UNIVERSIDAD NACIONAL DEL ALTIPLANO - PUNO

FACULTAD DE INGENIERÍA ESTADÍSTICA E INFORMÁTICA

ESCUELA PROFESIONAL DE INGENIERÍA ESTADÍSTICA E INFORMÁTICA

TRABAJO ENCARGADO

CURSO: Programacion Numerica

TEMA: Restricciones

DOCENTE: Ing. Fred Torres Cruz

PRESENTADO POR: Sadith Lina Apaza Huayta

SEMESTRE: cuarto

SECCION: “A”

PUNO – PERU

2025 – II

Ejercicios Propuestos:

Ejercicio 1:

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

• 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. Formule las restricciones, represéntelas gráficamente e identifique las combinaciones posibles de tiempo a invertir en cada actividad.

Restricciones:

- $x \geq 5$
- $x + y \leq 15$
- $x, y \geq 0$

Interpretación de la solución gráfica:

La línea x + y = 15 representa el límite máximo de horas disponibles. La región factible se encuentra debajo de esa línea y a la derecha de x = 5. Cualquier punto dentro de esa región muestra una combinación posible de tiempo entre desarrollo front-end y back-end sin superar el total de 15 horas.

Ejercicio 2:

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 y de Servidor B es S/5.

El presupuesto máximo semanal asignado para mantener los servidores es de S/20.

Determine cuantas horas puede mantener activos cada tipo de servidor, formule el

sistema de ecuaciones y represéntelo gráficamente.

Restricciones:

- $3x + 5y \leq 20$
- $x, y \geq 0$

Interpretación de la solución gráfica:

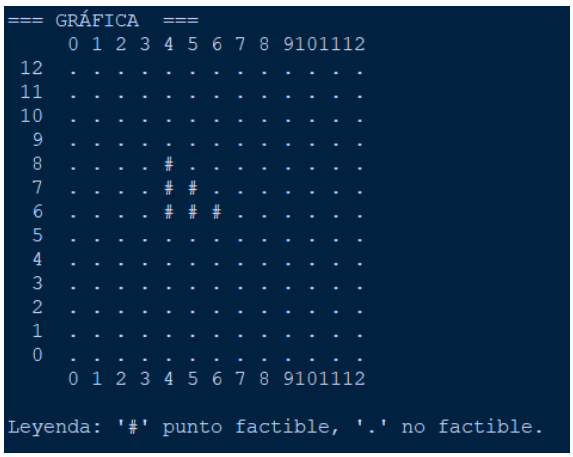
Cada punto (x, y) representa las horas de uso de los servidores A y B. La línea $3x + 5y = 20$ limita el presupuesto máximo. La región factible se encuentra debajo de esa línea, indicando las combinaciones de tiempo que no exceden el presupuesto semanal.

Ejercicio 3:

Un administrador de proyectos tecnológicos organiza su tiempo entre reuniones con stakeholders (x) y trabajo en la documentación técnica (y). Las reuniones requieren al menos 4 horas semanales y la documentación al menos 6 horas. Si dispone de 12 horas para ambas actividades, determine la región factible y analice las combinaciones posibles de tiempo.

Restricciones:

- $x \geq 4$
- $y \geq 6$
- $x + y \leq 12$



Interpretación de la solución gráfica:

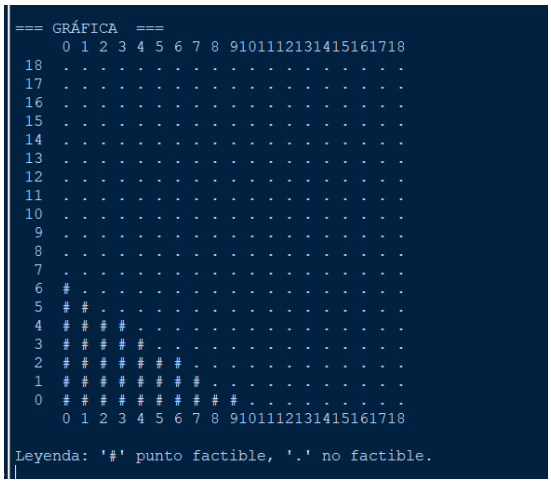
La región factible está limitada por las líneas x = 4, y = 6 y x + y = 12. Esta región representa las combinaciones posibles de tiempo entre reuniones y documentación sin exceder las 12 horas disponibles.

Ejercicio 4:

Una empresa de desarrollo de videojuegos produce dos tipos de assets: Modelos 3D (P1) y Texturas (P2). Cada modelo 3D requiere 2 horas de trabajo y cada textura requiere 3 horas. El equipo de arte tiene un total de 18 horas disponibles semanalmente. Formule las restricciones, represéntelas gráficamente y determine cuantos assets de cada tipo pueden producirse en función del tiempo disponible.

