restricciones, ubicado en el punto $(5, 10)$.

Región Factible

La región factible está definida por el área que satisface simultáneamente todas las restricciones:

- A la derecha de la línea $x = 5$ (cumple $x \geq 5$)
- Debajo de la línea $y = 15 - x$ (cumple $x + y \leq 15$)
- En el primer cuadrante (cumple $x \geq 0, y \geq 0$)

Vértices de la Región Factible

Los vértices que delimitan la región factible son:

$$A = (5, 0) \quad 5 \text{ horas front-end, } 0 \text{ horas back-end} \quad (8)$$

$$B = (15, 0) \quad 15 \text{ horas front-end, } 0 \text{ horas back-end} \quad (9)$$

$$C = (5, 10) \quad 5 \text{ horas front-end, } 10 \text{ horas back-end} \quad (10)$$

Combinaciones Posibles

Cualquier punto (x, y) dentro de la región factible representa una asignación válida de tiempo. Algunos ejemplos de combinaciones factibles son:

- $(8, 7)$: 8 horas front-end + 7 horas back-end = 15 horas totales
- $(10, 3)$: 10 horas front-end + 3 horas back-end = 13 horas totales
- $(5, 5)$: 5 horas front-end + 5 horas back-end = 10 horas totales
- $(12, 2)$: 12 horas front-end + 2 horas back-end = 14 horas totales

Ejercicio 2: Restricciones de Presupuesto para Servidores en la Nube

Enunciado del Problema

Un ingeniero de datos administra dos tipos de servidores en la nube: Servidores A y Servidores B.

- El costo por hora de Servidor A es $S/3$. - El costo por hora de Servidor B es $S/5$. - El presupuesto máximo semanal para mantener los servidores es $S/20$.

Se solicita determinar cuántas horas puede mantener activos cada tipo de servidor, formular el sistema de ecuaciones y representarlo gráficamente.

Formulación Matemática

Variables de decisión:

$$x = \text{horas de Servidor A} \quad (11)$$

$$y = \text{horas de Servidor B} \quad (12)$$

Restricción de presupuesto:

$$3x + 5y \leq 20 \quad (13)$$

Restricciones de no negatividad:

$$x \geq 0, \quad y \geq 0 \quad (14)$$

Ecuación de frontera para graficar:

$$y = 4 - 0,6x \quad (15)$$

Implementación del Software

```
1 class GraficadoraTexto:
2     def __init__(self, xmin=0, xmax=10, ymin=0, ymax=5):
3         self.xmin = xmin
4         self.xmax = xmax
5         self.ymin = ymin
6         self.ymax = ymax
7         self.funciones = []
8
9     def preparar_expresion(self, expr: str) -> str:
10        expr = expr.replace(" ", "")
11        expr = expr.replace("^", "**")
12        if expr.startswith("x"):
13            expr = "1*" + expr
14        expr = expr.replace("-x", "-1*x")
15        expr = expr.replace("+x", "+1*x")
16        expr = expr.replace("x", "*x")
17        expr = expr.replace("**x", "*x")
18        return expr
19
20    def agregar_funcion(self, expresion: str, simbolo: str):
21        expr_preparada = self.preparar_expresion(expresion)
22        self.funciones.append((expr_preparada, simbolo))
23
24    def graficar(self, sombrear=False):
25        for y in range(self.ymax, self.ymin - 1, -1):
26            linea = ""
27            for x in range(self.xmin, self.xmax + 1):
28                simbolo = " "
29                intersecciones = []
30
```

