

## Universidad CENFOTEC



**Escuela de Software**

**Código del curso:** SOFT-01.

**Nombre del curso:** Principios de Programación 1.

**Sección:** SCV1.

**Periodo:** C3-2025.

**Docente facilitador:** Stephanie Delgado Brenes

### CONSIGNA DEL PRIMER AVANCE DEL PROGRAMA DE COMPUTADORA

#### 1. Datos generales de la actividad

<b>Tipo de actividad:</b>	Avance de proyecto.		
<b>Fecha de entrega:</b>	13 de septiembre del 2025, 11:59pm.	<b>Valor porcentual:</b>	10%.
<b>Formato de entrega:</b>	Archivo comprimido.	<b>Puntaje total:</b>	30.
<b>Individual:</b> No.	<b>Grupal:</b> Sí.		

#### 2. Instrucciones generales

1. Lea cuidadosamente las instrucciones de la actividad. En caso de tener alguna duda, puede consultar con su docente.
2. Esta actividad se desarrolla de manera grupal. Cualquier intento de plagio será sancionado de acuerdo con el reglamento académico vigente.
3. Al completar la actividad, un representante del grupo debe subir un archivo comprimido a la plataforma Moodle en el tiempo y espacio indicado por el docente. El archivo comprimido debe contener un archivo .pdf con los diagramas y la documentación solicitada y los archivos de código fuente .py requeridos para la implementación completa del programa descrito a continuación.

#### 3. Objetivos o competencias del curso que se evaluarán en la actividad de aprendizaje

<b>Objetivo general o competencia del curso</b>	Desarrollar el pensamiento lógico-computacional para la resolución de problemas bien definidos de complejidad baja, utilizando el paradigma imperativo, un lenguaje de alto nivel y un razonamiento riguroso de forma precisa.
<b>Objetivos específicos que se evalúan</b>	<ul style="list-style-type: none"> <li>• Implementar un programa de computadora en el paradigma imperativo, usando algoritmos, estructuras computacionales, criterios de calidad y</li> </ul>

	<p>buenas prácticas de programación.</p> <ul style="list-style-type: none"> <li>• Elaborar un documento que incluye los diagramas del problema y las tablas de datos: de entrada, de salida, intermedios y constantes.</li> <li>• Confeccionar el diagrama general, el diagrama explicativo, el diagrama de flujo final y el diseño de los casos de prueba.</li> <li>• Realizar un programa de computadora en un lenguaje de programación de alto nivel, de acuerdo con los criterios y estructuras vistas en el curso.</li> <li>• Aplicar las diferentes estructuras (secuenciales, condicionales, iterativas, entre otras) en la implementación computacional.</li> <li>• Verificar la funcionalidad del programa utilizando como mínimo los dos casos de prueba diseñados.</li> <li>• Utilizar buenas prácticas en la codificación del programa: robustez, nitidez del programa, el seguimiento a estándares, entre otros.</li> <li>• Realizar la explicación del programa computacional y contestar de forma correcta las preguntas propuestas por el docente.</li> <li>• Incorporar la retroalimentación que el docente brindó a los diferentes avances.</li> </ul>
--	--

#### 4. Descripción de la actividad

##### Contexto:

El programa de computadora promueve la aplicación de los fundamentos de programación estructurada, imperativa y procedimental necesarios para la construcción de algoritmos y la resolución de problemas computacionales mediante su análisis y el diseño e implementación de programas de computadora.

##### Instrucciones:

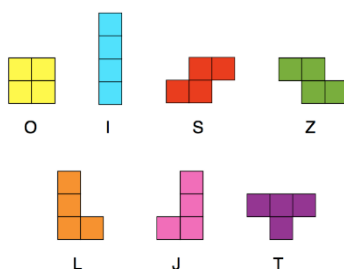
El grupo de trabajo debe implementar los siguientes programas en Python:

1. Un programa que reciba del usuario un dato de entrada con un valor numérico entero positivo y lo evalúe; si el número es menor que **20**, el programa debe imprimir un mensaje indicando que el juego se encuentra en curso; si el número es igual a **20**, el programa debe imprimir un mensaje indicando que el jugador ha perdido la partida. Si el dato ingresado por el usuario es mayor que **20**, o un número entero negativo, o un número con decimales, o cualquier otro valor que no represente un número del todo, el programa debe imprimir un mensaje indicando que el dato de entrada no es válido. El

número recibido por este programa como dato de entrada representa el apilamiento vertical de las piezas alcanzado durante la partida en curso, por lo que un valor de **20** o más es indicador de que el juego ha finalizado porque las piezas se han apilado hasta el límite superior del tablero.