Restricciones:

- $2x + 3y \leq 18$
- $x, y \geq 0$



Interpretación de la solución gráfica:

Cada punto representa una posible cantidad de modelos 3D (x) y texturas (y) producidas. La región factible está debajo de la línea 2x + 3y = 18, indicando las combinaciones posibles que el equipo puede producir con las 18 horas disponibles.

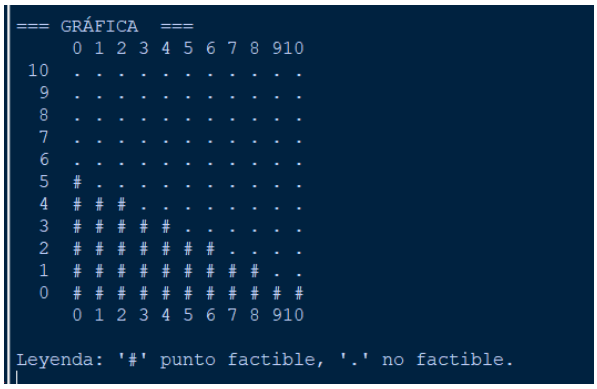
Ejercicio 5:

Una startup de hardware dispone de un máximo de 50 unidades de componentes electrónicos. Para ensamblar un dispositivo tipo A se necesitan 5 unidades y para un dispositivo tipo B se necesitan 10 unidades. Determine cuantos dispositivos de cada tipo puede ensamblar sin exceder las 50 unidades de componentes. Formule el problema, resuélvalo gráficamente y explique las posibles combinaciones de

producción.

Restricciones:

- $5x + 10y \leq 50$
- $x, y \geq 0$



Interpretación de la solución gráfica:

Cada punto indica una combinación posible de dispositivos tipo A y B ensamblados. La línea 5x + 10y = 50 es el límite de componentes. La región factible está debajo de esa línea, mostrando cuántos dispositivos pueden producirse sin exceder los recursos disponibles.

CÓDIGO

```
class Restriccion:
    def __init__(self, expresion):
        self.expresion = expresion.replace(" ", "")

    def cumple(self, x, y):
        expr = self.expresion.replace("x", str(x)).replace("y", str(y))
        try:
            return eval(expr)
        except Exception as e:
            print("Error evaluando la restricción:", self.expresion, "->", e)
            return False

class ProblemaASCII:
```

```
def __init__(self):
    self.restricciones = []

def ingresar_datos(self):
    n = int(input("¿Cuántas restricciones?: ").strip())
    for i in range(n):
        r = input(f"Restricción #{i+1}: ")
        self.restricciones.append(Restriccion(r))

    print("\n--- RANGOS ---")
    self.x_min = int(input("x mínimo : ").strip())
    self.x_max = int(input("x máximo : ").strip())
    self.y_min = int(input("y mínimo : ").strip())
    self.y_max = int(input("y máximo : ").strip())

    paso = input("Paso de la cuadrícula (enteros, por defecto 1): ").strip()
    self.step = int(paso) if paso != "" else 1

def punto_factible(self, x, y):
    for r in self.restricciones:
        if not r.cumple(x, y):
            return False
    return True

def listar_puntos_factibles(self):
    factibles = []
    for y in range(self.y_min, self.y_max + 1, self.step):
        for x in range(self.x_min, self.x_max + 1, self.step):
            if self.punto_factible(x, y):
                factibles.append((x, y))
    return factibles
```

```
def imprimir_ascii(self):
    xs = list(range(self.x_min, self.x_max + 1, self.step))
    ys = list(range(self.y_min, self.y_max + 1, self.step))

    print("\n=== GRÁFICA  ===")
    header = "    "
    for x in xs:
        if (x - self.x_min) % (max(1, (len(xs)//10))) == 0:
            header += f"{str(x).rjust(2)}"
        else:
            header += "  "
    print(header)

    for y in reversed(ys):
        line = f"{str(y).rjust(3)} "
        for x in xs:
            if self.punto_factible(x, y):
                line += " #"
            else:
                line += " ."
        print(line)

    footer = "    "
    for x in xs:
        if (x - self.x_min) % (max(1, (len(xs)//10))) == 0:
            footer += f"{str(x).rjust(2)}"
        else:
            footer += "  "
    print(footer)
    print("\nLeyenda: '#' punto factible, '.' no factible.")

def ejecutar(self):
```

```
        self.ingresar_datos()
        factibles = self.listar_puntos_factibles()
        if not factibles:
            print("\nNo se encontraron puntos factibles en el rango dado.")
        else:
            print(f"\nSe encontraron {len(factibles)} puntos factibles:")
            for p in factibles[:20]:
                print(" ", p)
        self.imprimir_ascii()

def main():
    prog = ProblemaASCII()
    prog.ejecutar()

if __name__ == "__main__":
    main()
```