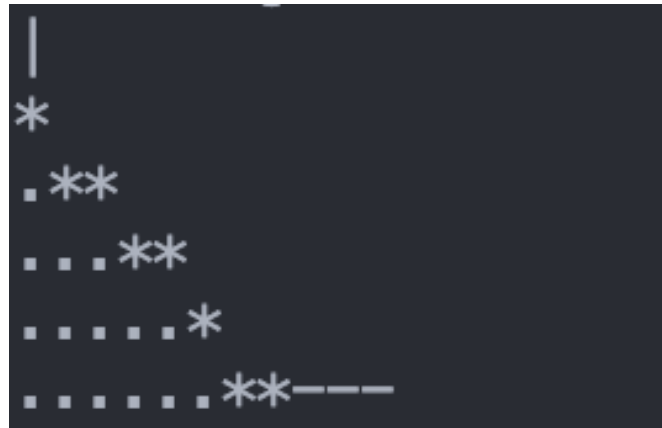


Figura 2: Ejecucion de Ejercicio 02:

```

31     if sombrear:
32         if 3*x + 5*y <= 20 and x >= 0 and y >= 0:
33             simbolo = "."
34
35     for expr, simb in self.funciones:
36         try:
37             y_eval = eval(expr, {"x": x})
38             if abs(y - y_eval) < 0.5:
39                 intersecciones.append(simb)
40         except:
41             pass
42
43     if len(intersecciones) > 1:
44         simbolo = "#"
45     elif len(intersecciones) == 1:
46         simbolo = intersecciones[0]
47     elif simbolo == " ":
48         if x == 0 and y == 0:
49             simbolo = "+"
50         elif x == 0:
51             simbolo = "|"
52         elif y == 0:
53             simbolo = "—"
54
55     linea += simbolo
56     print(linea)
57
58
59 if __name__ == "__main__":
60     graf = GraficadoraTexto(xmin=0, xmax=10, ymin=0, ymax=5)
61
62     f1 = "4 - 0.6*x"
63
64     graf.agregar_funcion(f1, "*")
65     graf.graficar(sombrear=True)

```

Resultado de la Ejecución

Al ejecutar el programa, se obtiene una representación ASCII del plano cartesiano donde:

- **Símbolo ***: Representa la frontera de presupuesto $y = 4 - 0,6x$. - **Símbolo .**: Representa la región factible donde se cumplen $3x + 5y \leq 20$ y $x, y \geq 0$. - **Ejes**: Se representan con | y — y el origen con +.

Interpretación de la Solución Gráfica

- El punto donde la recta corta el eje Y: $x = 0 \implies y = 4$ horas.
- El punto donde la recta corta el eje X: $y = 0 \implies x \approx 6,67$ horas.
- Cualquier punto dentro de la región sombreada representa una combinación de horas factible para los servidores A y B que no excede el presupuesto.

Ejercicio 3: Organización del Tiempo del Administrador de Proyectos

Enunciado del Problema

Un administrador de proyectos tecnológicos organiza su tiempo entre:

- Reuniones con stakeholders (x)
- Trabajo en la documentación técnica (y)

Condiciones:

- Reuniones: al menos 4 horas semanales ($x \geq 4$)
- Documentación: al menos 6 horas semanales ($y \geq 6$)
- Tiempo total disponible: 12 horas ($x + y \leq 12$)

Se solicita determinar la región factible y analizar las combinaciones posibles de tiempo.

Formulación Matemática

Variables de decisión:

$$x = \text{horas semanales en reuniones} \quad (16)$$

$$y = \text{horas semanales en documentación} \quad (17)$$

Restricciones:

$$x \geq 4 \quad (18)$$

$$y \geq 6 \quad (19)$$

$$x + y \leq 12 \quad (20)$$

$$x \geq 0, \quad y \geq 0 \quad (21)$$

Ecuaciones de frontera:

$$\text{Línea vertical: } x = 4 \quad (22)$$

$$\text{Línea horizontal: } y = 6 \quad (23)$$

$$\text{Línea diagonal: } y = 12 - x \quad (24)$$

Implementación del Software

```
1 class GraficadoraTexto:
2     def __init__(self, xmin=0, xmax=12, ymin=0, ymax=12):
3         self.xmin = xmin
4         self.xmax = xmax
5         self.ymin = ymin
6         self.ymax = ymax
7         self.funciones = []
8
9     def preparar_expresion(self, expr: str) -> str:
```

```

10     expr = expr.replace(" ", "")
11     expr = expr.replace("^", "**")
12     if expr.startswith("x"):
13         expr = "1*" + expr
14     expr = expr.replace("-x", "-1*x")
15     expr = expr.replace("+x", "+1*x")
16     expr = expr.replace("x", "*x")
17     expr = expr.replace("**x", "*x")
18     return expr
19
20 def agregar_funcion(self, expresion: str, simbolo: str):
21     expr_preparada = self.preparar_expresion(expr)
22     self.funciones.append((expr_preparada, simbolo))
23
24 def graficar(self, sombrear=False):
25     print("\nGráfico en Plano Cartesiano\n")
26     for y in range(self.ymax, self.ymin - 1, -1):
27         linea = ""
28         for x in range(self.xmin, self.xmax + 1):
29             simbolo = " "
30             intersecciones = []
31
32             if sombrear:
33                 if x >= 4 and y >= 6 and x + y <= 12:
34                     simbolo = "."
35
36             for expr, simb in self.funciones:
37                 try:
38                     y_eval = eval(expr, {"x": x})
39                     if abs(y - y_eval) < 0.5:
40                         intersecciones.append(simb)
41                 except:
42                     pass
43
44             if len(intersecciones) > 1:
45                 simbolo = "#"
46             elif len(intersecciones) == 1:
47                 simbolo = intersecciones[0]
48             elif simbolo == " ":
49                 if x == 0 and y == 0:
50                     simbolo = "+"
51                 elif x == 0:
52                     simbolo = "|"
53                 elif y == 0:
54                     simbolo = "-"
55
56             linea += simbolo
57         print(linea)
58
59
60 if __name__ == "__main__":
61     graf = GraficadoraTexto(xmin=0, xmax=12, ymin=0, ymax=12)
62
63     # Restricciones de frontera
64     graf.agregar_funcion("4", "*")          # x >= 4 (línea vertical)
65     graf.agregar_funcion("6", "@")          # y >= 6 (línea horizontal)
66     graf.agregar_funcion("12 - x", "#")     # x + y <= 12 (línea diagonal)
67
68     graf.graficar(sombrear=True)

```

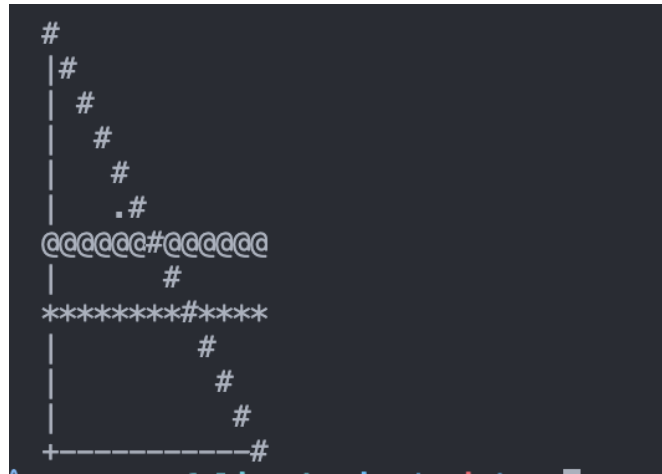


Figura 3: Ejecucion de Ejercicio 03

Interpretación de la Solución Gráfica

- La región sombreada (.) representa todas las combinaciones posibles de horas que cumplen simultáneamente las tres restricciones.
- La línea vertical (*) indica $x = 4$ (mínimo de reuniones).
- La línea horizontal (@) indica $y = 6$ (mínimo de documentación).
- La línea diagonal (#) indica $x + y = 12$ (tiempo total máximo).
- Cualquier punto dentro de la región sombreada es una asignación válida de tiempo para reuniones y documentación.

Vértices de la Región Factible

$$A = (4, 6) \quad (25)$$

$$B = (4, 8) \quad (26)$$

$$C = (6, 6) \quad (27)$$

Ejemplos de Combinaciones Posibles

- (5, 6): 5 horas reuniones + 6 horas documentación = 11 horas
- (4, 7): 4 horas reuniones + 7 horas documentación = 11 horas
- (6, 6): 6 horas reuniones + 6 horas documentación = 12 horas