2. Un programa que reciba del usuario un dato de entrada con un valor numérico entero positivo **cantidad\_lineas** y lo evalúe; si el número es menor que **0**, o si es mayor que **50**, o si es un número con decimales, o si es cualquier otro valor que no represente un número del todo, el programa debe imprimir un mensaje indicando que el dato de entrada no es válido. Si el número se encuentra entre **0** y **50**, incluyendo ambos límites, el programa debe imprimir en un mensaje adecuado el resultado de la función  $f(N) = \frac{5}{2}N(N + 1)$ , donde el valor del dato **cantidad\_lineas** se utiliza como el valor de la variable **N**. El valor resultante de la función **f(N)**, evaluado con el valor de **N** indicado por **cantidad\_lineas**, representa la cantidad de puntos acumulada por el jugador durante una partida en la que logró eliminar efectivamente dicha cantidad de líneas.

3. Un programa que reciba del usuario un dato de entrada con un valor de carácter y lo evalúe; si el carácter no es 'o', ni 'l', ni 's', ni 'z', ni 'L', ni 'J', ni 'T', o si es cualquier otro valor que no represente un carácter del todo, el programa debe imprimir un mensaje indicando que el dato de entrada no es válido. Si el carácter ingresado es alguno de los indicados antes, el programa debe imprimir la forma de la pieza correspondiente en la consola, según las formas mostradas en la Figura 1, utilizando asteriscos para representar los bloques que conforman la pieza. Las representaciones imprimibles de las siete piezas deben estar almacenadas en el programa como variables de tipo **str**.



**Figura 1.** Piezas del juego de Tetris. (Fuente:

<https://markmliu.medium.com/the-tetris-proof-60a7a69a8e04> )

## 5. Rúbrica

Esta actividad de aprendizaje será evaluada mediante la siguiente rúbrica:

Criterio de evaluación	Insuficiente (1 punto)	Aceptable (2 puntos)	Bueno (3.5 puntos)	Excelente (5 puntos)
Diagramación	Elaboración inadecuada o incompleta de los diagramas generales, explicativos o de flujo para los diferentes programas.	Elaboración superficial de los diagramas generales, explicativos o de flujo para los diferentes programas.	Elaboración básica de los diagramas generales, explicativos o de flujo para los diferentes programas.	Elaboración correcta y exhaustiva de los diagramas generales, explicativos o de flujo para los diferentes programas.
Determinación de datos del programa	Determinación inadecuada o incompleta de los datos del programa en tablas de variables de entrada, de salida, intermedias y constantes de cada programa.	Determinación parcial de los datos del programa en tablas de variables de entrada, de salida, intermedias y constantes de cada programa.	Determinación básica de los datos del programa en tablas de variables de entrada, de salida, intermedias y constantes de cada programa.	Determinación completa y ejemplificada de los datos del programa en tablas de variables de entrada, de salida, intermedias y constantes de cada programa.
Documentación de casos de prueba	Documentación incompleta o confusa del funcionamiento de cada programa.	Documentación superficial del funcionamiento de cada programa.	Documentación clara pero básica del funcionamiento de cada programa.	Documentación completa y exhaustiva del funcionamiento de cada programa.
Aplicación de buenas prácticas y estándares de programación	Buenas prácticas y estándares de programación aplicados pocas veces en cada programa.	Buenas prácticas y estándares de programación aplicados frecuentemente en cada programa.	Buenas prácticas y estándares de programación aplicados en la mayoría de cada programa.	Cumplimiento de las buenas prácticas y estándares de programación en la totalidad de cada programa.

Aplicación de lógica de programación	Lógica inadecuada de programación en cada programa.	Lógica parcialmente adecuada de programación en cada programa.	Lógica mayoritariamente adecuada de programación en cada programa.	Lógica robusta y adecuada de programación en cada programa.
Cumplimiento de requisitos	Cumple una cuarta parte o menos de los requisitos solicitados.	Cumple al menos la mitad de los requisitos solicitados.	Cumple tres cuartas partes de los requisitos solicitados.	Cumple la totalidad de los requisitos solicitados.